Team-592390

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

Lohith Bheemisetty(21BAI1603)

Bulle Likhith Raj Anvesh(21BAI1849)

Chebrolu Siddharth (21BAI1638)

J Surya Sathvik(21BRS1500)

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-592390 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 & Proposed Solution

Ideation Phase
Empathize & Discover

Date	18th of October,2023
Team ID	Team-592390
Project Name	Time Series Analysis For Bitcoin Price Prediction Using Prophet
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.

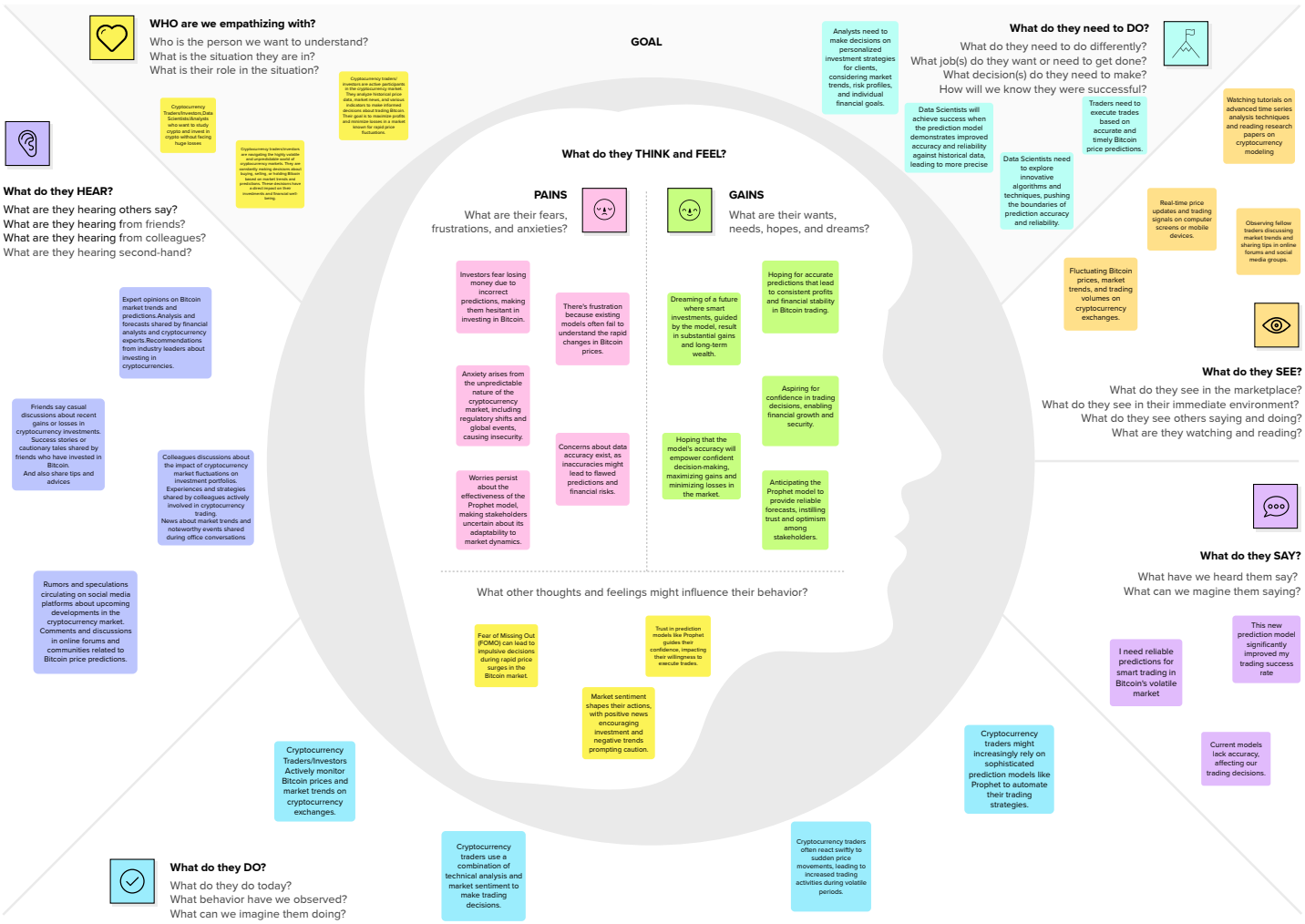
It is a useful tool to helps teams better understand their users.
Creating an effective solution requires understanding the true problem and the person who is experiencing it. The exercise of creating the map helps participants consider things from the user's perspective along with his or her goals and challenges.

Access file here:

https://app.mural.co/t/lighto3166/m/lighto3166/1697634856039/9f8edea3548d08f3754344a8d74a2386ce354778?sender=uc8d53f5bc9c73cb91d485350

Time Series analysis for Bitcoin price prediction using Prophet

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.



Ideation Phase
Brainstorm & Idea Prioritization Template

Date	19 September 2022
Team ID	Team-592390
Project Name	Time Series Analysis for Bitcoin Price Prediction Using Prophet
Maximum Marks	4 Marks

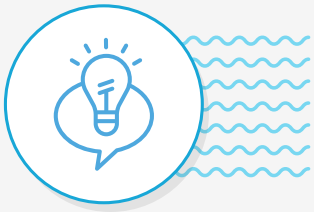
Project Brainstorming Template:

Unlocking the Future of Bitcoin Price Prediction

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/lighto3166/m/lighto3166/1697634968186/b7eab8cf109d832fad1803197fdbfcc85294b829?sender=u15d1a4cab87108fe2a483675

Template



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

1

Defining the 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

Brainstorming

CRYPTOINSIGHTS: ENHANCING BITCOIN PRICE PREDICTION AND RISK ASSESSMENT

LIKHITH RAJ ANVESH BULLE

SENTIMENT ANALYSIS INTEGRATION

Consider incorporating sentiment analysis of news articles, social media posts, and forums related to Bitcoin. Analyzing public sentiment can be a valuable addition to your model, as it may help explain sudden price movements and investor sentiment.

Machine Learning Model Comparison

Experiment with various machine learning models in addition to FbProphet, such as LSTM neural networks or ARIMA, and compare their performance in Bitcoin price prediction. This can help identify the most effective approach.

J SURYA SATHVIK

EXTERNAL SOURCES

Explore the integration of external data sources such as macroeconomic indicators, regulatory changes, and global events. These factors can significantly impact Bitcoin's price, and including them in your model may improve accuracy.

User-Friendly Interface

Focus on creating a user-friendly and visually appealing interface for your prediction platform, with easy-to-understand visualizations and tools for users to customize their predictions and portfolios.

LOHITH BHEEMISETTY

PORTFOLIO MANAGEMENT TOOL

Develop a comprehensive platform that not only predicts Bitcoin's price but also assists users in managing their cryptocurrency portfolios. This could include features for diversification, risk assessment, and automated trading strategies.

Risk Assessment and Mitigation

Develop a risk assessment module that evaluates potential risks associated with Bitcoin investments, including market volatility, regulatory changes, and security threats. Provide recommendations for risk mitigation strategies.

C ROHAN SIDDARTH

REAL TIME PREDICTIONS

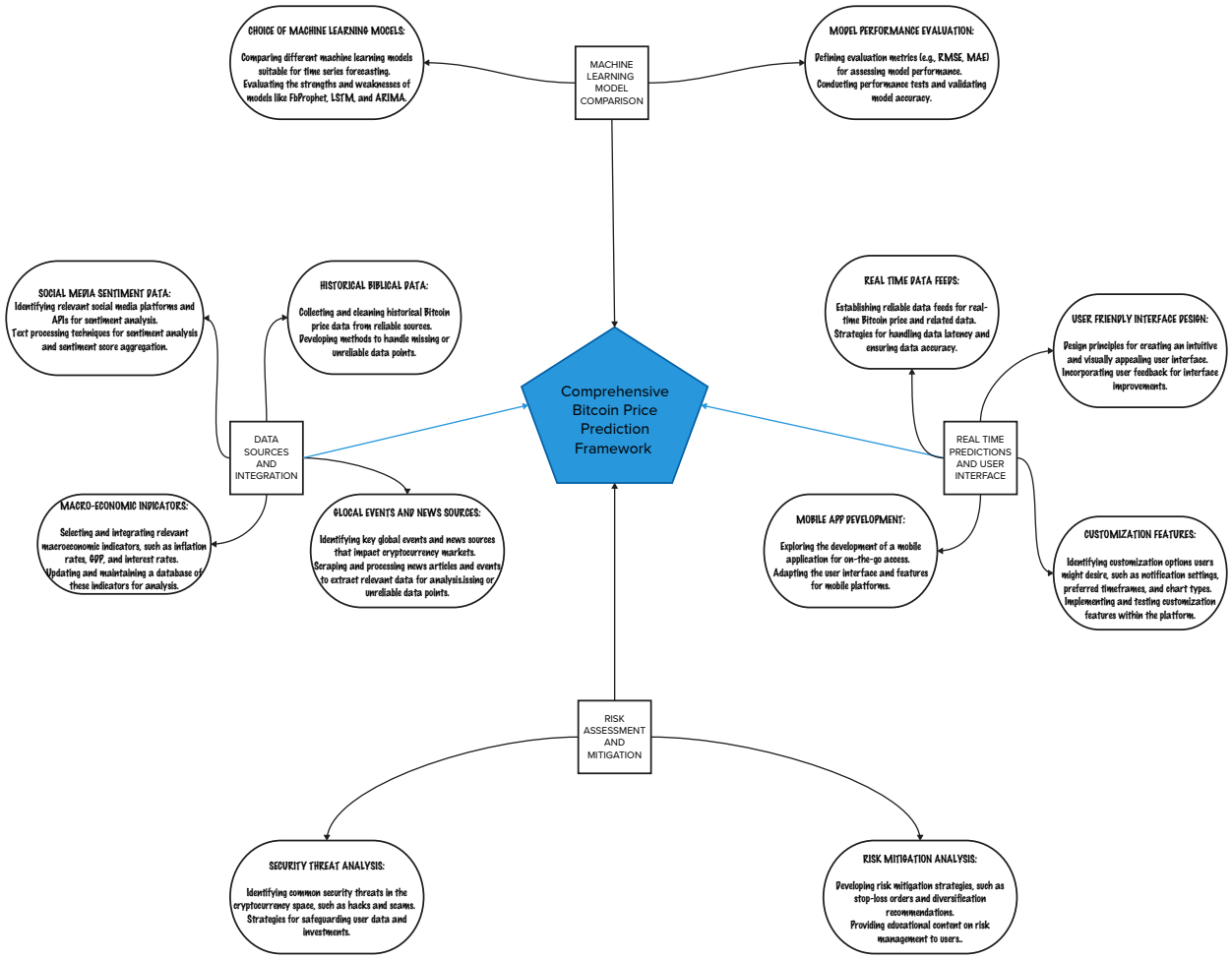
Extend the project to provide real-time or near-real-time predictions, enabling users to make timely decisions in the fast-paced cryptocurrency market. This can involve setting up automated data feeds and regular model updates

Cryptocurrency Market Trends Analysis

Extend our project to analyze and predict broader cryptocurrency market trends, not just Bitcoin. This could provide users with insights into how Bitcoin behaves in the context of the entire market.

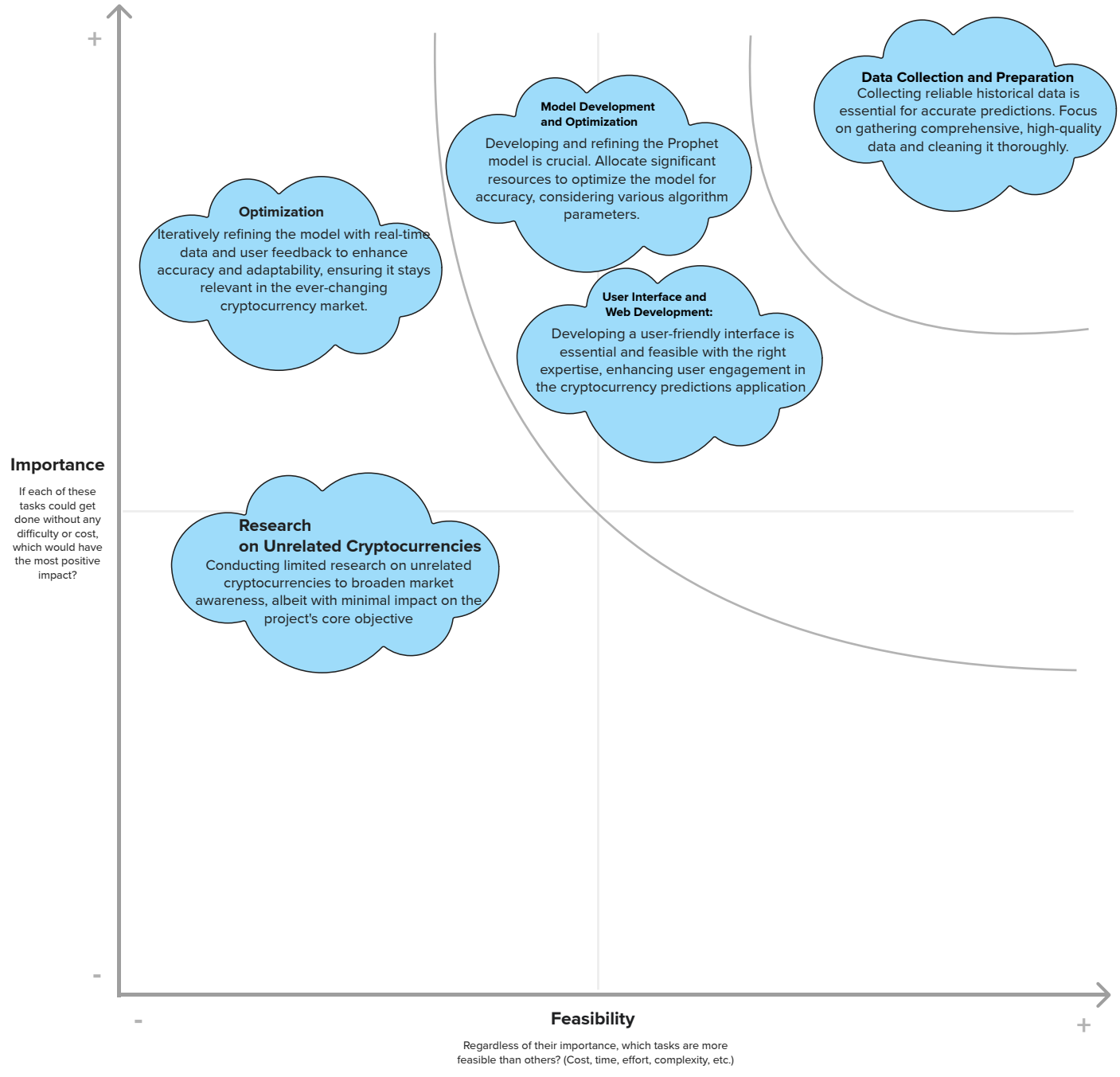
3

Group ideas clustering!



4

Prioritizing



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 Template

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 its application of FbProphet to the cryptocurrency market, specifically Bitcoin price prediction. FbProphet is known for its ability 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 / Customer Satisfaction	This solution has a significant social impact as 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.

