

**Forecast360 — Functional Specification**

Version 1.0 • iSOFT ANZ Pvt Ltd



# Document Control

|  |  |  |  |
| --- | --- | --- | --- |
| **Author** | **Version** | **Status** | **Date** |
| Amitesh Jha | 1.0 | Draft | 01/10/2025 |

Table of Contents

[Document Control 2](#_Toc210209455)

[1. Overview 5](#_Toc210209456)

[1.1 Purpose 5](#_Toc210209457)

[1.2 Goals 5](#_Toc210209458)

[1.3 Out of Scope 5](#_Toc210209459)

[2. Users & Personas 5](#_Toc210209460)

[3. High-Level Flow 6](#_Toc210209461)

[4. Functional Requirements 6](#_Toc210209462)

[4.1 Data Ingestion 6](#_Toc210209463)

[4.2 Column Detection & Selection 6](#_Toc210209464)

[4.3 Data Preparation 7](#_Toc210209465)

[4.4 Exogenous Features 7](#_Toc210209466)

[4.5 Pattern & Seasonality 7](#_Toc210209467)

[4.6 Validation & Metrics 7](#_Toc210209468)

[4.7 Model Families (Selectable) 7](#_Toc210209469)

[4.8 Leaderboard & Best Model 7](#_Toc210209470)

[4.9 Forecasting 7](#_Toc210209471)

[4.10 Narrative Summary 8](#_Toc210209472)

[5. UI/UX 8](#_Toc210209473)

[5.1 Navigation 8](#_Toc210209474)

[5.2 Inputs (Typical Labels) 8](#_Toc210209475)

[5.3 Outputs 8](#_Toc210209476)

[6. Data & State 8](#_Toc210209477)

[7. Non-Functional Requirements 9](#_Toc210209478)

[7.1 Performance 9](#_Toc210209479)

[7.2 Reliability & Error Handling 9](#_Toc210209480)

[7.3 Security & Privacy 9](#_Toc210209481)

[7.4 Accessibility & Usability 9](#_Toc210209482)

[8. Dependencies 9](#_Toc210209483)

[8.1 Core 9](#_Toc210209484)

[8.2 Optional 9](#_Toc210209485)

[9. Testing Strategy 9](#_Toc210209486)

[10. Open Questions / Decisions 10](#_Toc210209487)

# 1. Overview

## 1.1 Purpose

Forecast360 is a Streamlit-based application for time-series exploration, preparation, modeling, and forecasting. It guides users from data ingestion through preprocessing and model selection to evaluation and forecast generation.

## 1.2 Goals

* Lower the barrier to running robust time-series forecasts for analysts and business users.
* Provide a guided “Getting Started” flow from data ingestion → cleaning → feature setup → modeling → evaluation.
* Present clear, interpretable outputs (leaderboards, best-model highlights, forecast plots, downloadable tables).

## 1.3 Out of Scope

* Authentication/authorization or multi-tenant account management.
* Long-term experiment tracking or model registry.
* Distributed training or large-scale hyperparameter search.
* External database writes (outputs are downloaded by the user).

# 2. Users & Personas

* Business Analyst — uploads data, configures basic settings, exports forecasts.
* Data Scientist — tunes preprocessing, selects models, inspects metrics and iterates.
* Executive / Stakeholder — consumes charts, best-model highlights, and summary.

# 3. High-Level Flow

|  |
| --- |
|  |

1. **Upload Data:** The starting point, accepting various file formats like CSV, XLSX, JSON, Parquet, and XML.
2. **Auto-Detect & Select Columns:** Identifying and choosing the necessary date and target columns from the uploaded data.
3. **Data Preparation:** This crucial stage cleans and formats the data, involving resampling, missing-value handling, and optional outlier treatment & transforms.
4. **Exogenous Features:** Incorporating external, predictive variables such as calendar features, lags of the target, additional columns, and scaling for normalization.
5. **Backtesting Configuration:** Setting up the model evaluation framework, which defines folds, the forecasting horizon, the gap between training and testing data, and the evaluation metrics.
6. **Model Family Selection:** Choosing the appropriate forecasting algorithms, including statistical models (ARIMA variants, HWES, Prophet, TBATS) and machine learning/deep learning models (XGBoost/LightGBM, TFT).
7. **Run & Evaluate:** Executing the selected models and assessing their performance. This step generates metrics per model, a leaderboard for comparison, and identifies the best model.
8. **Forecast Generation:** The final output stage, which produces the forecast in multiple formats, including charts, a table, and a CSV download.

# 4. Functional Requirements

## 4.1 Data Ingestion

* Accept uploads: CSV (.csv), Excel (.xlsx/.xls), JSON (.json), Parquet (.parquet), XML (.xml).
* Show preview of loaded data and store original shape (raw\_rows, raw\_cols).
* Report read/parse errors gracefully in the UI.

