# Overview of tasks

### \*\*1. Project Planning and Requirements Gathering\*\*

- Define the core features of the web app (e.g., data upload, model training, forecasting visualization).

- Identify the time series forecasting models to be used (e.g., ARIMA, LSTM, Prophet).

- Set up the Agile workflow: create user stories, tasks, and sprints.

### \*\*2. Tech Stack and Architecture\*\*

- \*\*Frontend\*\*: Use Flask or FastAPI for the web framework, with HTML/CSS/JavaScript for the UI.

- \*\*Backend\*\*: Python for data processing, model training, and API endpoints.

- \*\*Version Control\*\*: GitHub for code repository and collaboration.

- \*\*Deployment\*\*: AWS services (EC2 for hosting, S3 for data storage, RDS for databases, Lambda if needed for serverless tasks).

### \*\*3. Development Workflow\*\*

- \*\*Sprint Planning\*\*: Break down tasks into manageable sprints (e.g., 1-2 weeks per sprint).

- \*\*Task Assignments\*\*: Prioritize and assign tasks using GitHub Issues or a project management tool like Jira or Trello.

- \*\*Code Development\*\*: Write code, commit regularly, and use pull requests for code review.

### \*\*4. Setting Up the Environment\*\*

- Set up the development environment with virtual environments or Docker.

- Initialize the GitHub repository and set up CI/CD pipelines with GitHub Actions.

### \*\*5. Backend Development\*\*

- Implement data preprocessing modules (cleaning, normalization).

- Develop forecasting models and optimize them.

- Build API endpoints for model training and prediction.

### \*\*6. Frontend Development\*\*

- Create the user interface for data input, model selection, and forecast visualization (charts, tables).

- Connect the frontend to backend APIs.

### \*\*7. Testing and Validation\*\*

- Unit tests, integration tests, and user acceptance testing.

- Ensure the model performance meets the requirements (accuracy, speed).

### \*\*8. Deployment to AWS\*\*

- Set up the AWS environment, including EC2 instances, S3 buckets, RDS, etc.

- Deploy the application and ensure it’s scalable and secure.

### \*\*9. Monitoring and Maintenance\*\*

- Set up monitoring for the app (CloudWatch, logging).

- Plan for regular updates, bug fixes, and feature enhancements.

### \*\*10. Documentation and Training\*\*

- Create comprehensive documentation for users and developers.

- Provide training or onboarding for any team members or end-users.

# Project Planning and Requirements Gathering

### \*\*1. Define the Project Objectives\*\*

- What is the primary goal of the web app? (e.g., enabling users to upload time series data and get accurate forecasts).

- Who are the target users? (e.g., data analysts, business professionals, researchers).

- What problem are we solving? (e.g., helping businesses predict sales, inventory levels, or other time-dependent metrics).

### \*\*2. Core Features and Functional Requirements\*\*

- \*\*Data Input\*\*: Allow users to upload time series data (CSV, Excel).

- \*\*Data Preprocessing\*\*: Basic data cleaning and validation to handle missing values, outliers, etc.

- \*\*Model Selection\*\*: Users can select forecasting models (e.g., ARIMA, LSTM, Prophet).

- \*\*Model Training\*\*: Train models on user data with adjustable parameters.

- \*\*Forecast Visualization\*\*: Display results in a clear, interactive format (e.g., line charts, confidence intervals).

- \*\*Forecast Export\*\*: Allow users to download the forecast results (CSV, Excel, PDF).

- \*\*User Authentication\*\*: Secure login with account management features.

- \*\*API Access\*\*: Optional, for advanced users to interact programmatically.

### \*\*3. Non-Functional Requirements\*\*

- \*\*Performance\*\*: Ensure the app is responsive and can handle multiple requests simultaneously.

- \*\*Scalability\*\*: Design the architecture to scale with increasing data volumes and user requests.

- \*\*Security\*\*: Protect data privacy and implement secure data handling practices.

- \*\*Availability\*\*: Ensure high availability with minimal downtime.

### \*\*4. Key Milestones\*\*

- \*\*Milestone 1\*\*: Project setup, GitHub repository creation, basic architecture defined.

- \*\*Milestone 2\*\*: Backend API for data upload and preprocessing.

- \*\*Milestone 3\*\*: Model implementation and integration.

- \*\*Milestone 4\*\*: Frontend development for data input and forecast visualization.

- \*\*Milestone 5\*\*: Testing and deployment to AWS.

- \*\*Milestone 6\*\*: Final review, user testing, and launch.

### \*\*5. Agile Workflow Setup\*\*

- \*\*Sprint Duration\*\*: Decide on sprint lengths (e.g., 1-2 weeks).

- \*\*User Stories\*\*: Define user stories that describe each feature from the end-user perspective (e.g., “As a user, I want to upload my data easily so that I can forecast future trends.”).

- \*\*Task Management\*\*: Use GitHub Projects, Jira, or Trello to manage tasks and track progress.

### \*\*6. Risk Assessment\*\*

- Identify potential risks (e.g., data privacy concerns, model accuracy issues, deployment challenges).