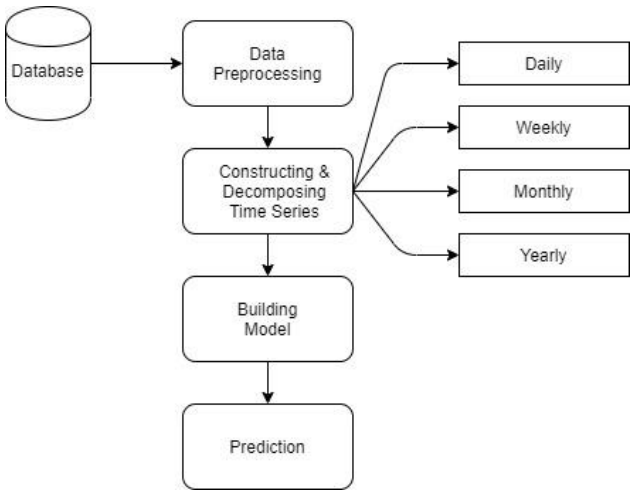
5.	Business Model (Revenue Model)	The business model for this solution could involve several revenue streams: **Subscription-based model**: Charging users a subscription fee to access premium features or more accurate predictions. **Freemium model**: Offering basic predictions for free and charging for advanced or real-time forecasting services. **Data licensing**: Selling historical and real-time Bitcoin price data to other businesses and researchers. **Consultation and advisory services**: Providing personalized advice and consultation on Bitcoin investments for a fee.
6.	Scalability of the Solution	The solution's scalability can be achieved by optimizing 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 in the digital asset space. This scalability can be achieved by investing in robust cloud infrastructure, data management, and user support systems to accommodate growing demand

Project Design Phase-II Data Flow Diagram & User Stories

Date	23-10-2023
Team ID	592390
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

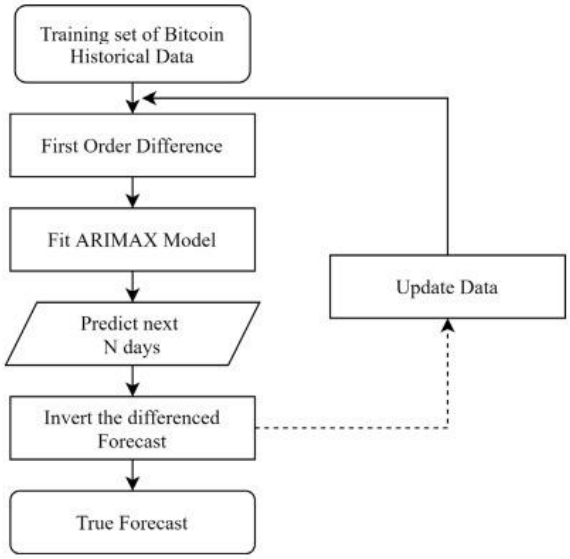


Figure 3. ARIMAX Model for BTC Prediction

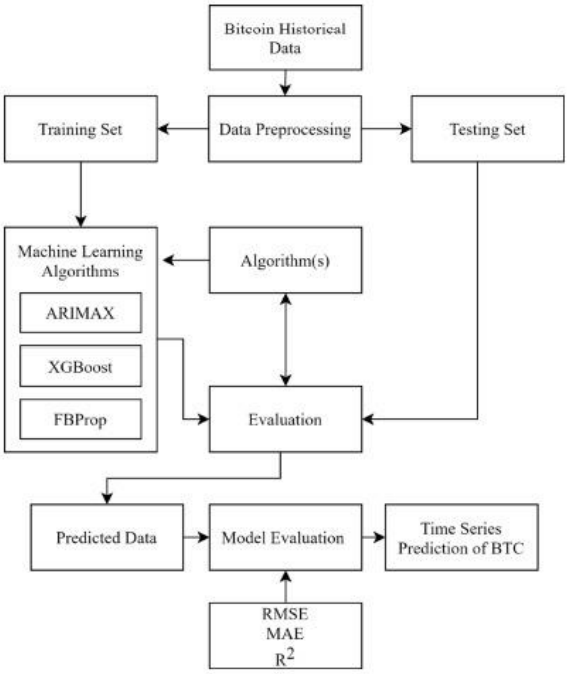


Figure 2. Basic Methodology Flowchart

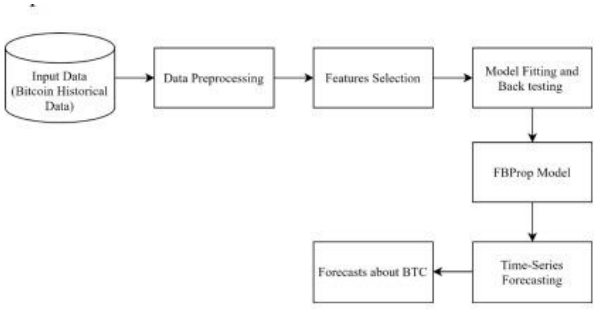


Figure 6. FBProp Algorithm for BTC

User Stories

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance Criteria	Priority	Release
Crypto Investor	Historical Data Collection	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 structured format.	High	Sprint 1

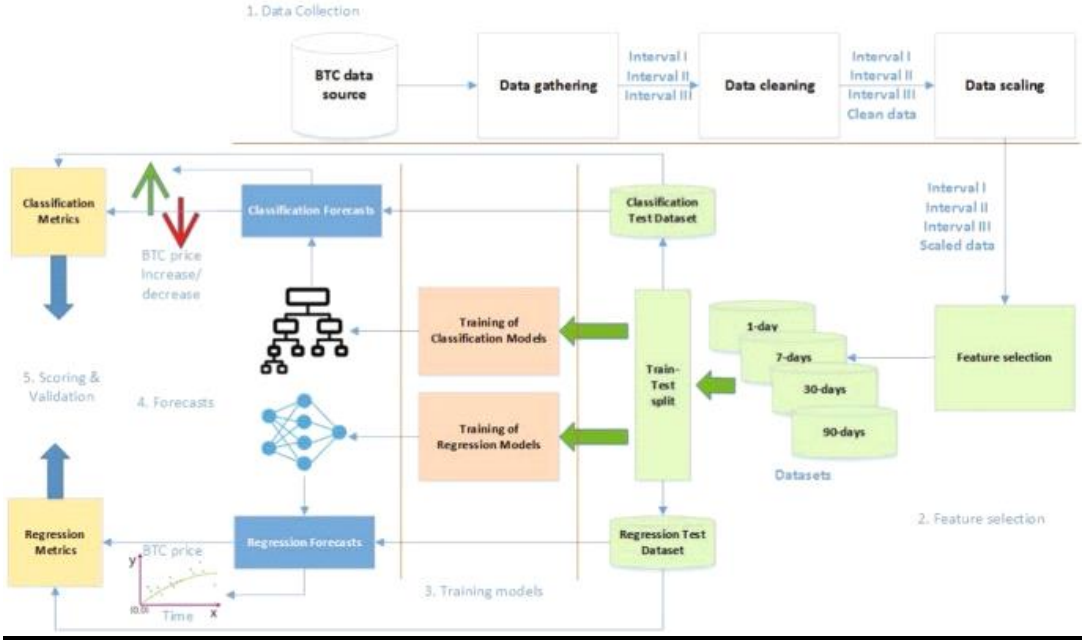
User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance Criteria	Priority	Release
Data Analyst	Data Preprocessing	USN-2	I want to preprocess the collected Bitcoin price data by cleaning, normalizing, and structuring it for time series analysis with Prophet.	Data should be cleaned and transformed into a suitable format for modelling.	High	Sprint 1
Data Scientist	Model Development and Training	USN-3	I want to select the Prophet model and train it using the pre-processed Bitcoin price dataset.	The model should be fine-tuned, and training results should meet performance criteria.	High	Sprint 2
Application Developer	Model Integration and Deployment	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.	The application should provide an easy-to-use interface for users to input their requirements.	High	Sprint 2

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance Criteria	Priority	Release
Crypto Trader	Real-Time Predictions	USN-5	I want to receive real-time Bitcoin price predictions using the integrated Prophet model to make informed trading decisions.	Real-time predictions should be available with low latency.	Medium	Sprint 3
Quality Assurance	Testing and Evaluation	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 Design Phase
Solution Architecture

Date	06 NOv 2023
Team members	Lohith Bheemisetty Bulle Likhith Raj Anvesh J Surya Sathvik C Rohan Siddarth
Project Name	Time Series Analysis For Bitcoin Price Prediction Using Prophet
Maximum Marks	**5 Marks**
Team id	**592390**

Solution Architecture



1.**Data collection and preparation:**

Data collection involves gathering relevant data from various sources such as meteorological stations, weather satellites, or public weather APIs. The data should include features like temperature, humidity, wind speed, atmospheric pressure, cloud cover, and historical rainfall measurements.

Data preprocessing includes cleaning the collected data by handling missing values, removing outliers, and ensuring consistency in formatting. It may also involve feature engineering, where additional meaningful features are derived from the raw data, such as aggregating data over specific time intervals or incorporating domain knowledge.

2.**Model selection and training:**

Model selection depends on the specific problem and the characteristics of the dataset. Common machine learning algorithms for rainfall prediction include linear regression, decision trees, random forests, gradient boosting, and neural networks. The choice of algorithm may consider factors such as interpretability, scalability, and the ability to handle complex relationships in the data.
Once the algorithm is selected, the model is instantiated and trained on the prepared dataset. The training process involves feeding the input features and corresponding rainfall measurements to the model and adjusting its internal parameters to minimize the prediction error.

3.**Model evaluation:**

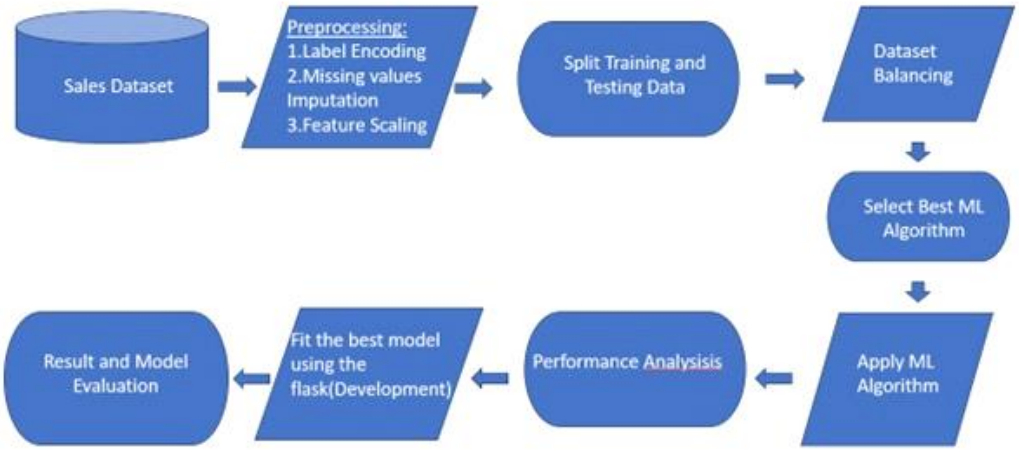
Model evaluation is crucial to assess the performance and generalization capability of the trained model. Evaluation metrics for rainfall prediction can include mean squared error (MSE), root mean squared error (RMSE), mean absolute error (MAE), or correlation coefficient. The model is evaluated on a held-out test set, which contains data that the model has not seen during training, to provide an unbiased estimate of its performance.

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**



visually represents the flow of data and the components involved in the system. It provides a high-level overview of how data is collected, processed, and used to train the model. It also illustrates the deployment of the trained model into an application or system for real-time rainfall prediction.

Reference: https://link.springer.com/article/10.1007/s00521-020-05129-6

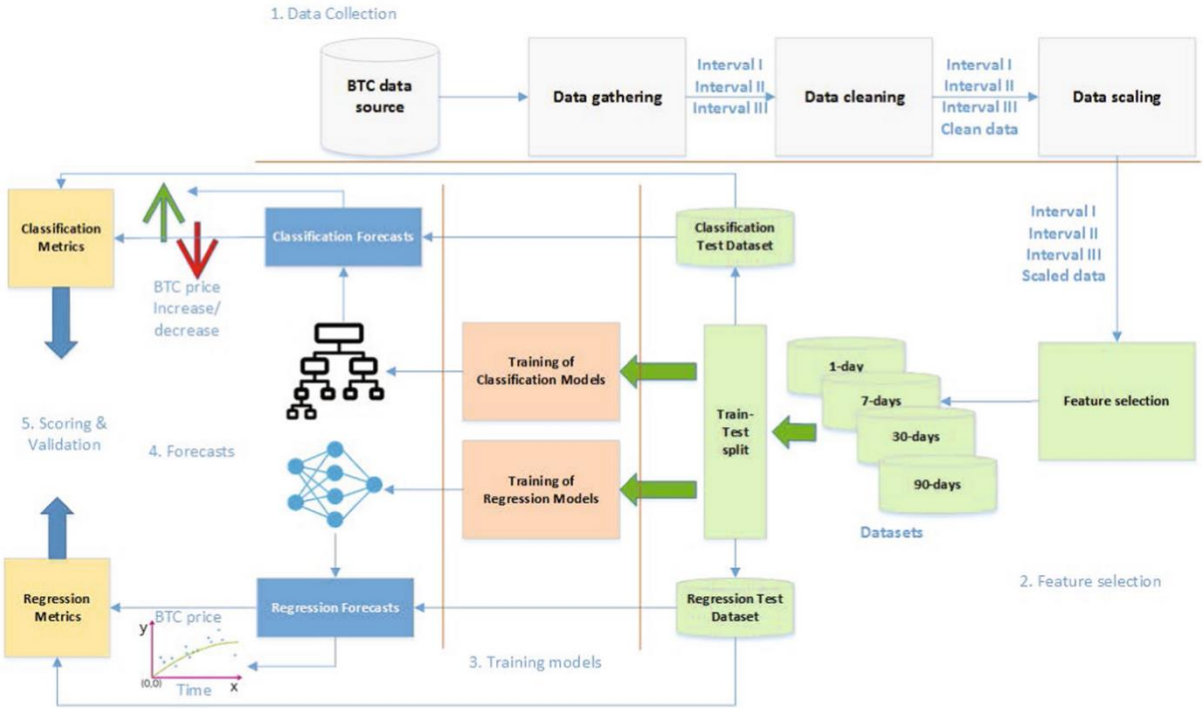
Project Planning Phase

Technology Stack

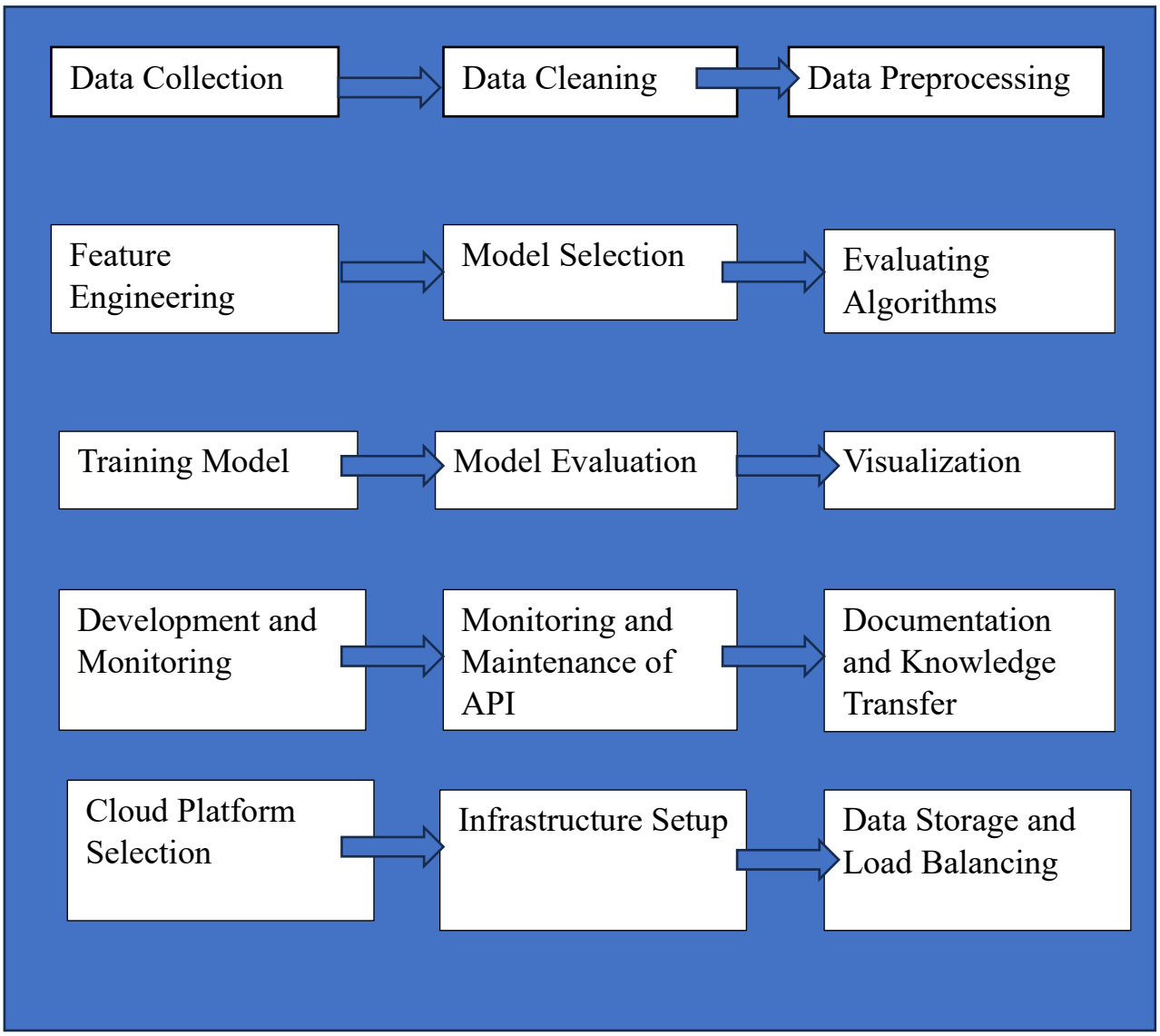
Date	27th October 2023
Team Id	Team-592390
Project Name	Time Series Analysis For Bitcoin Price Prediction Using Prophet
Maximum Marks	4
Team Members	Lohith Bheemisetty Bulle Likhith Raj Anvesh J Surya Sathvik C Rohan Siddarth

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 inflow of 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.

THANK YOU

Project Planning Phase

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

Date	28 October 2023
Team ID	Team-592390
Project Name	Time Series Analysis for Bitcoin Price Prediction Using Prophet
Maximum Marks	8 Marks

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

Use the below template to create product backlog and sprint schedule

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1,2	Data Collection and Preparation	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	Surya Sathvik
Sprint-3,4	Model Development and Training	USN-2	As a Machine Learning Engineer, I want to create a baseline Prophet model for Bitcoin price prediction, so I can establish a foundation for further optimization.	2	High	Rohan Siddarth
Sprint-5,6	Evaluation and Validation	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	Surya Sathvik, Rohan Siddarth

Sprint-7,8	Front-End Development	USN-4	As a Front-End Developer, I want to implement interactive charts on the user interface, so	5	High	Likhith Raj Anvesh, Lohith
			users can visualize both predicted and actual Bitcoin prices dynamically.			
Sprint-9,10	Cloud Infrastructure and Deployment	USN-5	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	Likhith Raj Anvesh, Lohith
Sprint-11,12	Performance Optimization	USN-6	As a QA Engineer, I want to define comprehensive test objectives and scope, so I can ensure the application functions as intended and meets quality standards.	3	Medium	Likhith Raj Anvesh, Lohith, Sathvik
Sprint-13	Evaluation & Testing	USN-7	As a quality assurance specialist, I want to test the model's performance rigorously and evaluate its effectiveness for Bitcoin price prediction.	2	High	Rohan Siddarth

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	18	1 Day	28 Oct 2023	28 Oct 2023	14	28 Oct 2023
Sprint-3,4	18	1 Day	29 Oct 2023	29 Oct 2023	18	29 Oct 2023
Sprint-5,6	18	3 Days	30 Oct 2023	01 Nov 2023	22	01 Nov 2023
Sprint-7,8,9	12	3 Days	02 Nov 2023	04 Nov 2023	12	04 Nov 2023
Sprint-10,11,12	12	2 Days	05 Nov 2023	06 Nov 2023	12	06 Nov 2023

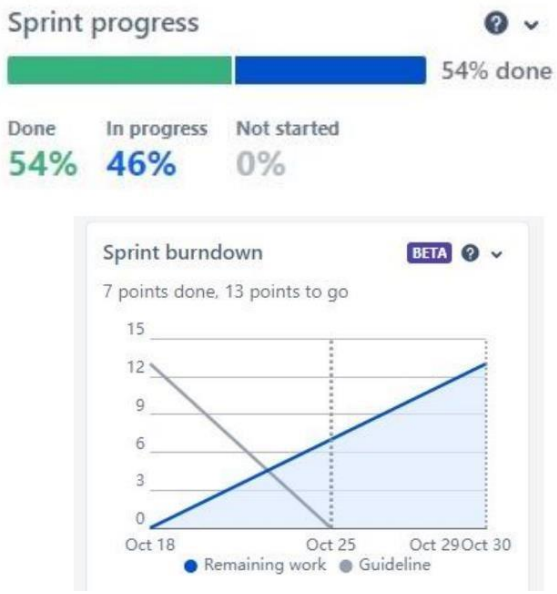
Velocity:

The team velocity of the 10-day sprint duration is:

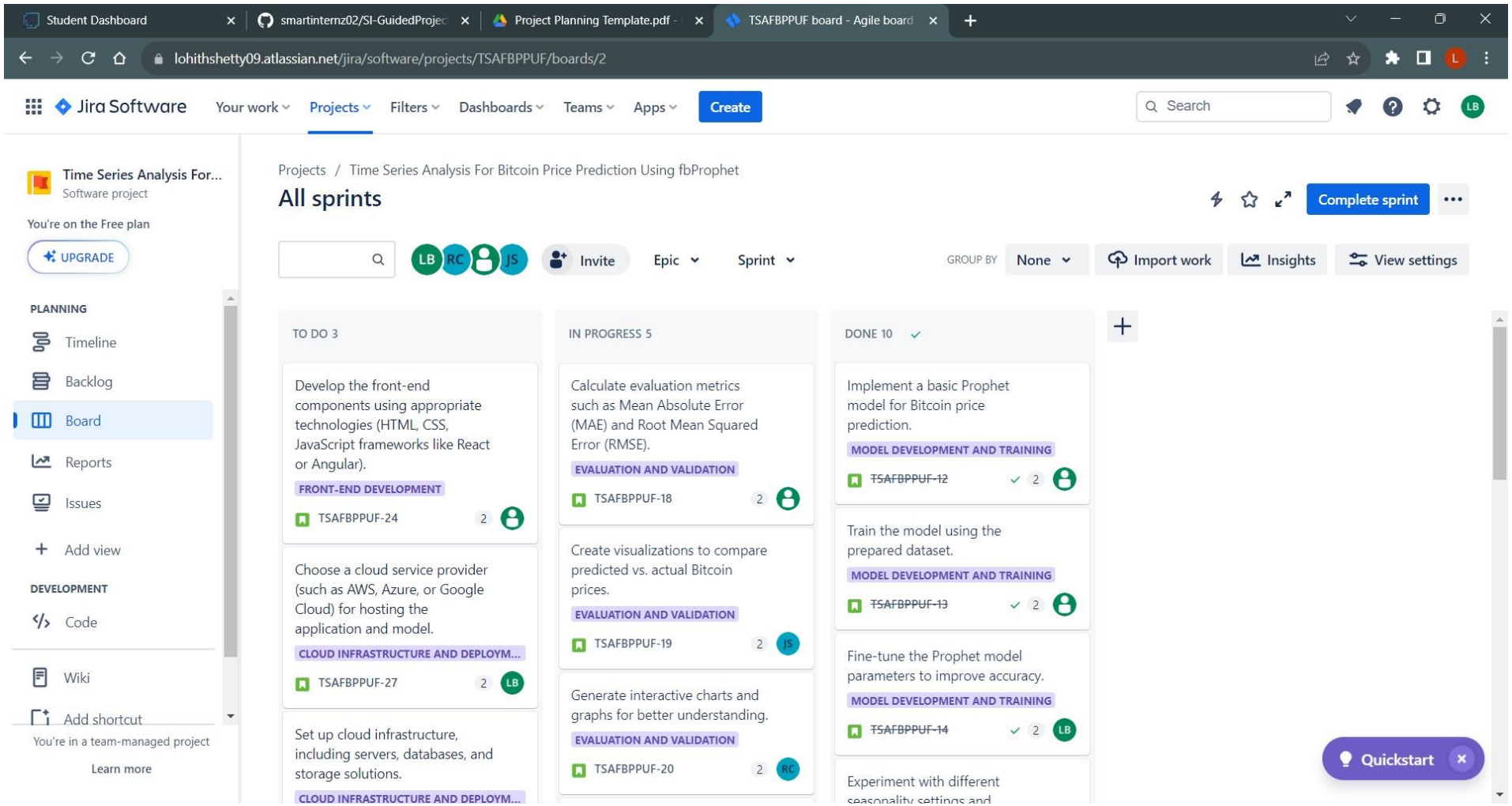
Velocity = (14+18+22+12+12) / 12 = 88 / 12 = 7.3

Average Velocity = *Sprint duration/velocity = 10/7.3* ***= 1.36***

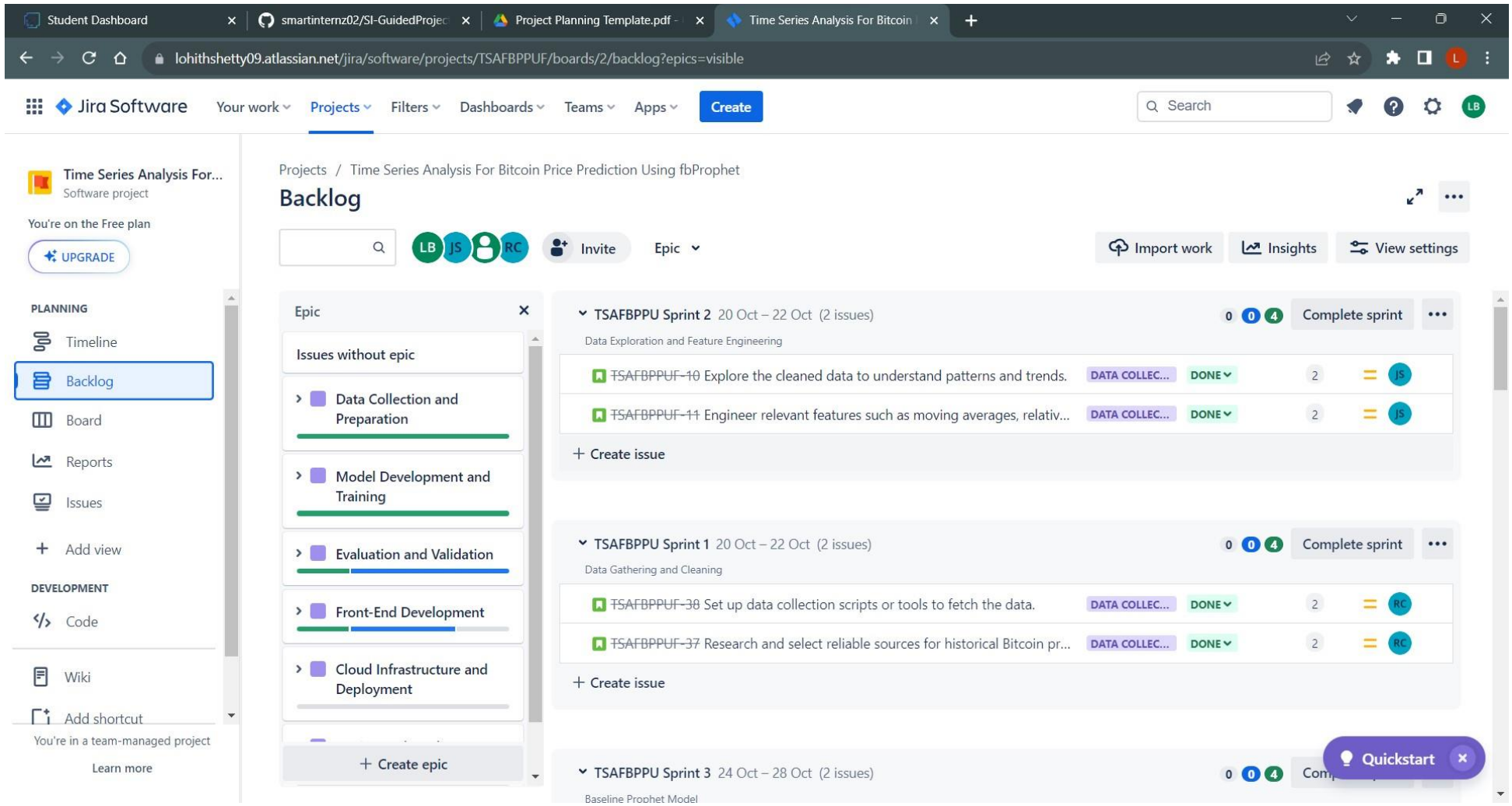
Burndown Chart: A burn down chart is a graphical representation of work left to do versus time. It is often used in agile software development methodologies such as Scrum. However, burn down charts can be applied to any project containing measurable progress over time.

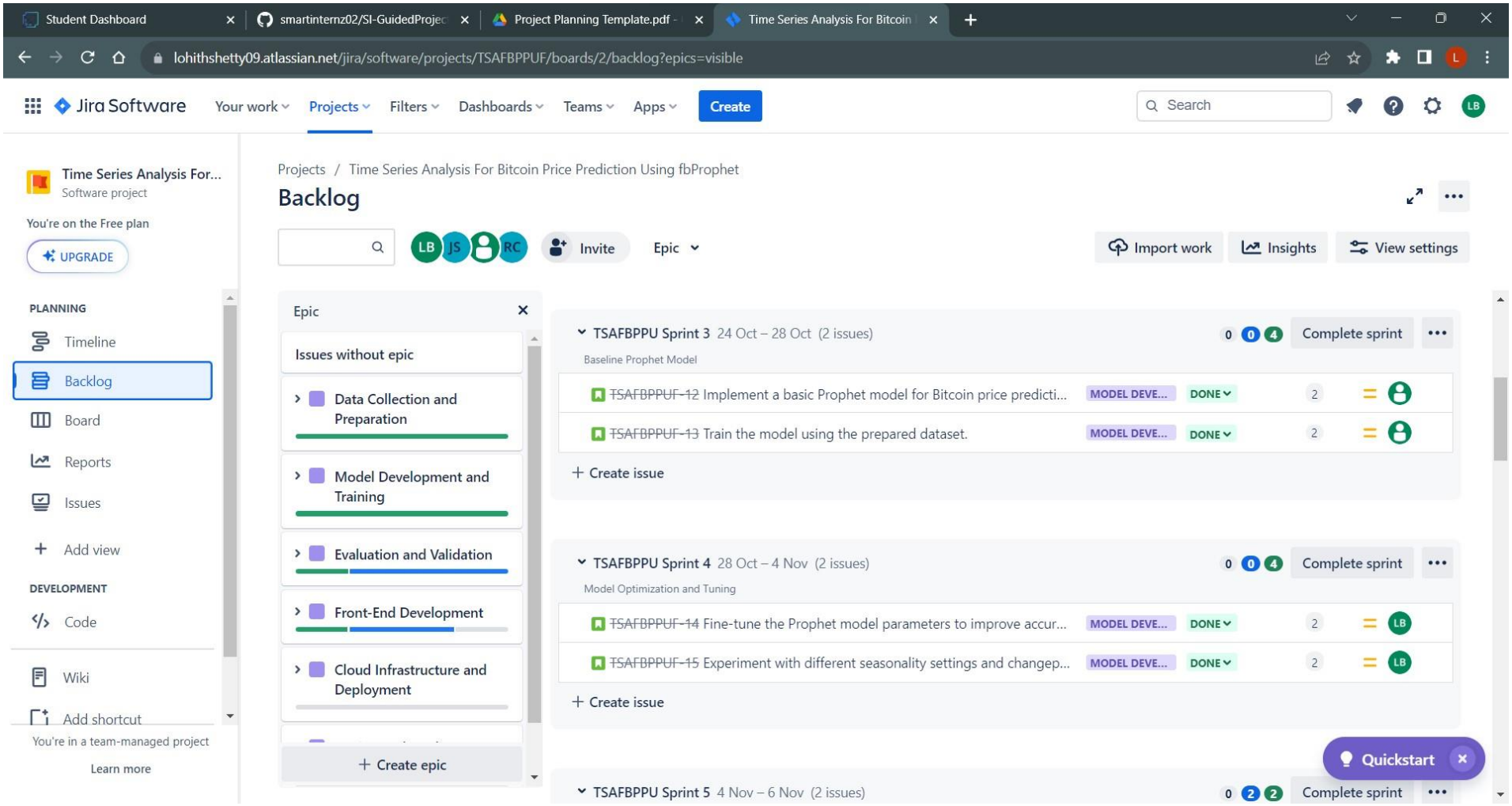

Sprint progress
54% done
Done
54%
In progress
46%
Not started
0%
Sprint burndown
BETA
7 points done, 13 points to go
15
12
9
6
3
0
Oct 18
Oct 25
Oct 29
Oct 30
Remaining work
Guideline

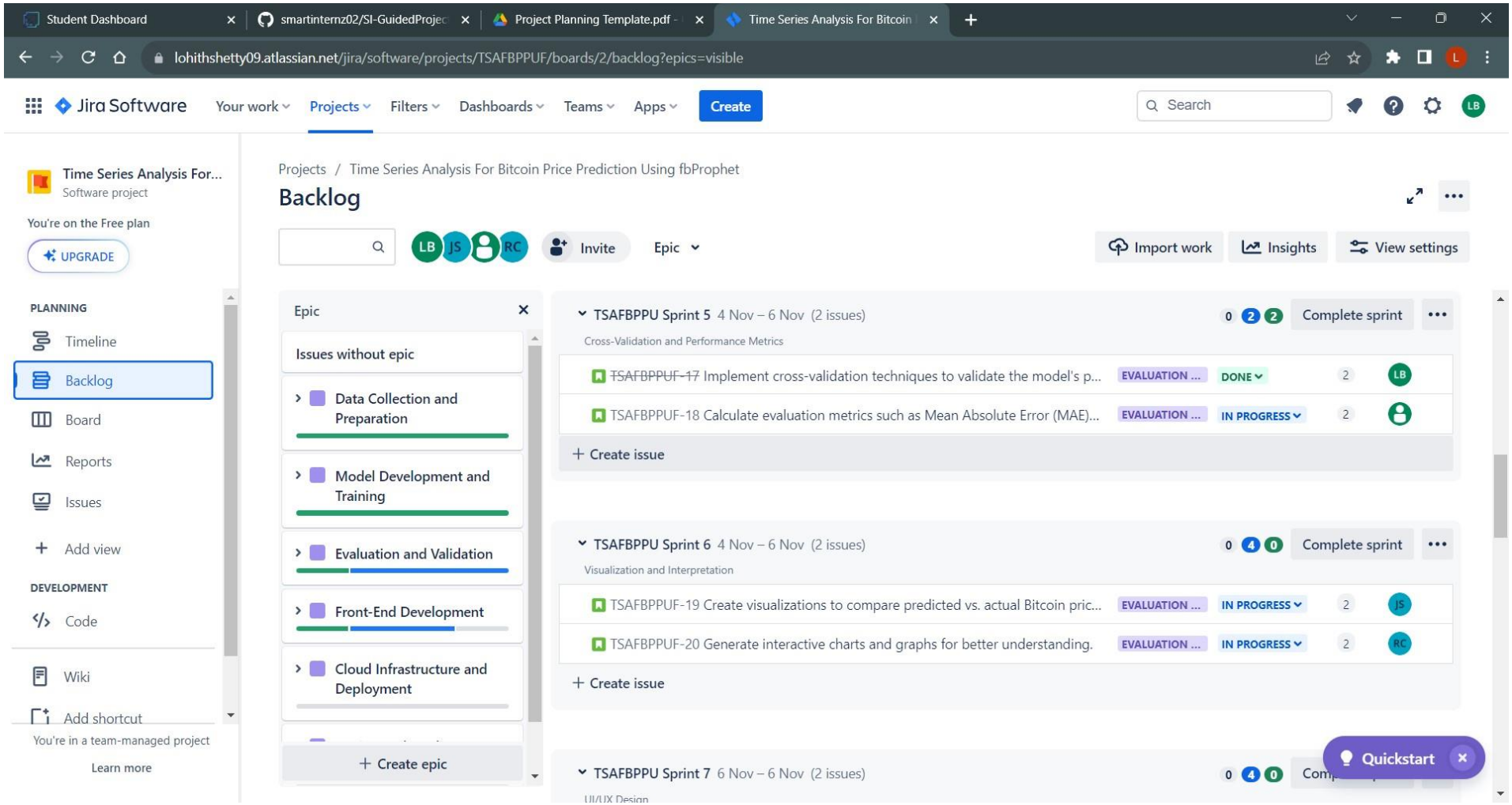
Board:

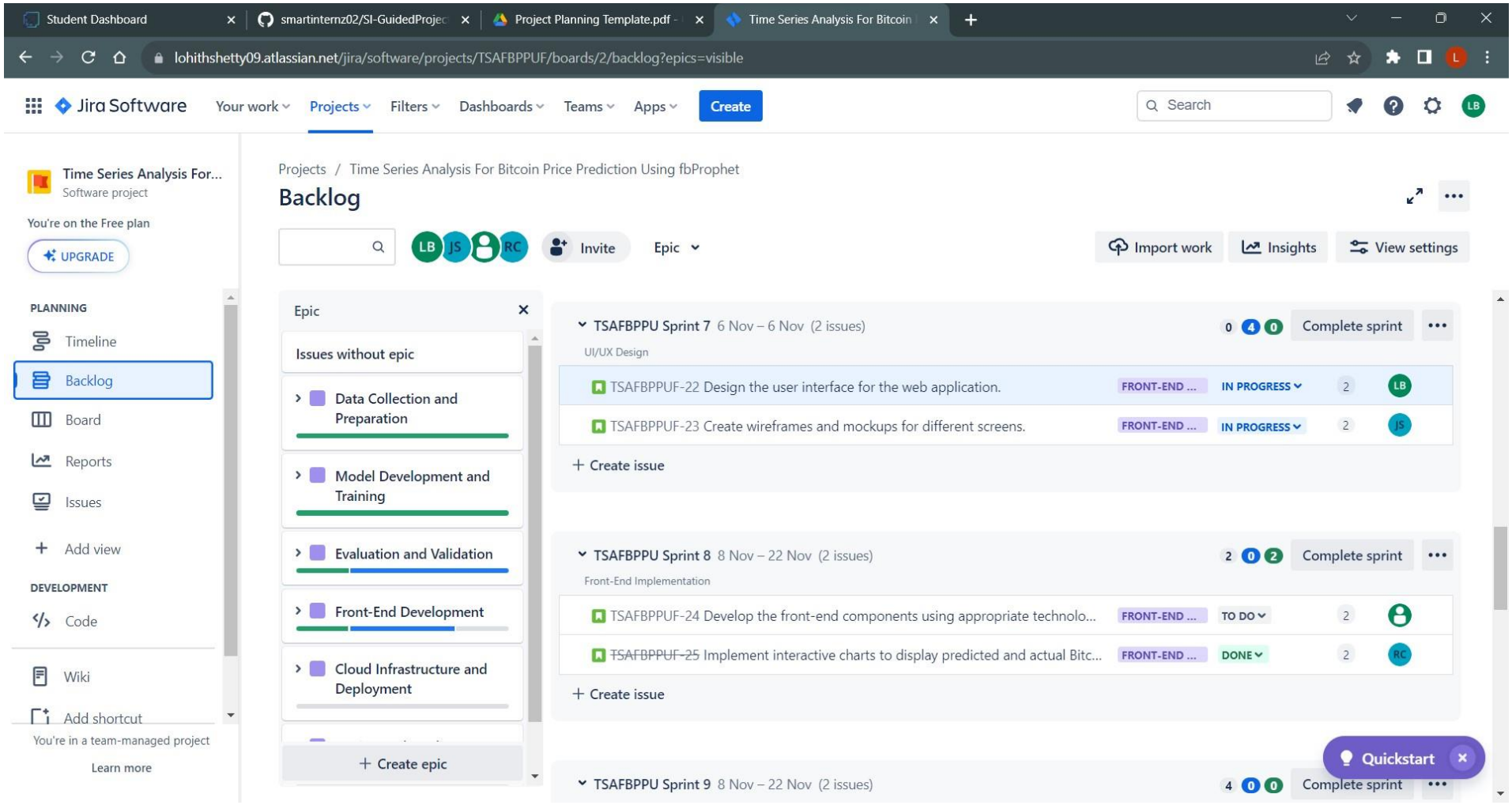

Student Dashboard
smartinternz02/SI-GuidedProjec
Project Planning Template.pdf -
TSAFBPPUF board - Agile board
lohithshetty09.atlassian.net/jira/software/projects/TSAFBPPUF/boards/2
Jira Software
Your work
Projects
Filters
Dashboards
Teams
Apps
Create
Search
Time Series Analysis For...
Software project
You're on the Free plan
UPGRADE
PLANNING
Timeline
Backlog
Board
Reports
Issues
Add view
DEVELOPMENT
Code
Wiki
Add shortcut
You're in a team-managed project
Learn more
Projects / Time Series Analysis For Bitcoin Price Prediction Using fbProphet
All sprints
Complete sprint
Invite
Epic
Sprint
GROUP BY
None
Import work
Insights
View settings
TO DO 3
IN PROGRESS 5
DONE 10
Develop the front-end components using appropriate technologies (HTML, CSS, JavaScript frameworks like React or Angular).
FRONT-END DEVELOPMENT
TSAFBPPUF-24
Choose a cloud service provider (such as AWS, Azure, or Google Cloud) for hosting the application and model.
CLOUD INFRASTRUCTURE AND DEPLOYM...
TSAFBPPUF-27
Set up cloud infrastructure, including servers, databases, and storage solutions.
CLOUD INFRASTRUCTURE AND DEPLOYM...
Calculate evaluation metrics such as Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE).
EVALUATION AND VALIDATION
TSAFBPPUF-18
Create visualizations to compare predicted vs. actual Bitcoin prices.
EVALUATION AND VALIDATION
TSAFBPPUF-19
Generate interactive charts and graphs for better understanding.
EVALUATION AND VALIDATION
TSAFBPPUF-20
Implement a basic Prophet model for Bitcoin price prediction.
MODEL DEVELOPMENT AND TRAINING
TSAFBPPUF-12
Train the model using the prepared dataset.
MODEL DEVELOPMENT AND TRAINING
TSAFBPPUF-13
Fine-tune the Prophet model parameters to improve accuracy.
MODEL DEVELOPMENT AND TRAINING
TSAFBPPUF-14
Experiment with different
Quickstart

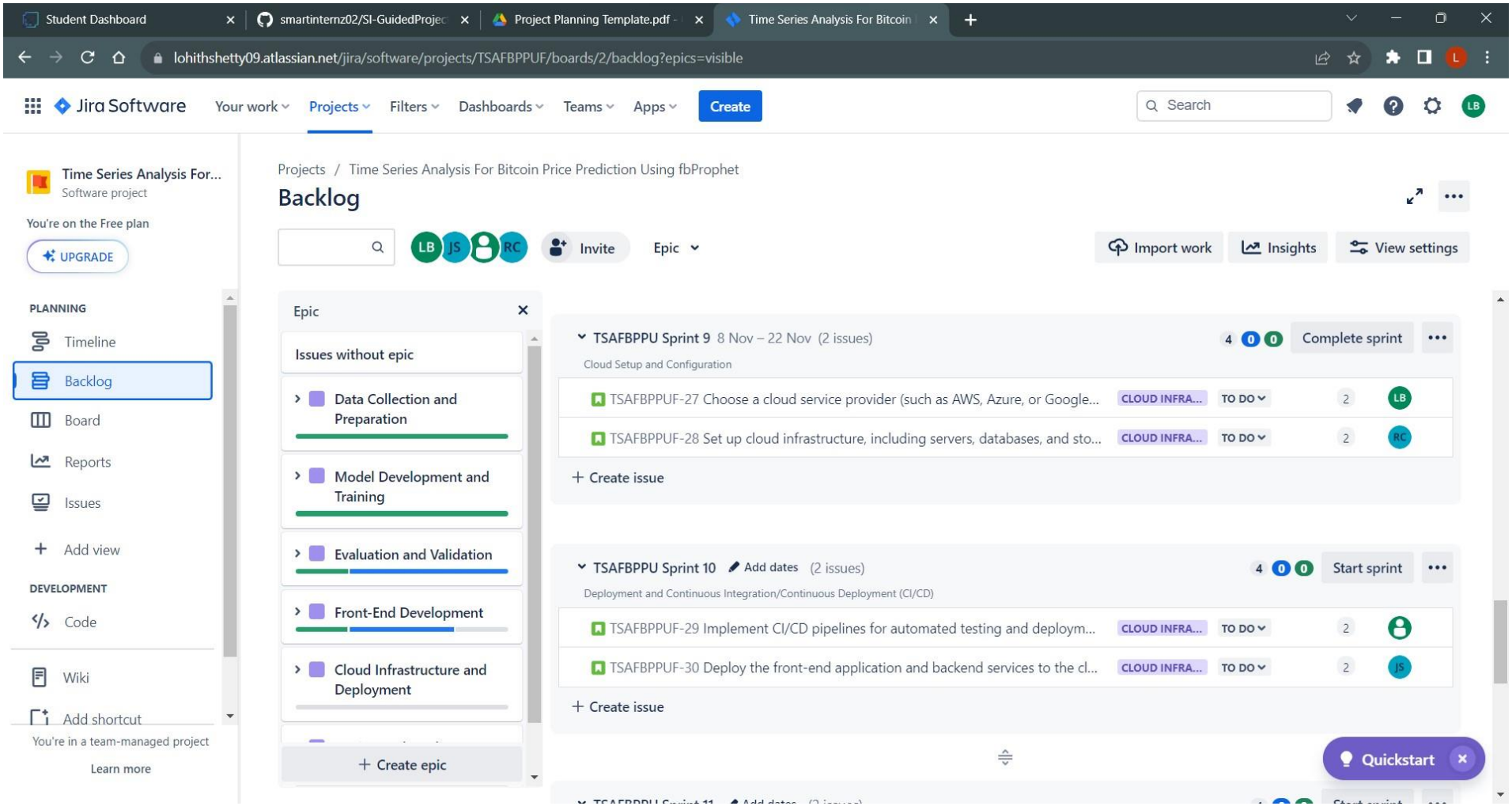
Backlog:

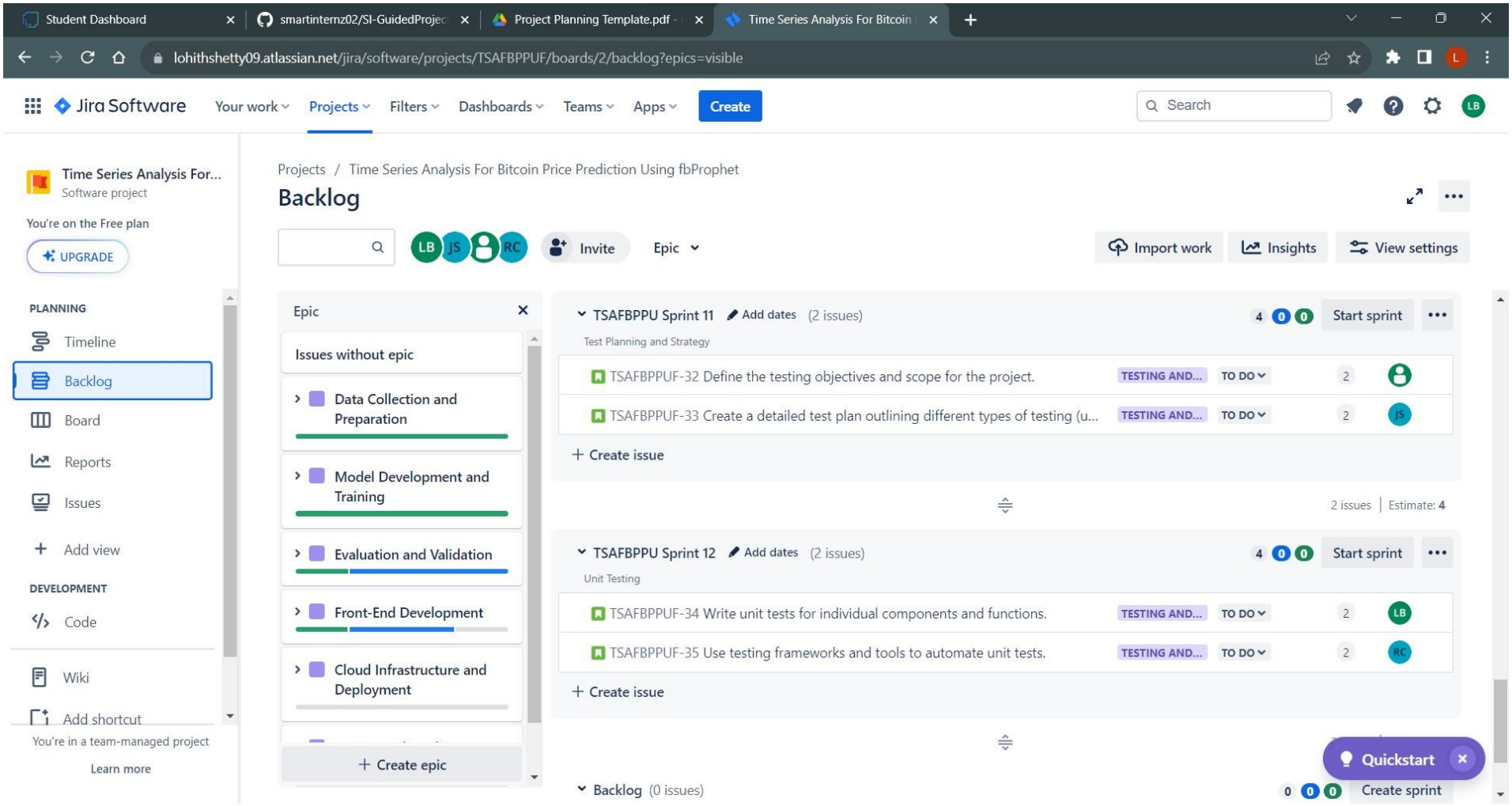

Student Dashboard
smartinternz02/SI-GuidedProjec
Project Planning Template.pdf -
Time Series Analysis For Bitcoin
lohithshetty09.atlassian.net/jira/software/projects/TSAFBPPUF/boards/2/backlog?epics=visible
Jira Software
Your work
Projects
Filters
Dashboards
Teams
Apps
Create
Search
Time Series Analysis For...
Software project
You're on the Free plan
UPGRADE
PLANNING
Timeline
Backlog
Board
Reports
Issues
Add view
DEVELOPMENT
Code
Wiki
Add shortcut
You're in a team-managed project
Learn more
Projects / Time Series Analysis For Bitcoin Price Prediction Using fbProphet
Backlog
Invite
Epic
Import work
Insights
View settings
Epic
Issues without epic
Data Collection and Preparation
Model Development and Training
Evaluation and Validation
Front-End Development
Cloud Infrastructure and Deployment
Create epic
TSAFBPPU Sprint 2 20 Oct – 22 Oct (2 issues)
Data Exploration and Feature Engineering
Complete sprint
TSAFBPPUF-10 Explore the cleaned data to understand patterns and trends.
TSAFBPPUF-11 Engineer relevant features such as moving averages, relativ...
DATA COLLEC...
DONE
Create issue
TSAFBPPU Sprint 1 20 Oct – 22 Oct (2 issues)
Data Gathering and Cleaning
TSAFBPPUF-38 Set up data collection scripts or tools to fetch the data.
TSAFBPPUF-37 Research and select reliable sources for historical Bitcoin pr...
TSAFBPPU Sprint 3 24 Oct – 28 Oct (2 issues)
Baseline Prophet Model
Quickstart


Student Dashboard
smartinternz02/SI-GuidedProjec
Project Planning Template.pdf -
Time Series Analysis For Bitcoin
lohithshetty09.atlassian.net/jira/software/projects/TSAFBPPUF/boards/2/backlog?epics=visible
Jira Software
Your work
Projects
Filters
Dashboards
Teams
Apps
Create
Search
Time Series Analysis For...
Software project
You're on the Free plan
UPGRADE
PLANNING
Timeline
Backlog
Board
Reports
Issues
Add view
DEVELOPMENT
Code
Wiki
Add shortcut
You're in a team-managed project
Learn more
Projects / Time Series Analysis For Bitcoin Price Prediction Using fbProphet
Backlog
Invite
Epic
Import work
Insights
View settings
Epic
Issues without epic
Data Collection and Preparation
Model Development and Training
Evaluation and Validation
Front-End Development
Cloud Infrastructure and Deployment
Create epic
TSAFBPPU Sprint 3 24 Oct – 28 Oct (2 issues)
Baseline Prophet Model
TSAFBPPUF-12 Implement a basic Prophet model for Bitcoin price predicti...
TSAFBPPUF-13 Train the model using the prepared dataset.
MODEL DEVE...
DONE
Create issue
TSAFBPPU Sprint 4 28 Oct – 4 Nov (2 issues)
Model Optimization and Tuning
TSAFBPPUF-14 Fine-tune the Prophet model parameters to improve accur...
TSAFBPPUF-15 Experiment with different seasonality settings and changep...
Create issue
Complete sprint
Quickstart
TSAFBPPU Sprint 5 4 Nov – 6 Nov (2 issues)