- Define mitigation strategies.

## Project Objectives

### \*\*Project Objectives Update:\*\*

1. \*\*Primary Goal\*\*: Produce accurate forecasts from data collected via APIs or through manual uploads by users.

### \*\*Additional Considerations for Objectives:\*\*

- \*\*API Integration\*\*: Define which APIs will be integrated (e.g., financial data, weather data, sales data). This could affect model selection and preprocessing steps.

- \*\*Data Handling\*\*: Ensure robust mechanisms for handling both API and manually uploaded data, including data validation, error handling, and format standardization.

- \*\*Model Accuracy\*\*: Implement performance metrics to evaluate the accuracy of forecasts (e.g., MAPE, RMSE) and allow users to compare results from different models.

## Use cases

### \*\*1. Retail and E-commerce\*\*

- \*\*Use Case\*\*: Inventory management, sales forecasting, and promotional planning.

- \*\*Need\*\*: Predicting customer demand, peak shopping periods, and stock replenishment to minimize overstock or stockouts.

### \*\*2. Supply Chain and Logistics\*\*

- \*\*Use Case\*\*: Optimizing routes, warehouse management, and demand planning.

- \*\*Need\*\*: Forecasting demand to ensure timely delivery and minimize transportation costs.

### \*\*3. Energy and Utilities\*\*

- \*\*Use Case\*\*: Forecasting energy consumption, managing grid loads, and planning maintenance.

- \*\*Need\*\*: Predicting electricity demand or renewable energy production (e.g., solar, wind) to balance supply and demand efficiently.

### \*\*4. Finance and Investment\*\*

- \*\*Use Case\*\*: Market trend prediction, stock price forecasting, and risk management.

- \*\*Need\*\*: Forecasting economic indicators, stock prices, or portfolio performance to make informed investment decisions.

### \*\*5. Healthcare\*\*

- \*\*Use Case\*\*: Patient admission forecasts, resource allocation, and medical inventory management.

- \*\*Need\*\*: Predicting hospital admissions, staffing needs, or medication demand to optimize healthcare delivery.

### \*\*6. Manufacturing\*\*

- \*\*Use Case\*\*: Production planning, quality control, and inventory management.

- \*\*Need\*\*: Forecasting raw material requirements, machine maintenance, and production cycles to reduce waste and downtime.

### \*\*7. Telecommunications\*\*

- \*\*Use Case\*\*: Network traffic forecasting, capacity planning, and customer usage patterns.

- \*\*Need\*\*: Predicting data usage to optimize bandwidth allocation and reduce network congestion.

### \*\*8. Travel and Hospitality\*\*

- \*\*Use Case\*\*: Predicting bookings, staffing, and resource management.

- \*\*Need\*\*: Forecasting hotel occupancy rates, flight bookings, or restaurant reservations to adjust staffing and resources accordingly.

### \*\*9. Agriculture\*\*

- \*\*Use Case\*\*: Crop yield prediction, pest control, and resource planning.

- \*\*Need\*\*: Forecasting weather patterns, soil moisture levels, or market prices to optimize planting and harvesting schedules.

### \*\*10. Automotive and Transportation\*\*

- \*\*Use Case\*\*: Predictive maintenance, fleet management, and traffic forecasting.

- \*\*Need\*\*: Forecasting vehicle maintenance needs, fuel consumption, or traffic patterns to optimize operations.

### \*\*11. Real Estate\*\*

- \*\*Use Case\*\*: Market analysis, property value forecasting, and rental demand prediction.

- \*\*Need\*\*: Predicting market trends, rental income, or property demand for better investment decisions.

### \*\*12. Consumer Goods\*\*

- \*\*Use Case\*\*: Product launch forecasting, trend analysis, and customer demand prediction.

- \*\*Need\*\*: Anticipating market response to new products, seasonal demand, or shifts in consumer preferences.

# Core Features and Functional Requirements

1. \*\*Data Input / Data Collection\*\*:

- \*\*Manual Upload\*\*: Allow users to upload time series data files (CSV, Excel).

- \*\*API Integration\*\*: Collect data directly from various APIs (e.g., financial data, sales data).

- \*\*Web Scraping\*\*: Gather data from websites when APIs are unavailable, ensuring compliance with data use policies.

2. \*\*Data Preprocessing\*\*:

- Basic data cleaning and validation, including handling missing values, outliers, and standardizing formats.

3. \*\*Model Selection\*\*:

- Provide options for users to select from various forecasting models (e.g., ARIMA, LSTM, Prophet).

4. \*\*Model Training\*\*:

- Train models on user data with adjustable parameters, allowing users to fine-tune their forecasting results.

5. \*\*Forecast Visualization\*\*:

- Display forecast results interactively using line charts, confidence intervals, and other visual aids.

6. \*\*Forecast Export\*\*:

- Allow users to download forecast results in different formats (CSV, Excel, PDF).

7. \*\*User Authentication\*\*:

- Secure login system with account management for personalized access and data security.

8. \*\*API Access\*\*:

- Optional API endpoints for advanced users who wish to interact with the forecasting models programmatically.