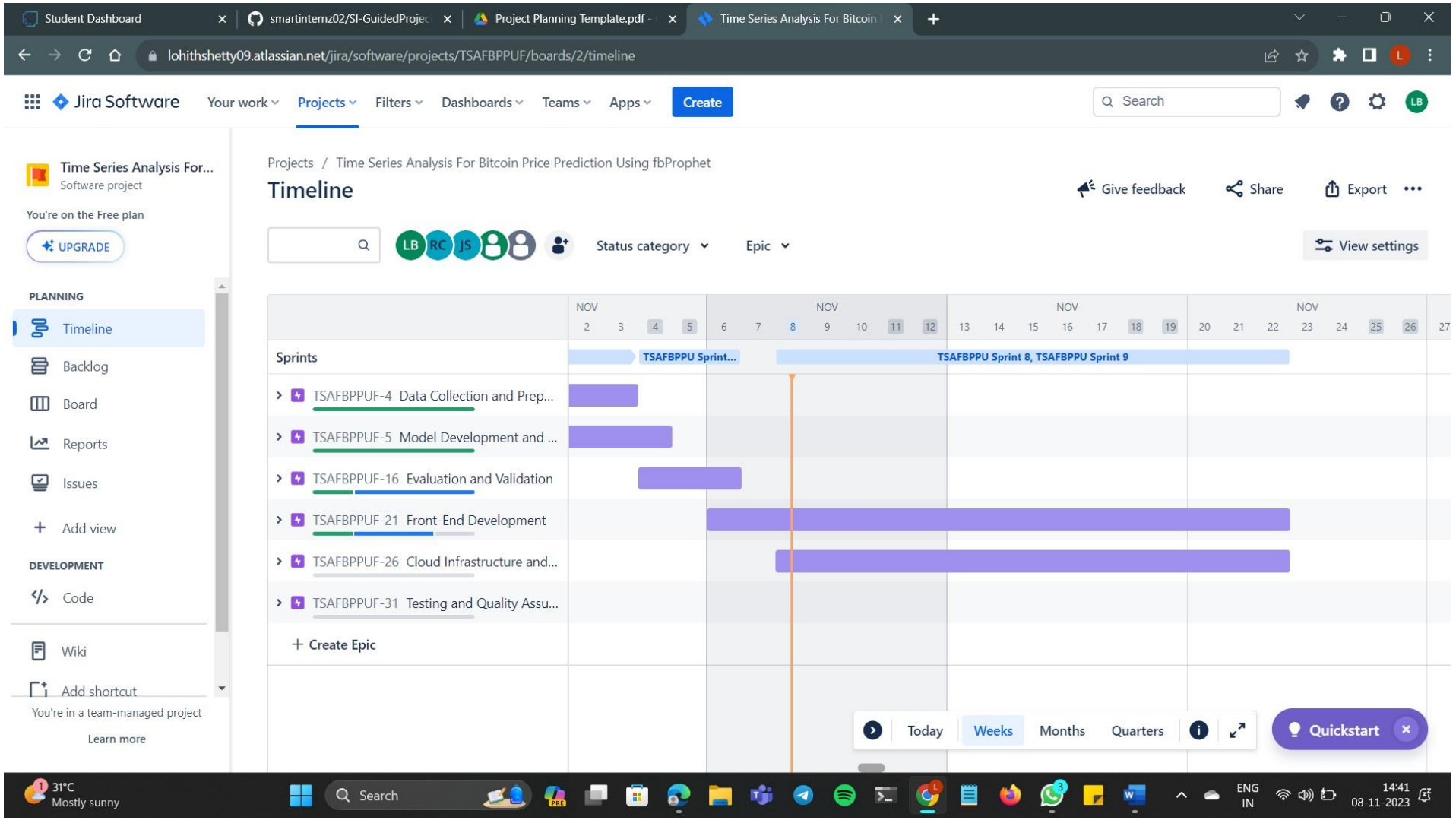

Student Dashboard
smartinternz02/SI-GuidedProjec
Project Planning Template.pdf -
Time Series Analysis For Bitcoin
lohithshetty09.atlassian.net/jira/software/projects/TSAFBPPUF/boards/2/backlog?epics=visible
Jira Software
Your work
Projects
Filters
Dashboards
Teams
Apps
Create
Search
Time Series Analysis For...
Software project
You're on the Free plan
UPGRADE
PLANNING
Timeline
Backlog
Board
Reports
Issues
Add view
DEVELOPMENT
Code
Wiki
Add shortcut
You're in a team-managed project
Learn more
Projects / Time Series Analysis For Bitcoin Price Prediction Using fbProphet
Backlog
Invite
Epic
Import work
Insights
View settings
Epic
Issues without epic
Data Collection and Preparation
Model Development and Training
Evaluation and Validation
Front-End Development
Cloud Infrastructure and Deployment
Create epic
TSAFBPPU Sprint 5 4 Nov – 6 Nov (2 issues)
Complete sprint
Cross-Validation and Performance Metrics
TSAFBPPUF-17 Implement cross-validation techniques to validate the model's p...
EVALUATION ...
DONE
TSAFBPPUF-18 Calculate evaluation metrics such as Mean Absolute Error (MAE)...
IN PROGRESS
Create issue
TSAFBPPU Sprint 6 4 Nov – 6 Nov (2 issues)
Visualization and Interpretation
TSAFBPPUF-19 Create visualizations to compare predicted vs. actual Bitcoin pric...
TSAFBPPUF-20 Generate interactive charts and graphs for better understanding.
TSAFBPPU Sprint 7 6 Nov – 6 Nov (2 issues)
Quickstart


Student Dashboard
smartinternz02/SI-GuidedProjec
Project Planning Template.pdf -
Time Series Analysis For Bitcoin
lohithshetty09.atlassian.net/jira/software/projects/TSAFBPPUF/boards/2/backlog?epics=visible
Jira Software
Your work
Projects
Filters
Dashboards
Teams
Apps
Create
Search
Time Series Analysis For...
Software project
You're on the Free plan
UPGRADE
PLANNING
Timeline
Backlog
Board
Reports
Issues
Add view
DEVELOPMENT
Code
Wiki
Add shortcut
You're in a team-managed project
Learn more
Projects / Time Series Analysis For Bitcoin Price Prediction Using fbProphet
Backlog
Invite
Epic
Import work
Insights
View settings
Epic
Issues without epic
Data Collection and Preparation
Model Development and Training
Evaluation and Validation
Front-End Development
Cloud Infrastructure and Deployment
Create epic
TSAFBPPU Sprint 7 6 Nov – 6 Nov (2 issues)
0 4 0
Complete sprint
UI/UX Design
TSAFBPPUF-22 Design the user interface for the web application.
FRONT-END ...
IN PROGRESS
2
TSAFBPPUF-23 Create wireframes and mockups for different screens.
FRONT-END ...
IN PROGRESS
2
Create issue
TSAFBPPU Sprint 8 8 Nov – 22 Nov (2 issues)
2 0 2
Complete sprint
Front-End Implementation
TSAFBPPUF-24 Develop the front-end components using appropriate technolo...
FRONT-END ...
TO DO
2
~~TSAFBPPUF-25~~ Implement interactive charts to display predicted and actual Bitc...
FRONT-END ...
DONE
2
Create issue
Quickstart
TSAFBPPU Sprint 9 8 Nov – 22 Nov (2 issues)
4 0 0
Complete sprint

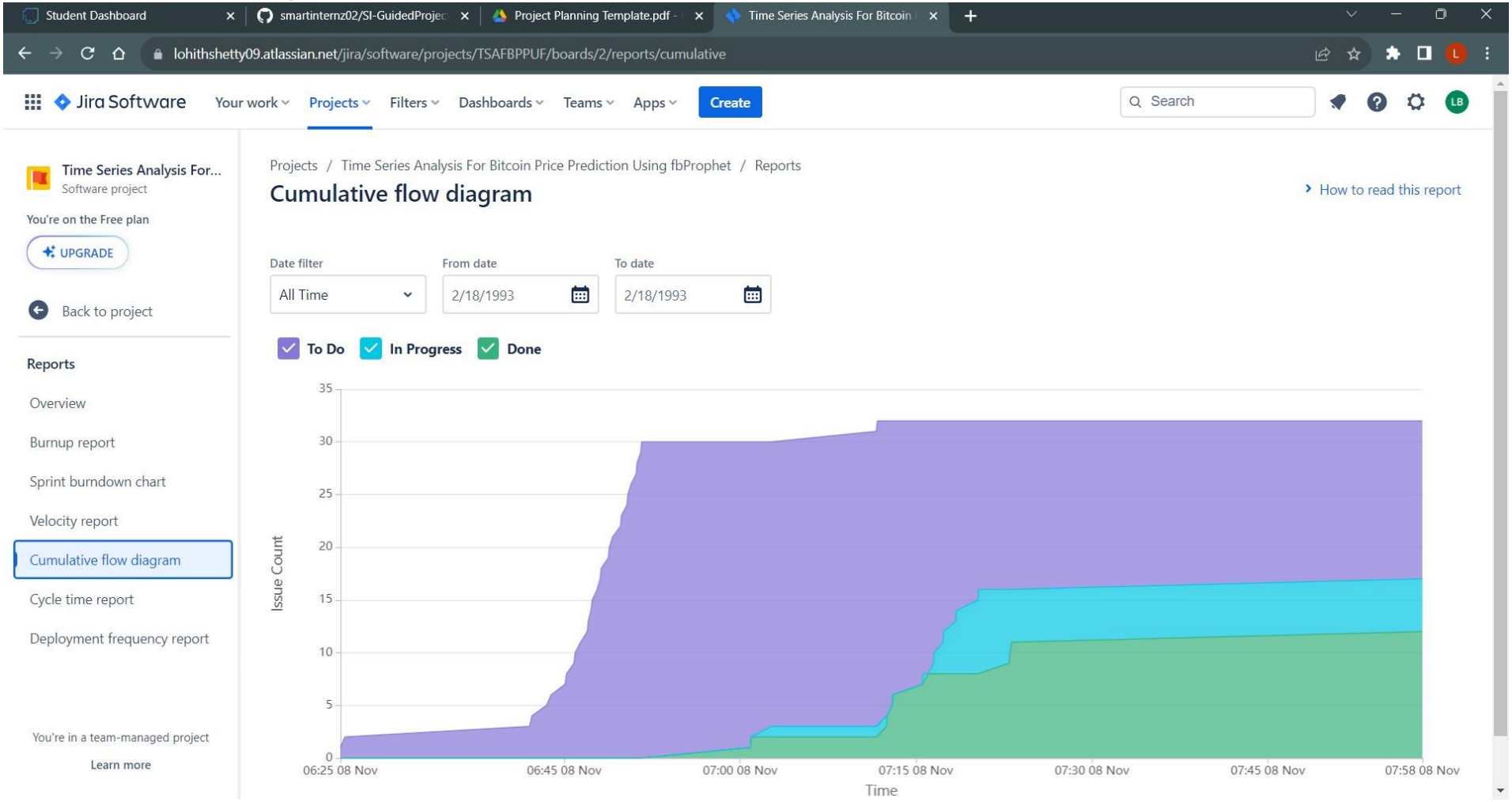

Student Dashboard
smartinternz02/SI-GuidedProjec
Project Planning Template.pdf -
Time Series Analysis For Bitcoin
lohithshetty09.atlassian.net/jira/software/projects/TSAFBPPUF/boards/2/backlog?epics=visible
Jira Software
Your work
Projects
Filters
Dashboards
Teams
Apps
Create
Search
Time Series Analysis For...
Software project
You're on the Free plan
UPGRADE
PLANNING
Timeline
Backlog
Board
Reports
Issues
Add view
DEVELOPMENT
Code
Wiki
Add shortcut
You're in a team-managed project
Learn more
Projects / Time Series Analysis For Bitcoin Price Prediction Using fbProphet
Backlog
Invite
Epic
Import work
Insights
View settings
Epic
Issues without epic
Data Collection and Preparation
Model Development and Training
Evaluation and Validation
Front-End Development
Cloud Infrastructure and Deployment
Create epic
TSAFBPPU Sprint 9 8 Nov – 22 Nov (2 issues)
Complete sprint
Cloud Setup and Configuration
TSAFBPPUF-27 Choose a cloud service provider (such as AWS, Azure, or Google...
CLOUD INFRA...
TO DO
TSAFBPPUF-28 Set up cloud infrastructure, including servers, databases, and sto...
Create issue
TSAFBPPU Sprint 10
Add dates
(2 issues)
Start sprint
Deployment and Continuous Integration/Continuous Deployment (CI/CD)
TSAFBPPUF-29 Implement CI/CD pipelines for automated testing and deploym...
TSAFBPPUF-30 Deploy the front-end application and backend services to the cl...
Create issue
Quickstart


Student Dashboard
smartinternz02/SI-GuidedProjec
Project Planning Template.pdf -
Time Series Analysis For Bitcoin
lohithshetty09.atlassian.net/jira/software/projects/TSAFBPPUF/boards/2/backlog?epics=visible
Jira Software
Your work
Projects
Filters
Dashboards
Teams
Apps
Create
Search
Time Series Analysis For...
Software project
You're on the Free plan
UPGRADE
PLANNING
Timeline
Backlog
Board
Reports
Issues
Add view
DEVELOPMENT
Code
Wiki
Add shortcut
You're in a team-managed project
Learn more
Projects / Time Series Analysis For Bitcoin Price Prediction Using fbProphet
Backlog
Invite
Epic
Import work
Insights
View settings
Epic
Issues without epic
Data Collection and Preparation
Model Development and Training
Evaluation and Validation
Front-End Development
Cloud Infrastructure and Deployment
Create epic
TSAFBPPU Sprint 11
Add dates
(2 issues)
Start sprint
Test Planning and Strategy
TSAFBPPUF-32 Define the testing objectives and scope for the project.
TSAFBPPUF-33 Create a detailed test plan outlining different types of testing (u...
TESTING AND...
TO DO
Create issue
2 issues | Estimate: 4
TSAFBPPU Sprint 12
Add dates
(2 issues)
Start sprint
Unit Testing
TSAFBPPUF-34 Write unit tests for individual components and functions.
TSAFBPPUF-35 Use testing frameworks and tools to automate unit tests.
Create issue
Quickstart
Backlog (0 issues)
Create sprint

Timeline:


Student Dashboard
smartinternz02/SI-GuidedProjec
Project Planning Template.pdf -
Time Series Analysis For Bitcoin
lohithshetty09.atlassian.net/jira/software/projects/TSAFBPPUF/boards/2/timeline
Jira Software
Your work
Projects
Filters
Dashboards
Teams
Apps
Create
Search
Time Series Analysis For...
Software project
You're on the Free plan
UPGRADE
PLANNING
Timeline
Backlog
Board
Reports
Issues
Add view
DEVELOPMENT
Code
Wiki
Add shortcut
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
NOV
Sprints
TSAFBPPU Sprint...
TSAFBPPU Sprint 8, TSAFBPPU Sprint 9
TSAFBPPUF-4 Data Collection and Prep...
TSAFBPPUF-5 Model Development and ...
TSAFBPPUF-16 Evaluation and Validation
TSAFBPPUF-21 Front-End Development
TSAFBPPUF-26 Cloud Infrastructure and...
TSAFBPPUF-31 Testing and Quality Assu...
Create Epic
Today
Weeks
Months
Quarters
Quickstart
31°C
Mostly sunny
Search
ENG
IN
14:41
08-11-2023

Cumulative flow Diagram



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

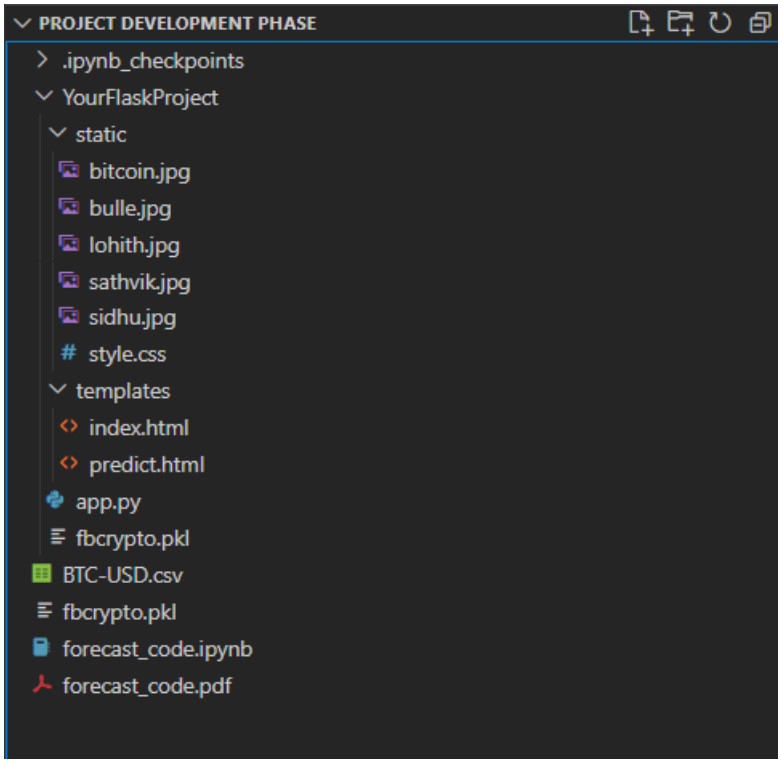
fbprophet for Time Series Analysis

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: Data 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.

```
In [1]: 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 fbprophet import Prophet
        from fbprophet.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.

```
In [2]: today=datetime.today().strftime('%Y-%m-%d')
        start_date = '2016-01-01'
        df = yf.download('BTC-USD', start_date, today)
```

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

```
In [3]: df
```

Out[3]:

Date	Open	High	Low	Close	Adj Close	Volume
2016-01-01	$430.72	$436.25	$427.52	$434.33	$434.33	36278900
2016-01-02	$434.62	$436.06	$431.87	$433.44	$433.44	30096600
2016-01-03	$433.58	$433.74	$424.71	$430.01	$430.01	39633800
2016-01-04	$430.06	$434.52	$429.08	$433.09	$433.09	38477500
2016-01-05	$433.07	$434.18	$429.68	$431.96	$431.96	34522600
...	...	...	...	...	...	...
2023-11-05	$35,090.01	$35,340.34	$34,594.24	$35,049.36	$35,049.36	12412743996
2023-11-06	$35,044.79	$35,286.03	$34,765.36	$35,037.37	$35,037.37	12693436420
2023-11-07	$35,047.79	$35,892.42	$34,545.82	$35,443.56	$35,443.56	18834737789
2023-11-08	$35,419.48	$35,994.42	$35,147.80	$35,655.28	$35,655.28	17295394918
2023-11-09	$35,633.63	$37,926.26	$35,592.10	$36,693.12	$36,693.12	37762672382

2870 rows × 6 columns

- Bitcoin Dataset contains the following Columns

```
In [4]: df.columns
```

```
Out[4]: Index(['Open', 'High', 'Low', 'Close', 'Adj Close', 'Volume'], dtype='object')
```

```
In [5]: df.head()
```

Out[5]:

Date	Open	High	Low	Close	Adj Close	Volume
2016-01-01	$430.72	$436.25	$427.52	$434.33	$434.33	36278900
2016-01-02	$434.62	$436.06	$431.87	$433.44	$433.44	30096600
2016-01-03	$433.58	$433.74	$424.71	$430.01	$430.01	39633800
2016-01-04	$430.06	$434.52	$429.08	$433.09	$433.09	38477500
2016-01-05	$433.07	$434.18	$429.68	$431.96	$431.96	34522600

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

```
In [6]: df.describe()
```

Out[6]:

	Open	High	Low	Close	Adj Close	Volume
count	$2,870.00	$2,870.00	$2,870.00	$2,870.00	$2,870.00	$2,870.00
mean	$16,450.12	$16,839.41	$16,029.14	$16,461.19	$16,461.19	$19,168,087,374.89
std	$16,145.30	$16,541.47	$15,696.21	$16,143.61	$16,143.61	$19,434,299,526.55
min	$365.07	$374.95	$354.91	$364.33	$364.33	$28,514,000.00
25%	$4,052.31	$4,115.34	$3,972.01	$4,066.18	$4,066.18	$3,674,422,500.00
50%	$9,518.59	$9,690.55	$9,305.47	$9,521.06	$9,521.06	$15,653,982,072.00
75%	$26,628.61	$27,053.17	$26,322.63	$26,705.61	$26,705.61	$29,906,093,638.50
max	$67,549.73	$68,789.62	$66,382.06	$67,566.83	$67,566.83	$350,967,941,479.00

```
In [7]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 2870 entries, 2016-01-01 to 2023-11-09
Data columns (total 6 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   Open       2870 non-null   float64
 1   High       2870 non-null   float64
 2   Low        2870 non-null   float64
 3   Close      2870 non-null   float64
 4   Adj Close  2870 non-null   float64
 5   Volume     2870 non-null   int64
dtypes: float64(5), int64(1)
memory usage: 157.0 KB
```

Check null values

```
In [8]: df.isnull().any()
```

```
Out[8]: Open         False
        High         False
        Low          False
        Close        False
        Adj Close    False
        Volume       False
        dtype: bool
```

```
In [9]: df.isnull().sum()
```

```
Out[9]: 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.

```
In [10]: df.reset_index(inplace=True)
         df.columns
```

```
Out[10]: Index(['Date', 'Open', 'High', 'Low', 'Close', 'Adj Close', 'Volume'], dtype='object')
```

```
In [11]: df.head()
```

Out[11]:

	Date	Open	High	Low	Close	Adj Close	Volume
0	2016-01-01	$430.72	$436.25	$427.52	$434.33	$434.33	36278900
1	2016-01-02	$434.62	$436.06	$431.87	$433.44	$433.44	30096600
2	2016-01-03	$433.58	$433.74	$424.71	$430.01	$430.01	39633800
3	2016-01-04	$430.06	$434.52	$429.08	$433.09	$433.09	38477500
4	2016-01-05	$433.07	$434.18	$429.68	$431.96	$431.96	34522600

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.

```
In [12]: df1=df[['Date','Open']]
         df1.head()
```

Out[12]:

	Date	Open
0	2016-01-01	$430.72
1	2016-01-02	$434.62
2	2016-01-03	$433.58
3	2016-01-04	$430.06
4	2016-01-05	$433.07

Rename to reuse easily

In [13]:
```
newn={
    "Date":"ds",
    "Open":"y",
}
df1.rename(columns=newn,inplace=True)
df1.head()
```

Out[13]:

	ds	y
0	2016-01-01	$430.72
1	2016-01-02	$434.62
2	2016-01-03	$433.58
3	2016-01-04	$430.06
4	2016-01-05	$433.07

Activity 5: Visualize Time Series Plot

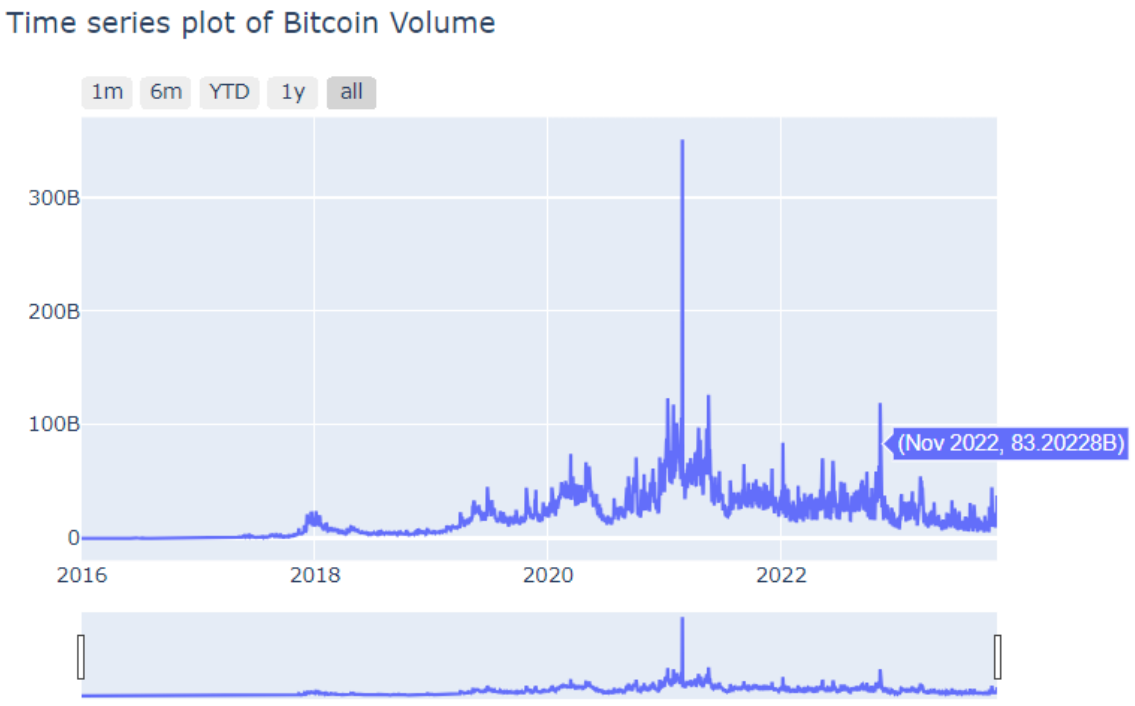
Date vs Open price:

In [14]:
```
import plotly.graph_objects as go

x = df1["ds"]
y = df1["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:

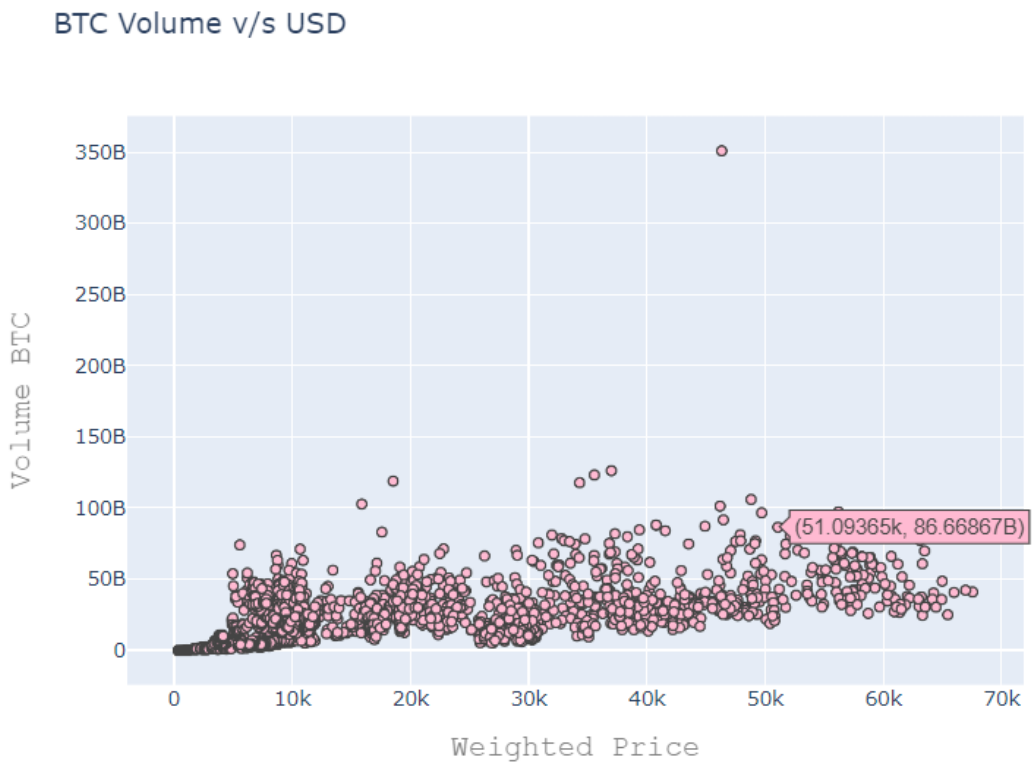
In [15]:

```
import plotly.graph_objects as go

x = df["Date"]
y = df["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:

In [16]:
```
import plotly.offline as pyo
pyo.init_notebook_mode(connected=True)


#BTC Volume vs USD visualization
trace = go.Scattergl(
    y = df['Volume'].astype(float),
    x = df['Close'].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.

In [17]:
```
m = Prophet(
    seasonality_mode="multiplicative",
)

m.fit(df1)
```

```
INFO:fbprophet:Disabling daily seasonality. Run prophet with daily_seasonality=True to override thi
s.
```

Out[17]:
```
<fbprophet.forecaster.Prophet at 0x1ee72fccc10>
```

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.

```
In [18]: future=m.make_future_dataframe(periods=365)
         future.tail()
```

Out[18]:

	ds
3230	2024-11-04
3231	2024-11-05
3232	2024-11-06
3233	2024-11-07
3234	2024-11-08

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.

In [19]:
```
forecast=m.predict(future)
forecast
```

Out[19]:

	ds	trend	yhat_lower	yhat_upper	trend_lower	trend_upper	multiplicative_terms	multiplicati
0	2016-01-01	$6.26	$-4,735.94	$4,711.12	$6.26	$6.26	$-0.00	
1	2016-01-02	$8.99	$-4,306.90	$4,650.01	$8.99	$8.99	$0.00	
2	2016-01-03	$11.72	$-4,762.28	$4,826.99	$11.72	$11.72	$0.01	
3	2016-01-04	$14.45	$-4,805.23	$4,474.76	$14.45	$14.45	$0.01	
4	2016-01-05	$17.18	$-5,040.90	$4,587.34	$17.18	$17.18	$0.02	
...	...	...	...	...	...	...	...	
3230	2024-11-04	$24,437.69	$6,069.07	$45,477.33	$6,159.27	$41,534.18	$0.09	
3231	2024-11-05	$24,435.23	$5,701.90	$44,746.41	$6,099.99	$41,612.01	$0.08	
3232	2024-11-06	$24,432.77	$6,677.38	$45,993.25	$6,012.79	$41,689.83	$0.08	
3233	2024-11-07	$24,430.31	$5,667.70	$46,753.96	$5,925.60	$41,754.17	$0.08	
3234	2024-11-08	$24,427.85	$6,166.71	$45,558.05	$5,838.41	$41,800.49	$0.07	

3235 rows × 19 columns

In [20]:
```
forecast[['ds','yhat','yhat_lower','yhat_upper']].tail()
```

Out[20]:

	ds	yhat	yhat_lower	yhat_upper
3230	2024-11-04	$26,543.50	$6,069.07	$45,477.33
3231	2024-11-05	$26,478.18	$5,701.90	$44,746.41
3232	2024-11-06	$26,391.42	$6,677.38	$45,993.25
3233	2024-11-07	$26,361.85	$5,667.70	$46,753.96
3234	2024-11-08	$26,149.90	$6,166.71	$45,558.05

In [21]:
```
forecast[['ds','yhat','yhat_lower','yhat_upper']].head()
```

Out[21]:

	ds	yhat	yhat_lower	yhat_upper
0	2016-01-01	$6.24	$-4,735.94	$4,711.12
1	2016-01-02	$9.01	$-4,306.90	$4,650.01
2	2016-01-03	$11.81	$-4,762.28	$4,826.99
3	2016-01-04	$14.64	$-4,805.23	$4,474.76
4	2016-01-05	$17.46	$-5,040.90	$4,587.34

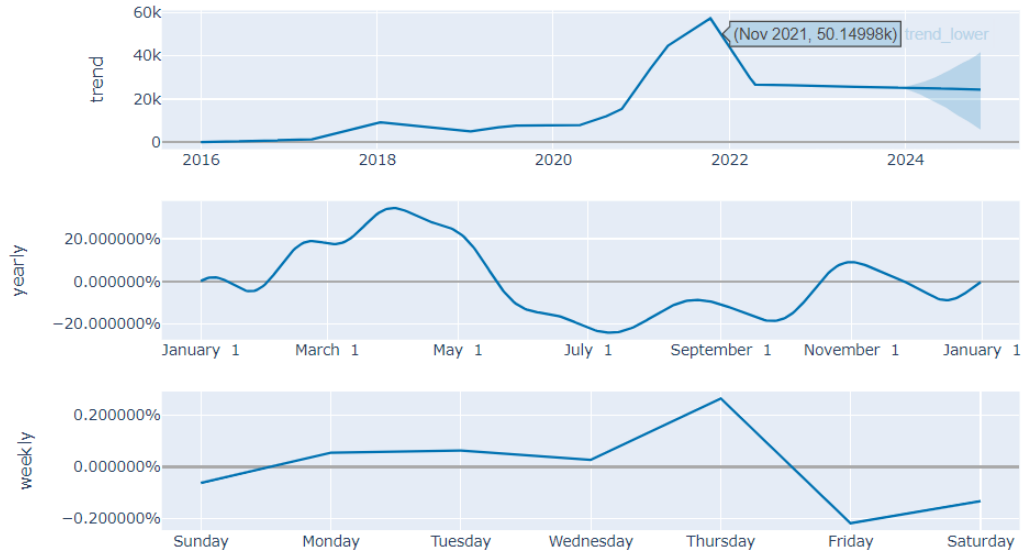
- To forecast the price for the next day, we calculate the DateTime and store it in the `next_day` variable. Subsequently, we use this variable to predict the corresponding value.

```
In [22]: next_day=(datetime.today()+timedelta(days=1)).strftime('%Y-%m-%d')
         forecast[forecast['ds']==next_day]['yhat'].item()

Out[22]: 26894.970744877926
```

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

```
In [23]: plot_plotly(m,forecast)
```

```
In [24]: plot_components_plotly(m,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.

```
In [25]: import pickle
         pickle.dump(m,open('fbcrypto.pkl','wb'))
```

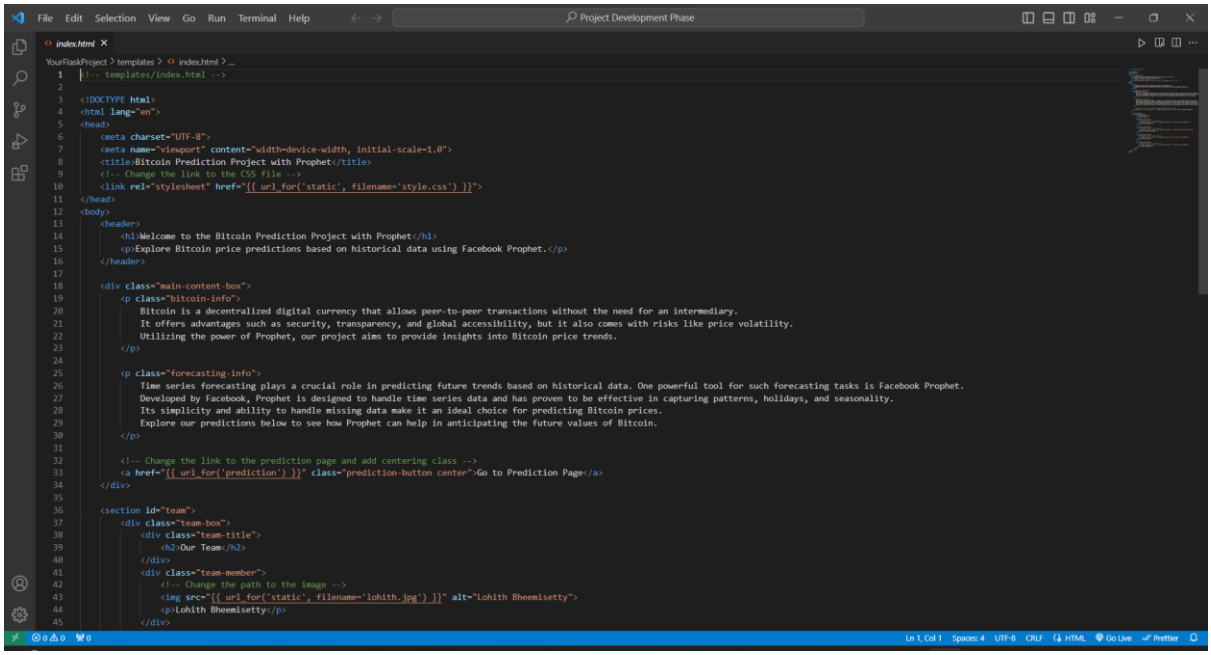
Milestone 5: Application Building

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.

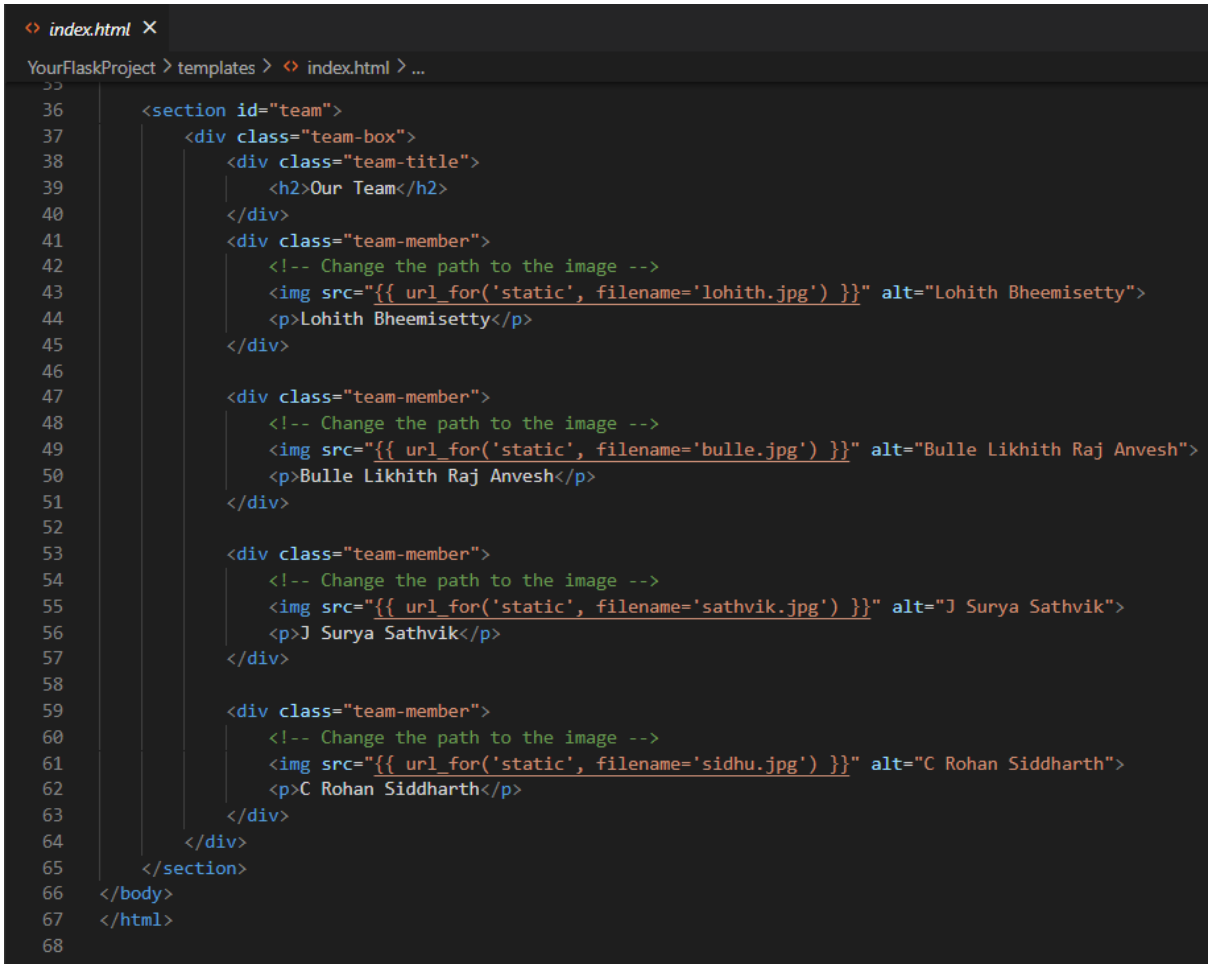
<!-- Change the link to the prediction page and add centering class -->
Go to Prediction Page

<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

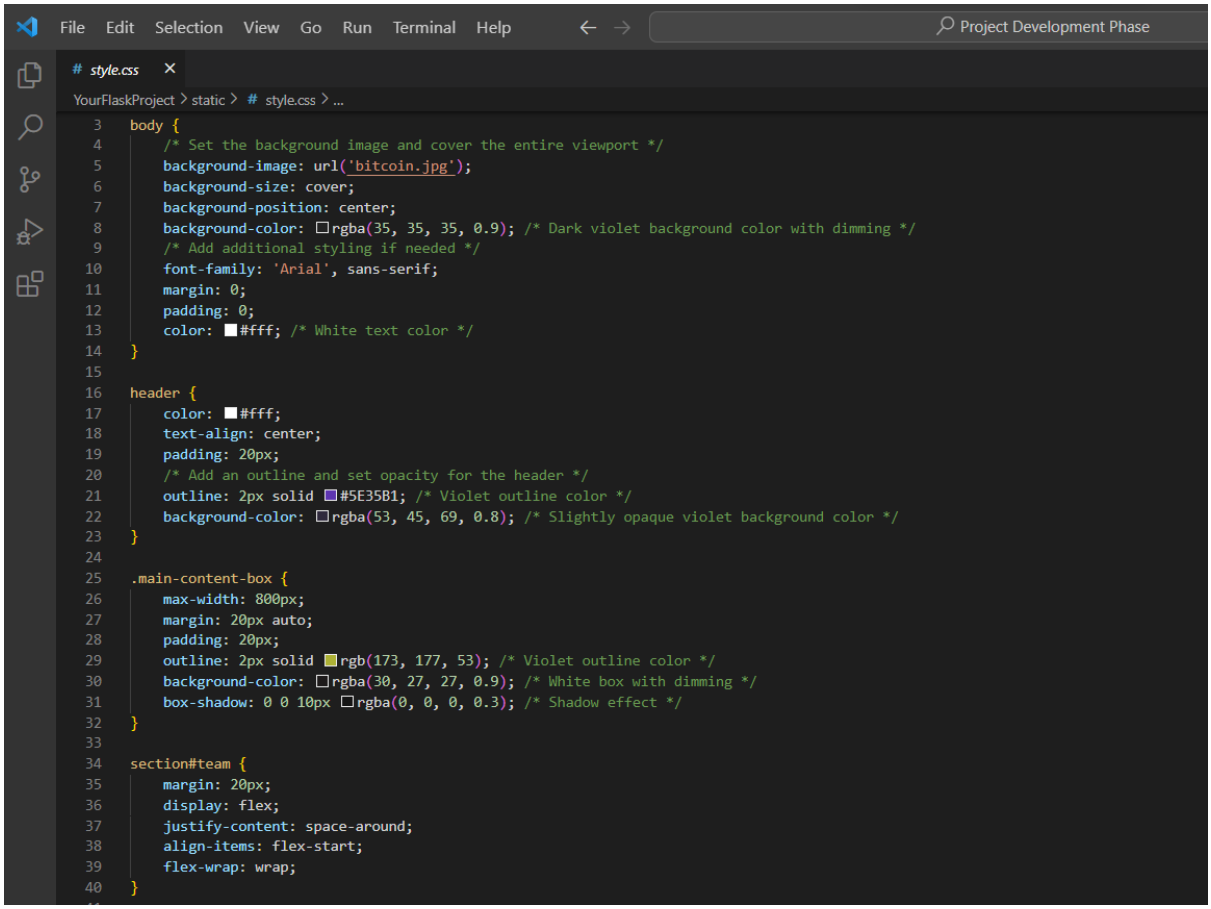
<!-- Change the path to the image -->
<img src="{{ url_for('static', filename='bulle.jpg') }}" alt="Bulle Likhith Raj Anvesh">
Bulle Likhith Raj Anvesh

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

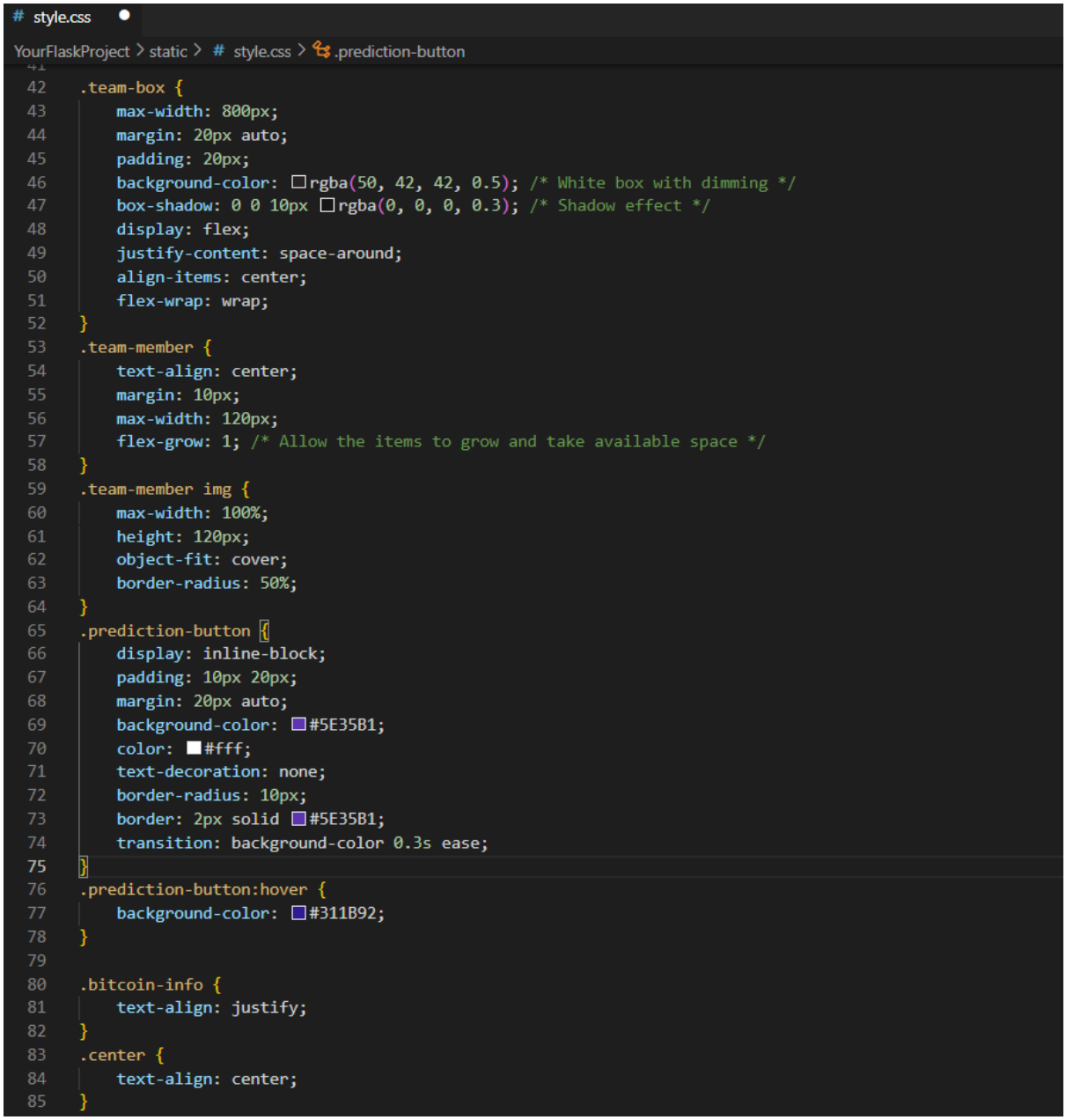
<!-- Change the path to the image -->
<img src="{{ url_for('static', filename='sidhu.jpg') }}" alt="C Rohan Siddharth">
C Rohan Siddharth

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

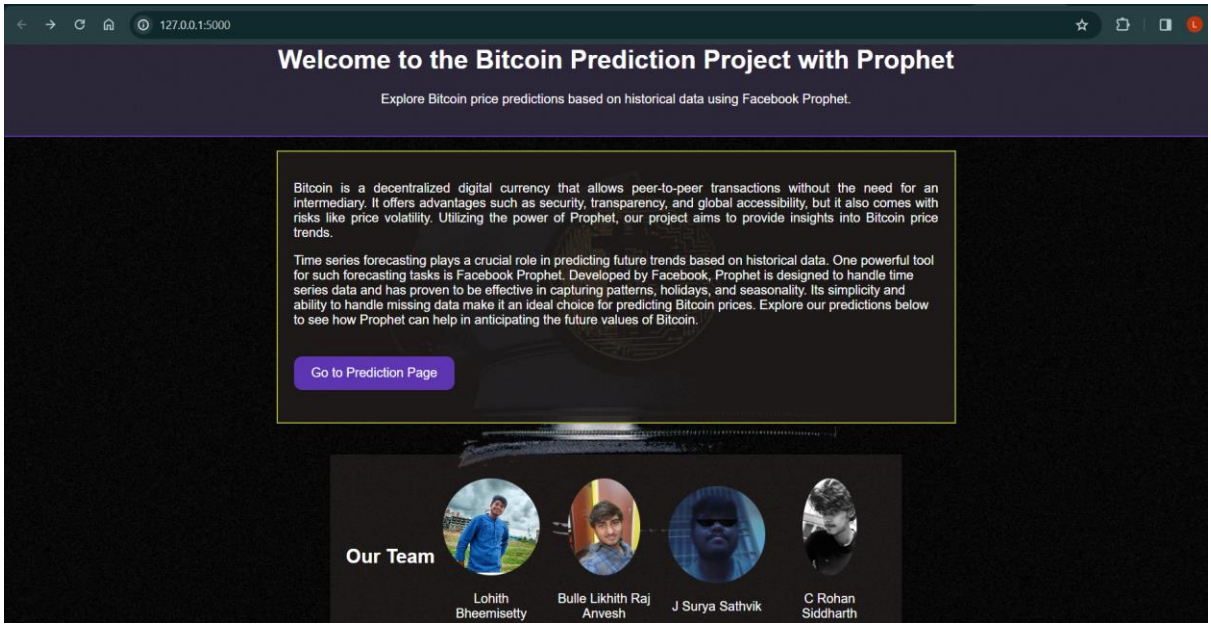
Style.css(for index page):




```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
.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
48          button:hover {
49              background-color: #311B92;
50          }
51
52          /* Add additional styles as needed */
53      </style>
54  </head>
55  <body>
56      <h1>Bitcoin Price Prediction using fbProphet</h1>
57
58      <form action="/predict" method="post">
59          <label for="Date">Select Date:</label>
60          <input type="date" id="Date" name="Date" required>
61          <button type="submit">Predict</button>
62      </form>
63
64      {% if prediction_text %}
65          <p>{{ prediction_text }}</p>
66      {% endif %}
67
68  </body>
69  </html>
70
```

Activity 2: Build app.py

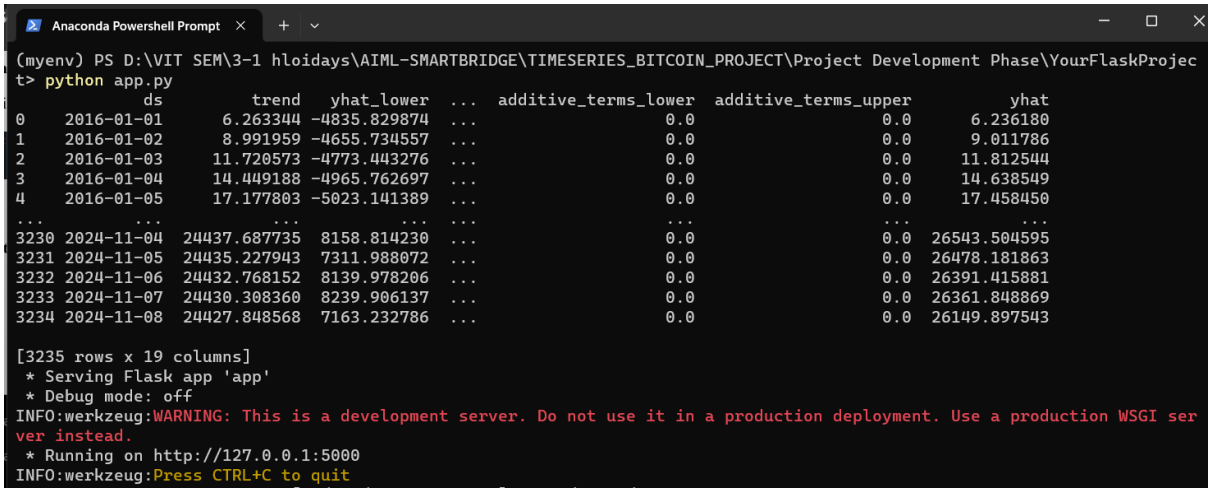
- **Task 1: Importing Libraries**
- **Task 2: Creating our flask application and loading our model by using pickle.load() method**
- **Task 3: Routing to HTML pages**
- **Task4: Making Future Prediction**
- **Taks 5: Showcasing prediction on UI**
- **Task 6: Run the application**

```
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.

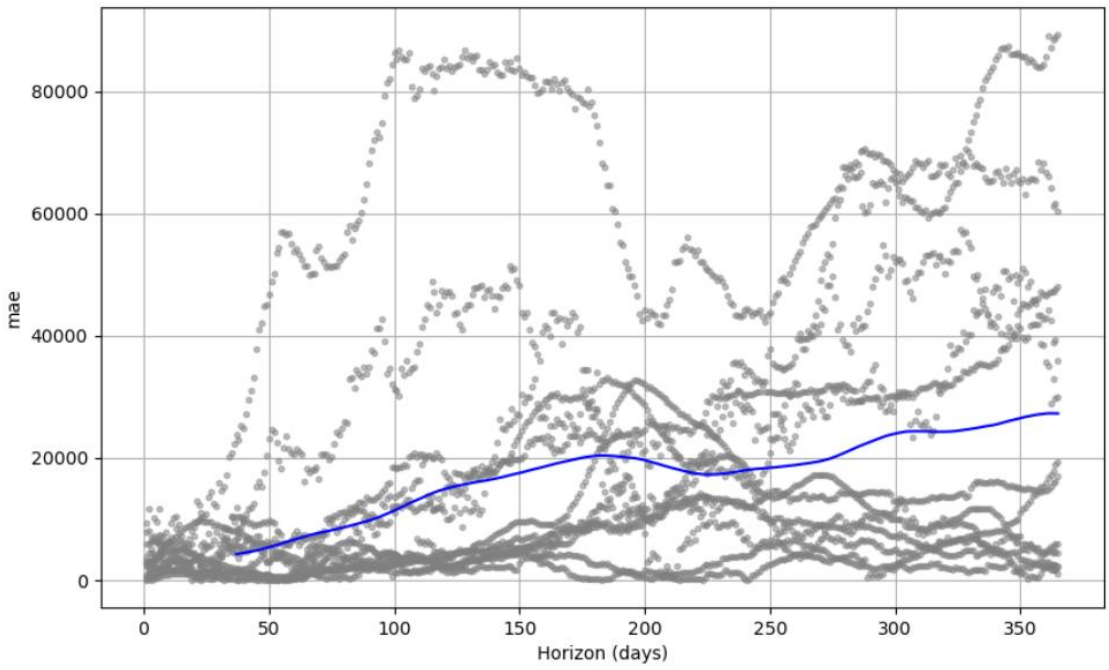


8. Performance Testing

"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."

In [26]:
```
from fbprophet.diagnostics import performance_metrics
from fbprophet.plot import plot_cross_validation_metric
from fbprophet.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')
```

```
INFO:fbprophet:Making 12 forecasts with cutoffs between 2017-06-08 00:00:00 and 2022-11-09 00:00:00
WARNING:fbprophet:Seasonality has period of 365.25 days which is larger than initial window. Consider increasing initial.
  0%|          | 0/12 [00:00<?, ?it/s]
  horizon            mse      rmse       mae  mape  mdape  coverage
0 37 days $30,460,837.99 $5,519.13 $4,190.97 $0.36  $0.24     $0.15
1 38 days $31,859,113.54 $5,644.39 $4,259.57 $0.37  $0.25     $0.15
2 39 days $33,694,115.73 $5,804.66 $4,340.80 $0.37  $0.25     $0.15
3 40 days $35,482,365.48 $5,956.71 $4,417.51 $0.38  $0.26     $0.15
4 41 days $37,429,333.02 $6,117.95 $4,489.15 $0.38  $0.27     $0.15
```



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."

In [27]:
```
from datetime import datetime
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
import numpy as np

# Assuming 'data_test' is your DataFrame containing true values and 'forecast' is the DataFrame with predicted values
y_true = df1['y']
forecast_before_nov_9 = forecast[forecast['ds'] <= datetime(2023, 11, 9)]

# Extract yhat column from the filtered DataFrame
y_pred = forecast_before_nov_9['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): 14187191.901268193
Mean Absolute Error (MAE): 2605.9734316735935
R-squared (R2): 0.9455552766056716
Root Mean Squared Error (RMSE): 3766.588894645684
```

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 perfor med to optimize its performance. Various hyperparameters, such as seasonality, holidays, and growth components, were adjusted to enhance the model's accurac

y. Additionally, a validation method was employed to assess the model's perfor mance on unseen data, ensuring robustness and generalizability."

In [32]:
```
from fbprophet import Prophet

m2 = Prophet(
    changepoint_prior_scale=0.5,  # Adjust as needed
    seasonality_prior_scale=10,     # Adjust as needed
    # Add more hyperparameters as necessary
)
# Fit the model
m2.fit(df1)


future=m2.make_future_dataframe(periods=365)
forecast=m2.predict(future)
y_true = df1['y']
forecast_before_nov_9 = forecast[forecast['ds'] <= datetime(2023, 11, 9)]
# Extract yhat column from the filtered DataFrame
y_pred = forecast_before_nov_9['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}")
```

```
INFO:fbprophet:Disabling daily seasonality. Run prophet with daily_seasonality=True to override this.
```

```
Mean Squared Error (MSE): 8006516.28151861
Mean Absolute Error (MAE): 1897.6986625308068
R-squared (R2): 0.9692742180881827
Root Mean Squared Error (RMSE): 2829.578816982946
```

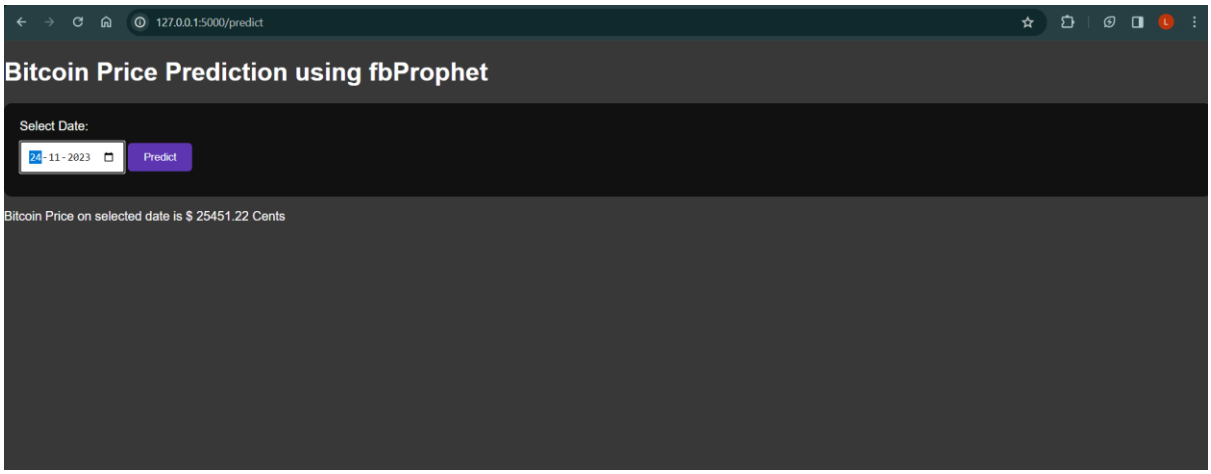
As we can see R2 score have been increased, that means we have used hyperpar ameters and increased our models accuracy.

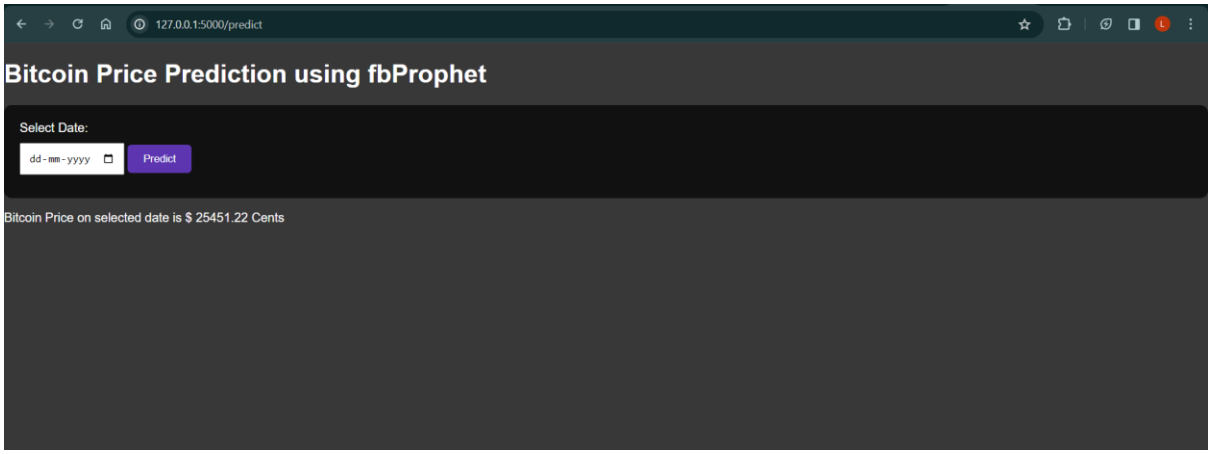
This example demonstrates adjusting **changepoint_prior_scale and seasonalit y_prior_scale.** Feel free to experiment with these and other parameters based o n your specific dataset and requirements. Additionally, you can include custom seasonality components using add_seasonality if needed. Cross-validation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

9. Results

Output Screenshots of prediction:

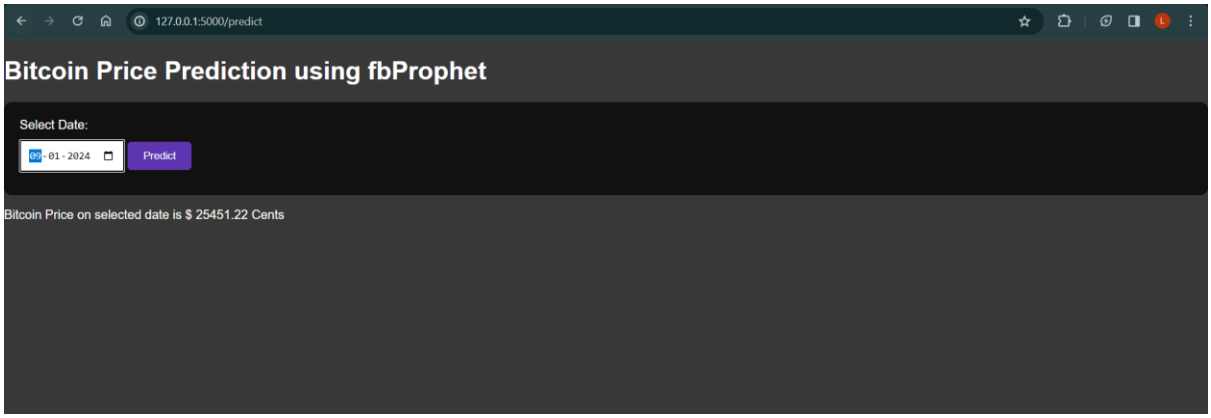
- **Select a date in predict page**

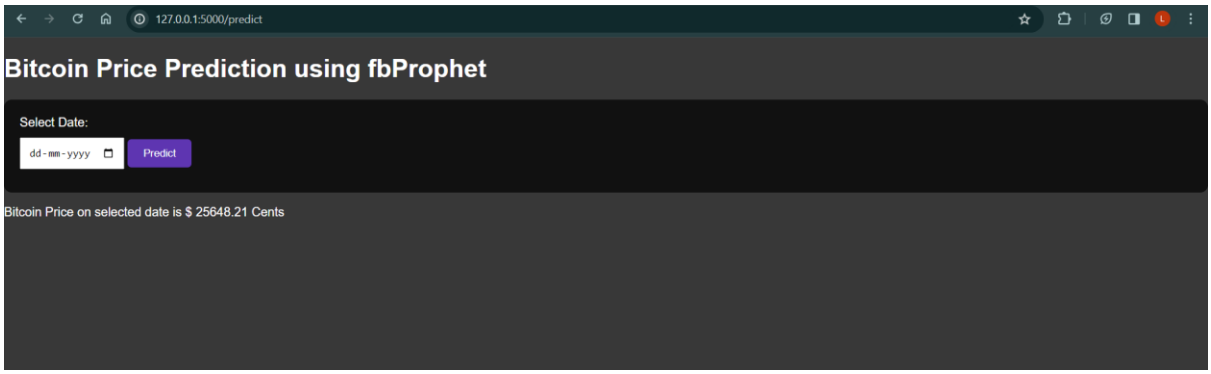




From this we can see we have predicted the bitcoin price on 24-11-23 as $25,451.22

Input 2:





From this we can see we have predicted the bitcoin price on 09-01-2024 as $25,648.21

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.

This educational component contributes to users' understanding of market dynamics.

6. **User Engagement:**

- Description: Interactive charts and graphs engage users visually, facilitating a deeper understanding of predicted trends. This engagement factor contributes to a more immersive and effective user experience.

7. **Accessibility:**

- Description: The forecasting tool is accessible to a broad audience, including both seasoned cryptocurrency enthusiasts and those new to the market. This inclusivity aligns with the project's goal of making cryptocurrency insights available to everyone.

8. **Risk Mitigation:**

- Description: By offering accurate predictions, the tool assists users in identifying potential risks and market fluctuations. This proactive approach to risk mitigation is valuable for investors aiming to protect their assets.

9. **Continuous Improvement:**

- Description: The project prioritizes continuous improvement based on user feedback and market dynamics. Regular updates and refinements ensure that the forecasting model remains adaptive and effective in evolving market conditions.

10. **Versatility:**

- Description: The forecasting model's versatility allows users to explore various scenarios and timeframes. This adaptability ensures that the tool remains relevant for users with diverse investment goals and strategies.

Disadvantages

1. **Market Volatility Challenges:**

- Description: The forecasting model may face challenges during periods of extreme market volatility. Sudden and unpredictable price movements can impact the accuracy of predictions, leading to potential discrepancies.

2. **Data Dependency:**

- Description: The model heavily relies on historical Bitcoin data. In situations where historical trends deviate from typical patterns, the forecasting accuracy may be compromised, highlighting the inherent dependence on past data.

3. **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.

4. **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.

5. **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.

6. **Learning Curve:**

- **Description**: Users unfamiliar with cryptocurrency markets or forecasting models might face a learning curve. The effectiveness of the tool relies on users' understanding of how to interpret and apply the generated predictions.

7. **Resource Intensiveness:**

- **Description:** Depending on the complexity of the forecasting algorithm and the volume of data processed, the tool might be resource-intensive. This can impact the speed and efficiency of real-time updates, especially during peak market activity.

8. **Uncertain Regulatory Environment:**

- **Description**: The cryptocurrency market operates in a dynamic regulatory environment. Changes in regulations, legal frameworks, or government policies can introduce uncertainties that affect the model's accuracy.

9. **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.

10. **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.

6. **Mobile Application Development:** Developing a mobile application version of the forecasting tool can enhance accessibility, allowing users to stay informed and make decisions on-the-go.

7. **Community Features:** Introducing community features, such as forums or social elements, can facilitate knowledge sharing among users, creating a collaborative environment for discussing trends and strategies.

8. **Blockchain Integration**: Exploring integration with blockchain technology for added transparency and security in data management can be a futuristic enhancement.

9. **Algorithm Explainability:** Addressing the interpretability of forecasting algorithms can build trust among users by providing insights into how predictions are generated.

10. **Global Expansion:** Expanding the platform's coverage to include a broader range of cryptocurrencies and markets globally can attract a more diverse user base and cater to a wider audience.

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 Demonsteration.

https://drive.google.com/drive/folders/1H7qSQYisHb-8JGdEGBFDY5ghLNAKyZWc?usp=sharing

Github link:

https://github.com/smartinternz02/SI-GuidedProject-603104-1697557017/tree/main

Project Demo Link:

https://drive.google.com/drive/folders/1_7M-LTuQB-WKEi7shLm1UkNfMogu_hTA?usp=sharing

THANK YOU