## 4.2 Column Detection & Selection

* Infer date column via heuristics (name patterns, datetime convertibility).
* Infer numeric target column using common keywords/first numeric column fallback.
* Allow user overrides for both via dropdowns.

## 4.3 Data Preparation

* Resampling by user-chosen frequency (e.g., W/D/M).
* Missing-value strategies: median, mean, ffill, bfill, zero, constant=<value>.
* Optional outlier handling (Winsorization by Z-score threshold).
* Optional target transforms (none by default).
* Track cleaned shape (clean\_rows, clean\_cols) and drop breakdown for summary.

## 4.4 Exogenous Features

* Calendar features toggle (e.g., day of week, month, holidays if available).
* Lags: user-specified list (default [0, 1, 7]).
* Additional columns: multiselect or free-text to include from dataset.
* Scaling toggle for exogenous matrix.
* Join exogenous features with target series aligned on date.

## 4.5 Pattern & Seasonality

* Pattern type selection: Auto-detect (hooks for decomposition/ACF if present).
* Seasonal period m: user-input or auto.

## 4.6 Validation & Metrics

* Backtesting: folds, horizon, gap configuration.
* Metrics: RMSE, MAE, MAPE, sMAPE, MASE, optional R²; criteria like AIC/BIC where applicable.
* Compute per-model fold metrics and aggregate into a leaderboard.

## 4.7 Model Families (Selectable)

* Traditional: ARMA / ARIMA / ARIMAX; SARIMA / SARIMAX; HWES; Auto-ARIMA (if pmdarima available).
* Prophet and TBATS (optional if installed).
* Gradient Boosting: XGBoost and LightGBM.
* TFT / deep learning (optional, if installed).
* Feature importance where available (coefficients, tree importances, optional permutation).

## 4.8 Leaderboard & Best Model

* Display leaderboard sorted by primary metric (e.g., RMSE).
* Highlight best model (lowest RMSE or configured rule).
* Provide compact top-k table for quick scan.

## 4.9 Forecasting

* User sets forecast horizon (steps).
* Generate predictions for the chosen/best model.
* Show combined chart (history + forecast) and forecast table.
* Provide CSV download for the forecasted values.

## 4.10 Narrative Summary

* Render polished English narrative at the end of Getting Started.
* Include Key facts & figures: file, shape (original → cleaned), dropped rows, date range & span, resampling, missing handling, exog config, seasonality, transforms, outliers, CV config, models, best model/metrics.

# 5. UI/UX

## 5.1 Navigation

* Sidebar: Getting Started flow with file upload and configuration controls.
* Main Panel: Sectioned content—preview/EDA, diagnostics, leaderboard, forecast, summary.

## 5.2 Inputs (Typical Labels)

* Upload: Data Upload; XML row path (optional XPath) for XML.
* Selectors: Date column, Target column.
* Data Prep: Resample frequency, Missing value strategy (+constant), Outlier Z threshold, Transform.
* Exogenous: Calendar toggle, Lags, Additional columns, Scaling toggle.
* Validation: Folds, Horizon, Gap, Metrics.
* Models: ARIMA variants, HWES, Prophet, TBATS, XGB, LGBM, TFT.
* Forecast: Forecast horizon (steps).

## 5.3 Outputs

* Preview table; charts for series/diagnostics; leaderboard; forecast chart & table.
* Download forecast CSV.
* Narrative summary (English + facts list).

# 6. Data & State

Key session state variables include (not exhaustive):

* source\_name, raw\_rows, raw\_cols, clean\_rows, clean\_cols, drop\_breakdown
* uploaded\_df, date\_col, target\_col
* resample\_freq, missing\_values, const\_missing\_value
* use\_calendar\_exog, exog\_lags, exog\_additional\_cols, scale\_exog
* pattern\_type, seasonal\_m, target\_transform, winsorize, outlier\_z
* cv\_folds, cv\_horizon, cv\_gap, cv\_metrics
* leaderboard\_df / leaderboard\_rows / best\_model\_name
* forecast\_horizon

Data model: input time series (date + numeric target) → cleaned ('ds','y') with optional aligned exogenous matrix → leaderboard + forecast series ('yhat').

# 7. Non-Functional Requirements

## 7.1 Performance

* Handle datasets ≈ up to 100k rows per session.
* Avoid heavy O(N²) operations on full data (limit correlation heatmaps).
* Cache expensive steps where feasible (optional).

## 7.2 Reliability & Error Handling

* User-friendly errors for unsupported formats, bad date parsing, or missing optional libs.
* Fail soft when optional libraries are unavailable (Prophet, TBATS, etc.).

## 7.3 Security & Privacy

* Files processed in-memory for the session; no external persistence by default.
* Advise users not to upload sensitive data.

## 7.4 Accessibility & Usability

* Clear labels and helper text; adequate contrast.
* Avoid color-only encodings in charts when possible.
* Keyboard/screen-reader friendly defaults (Streamlit).

# 8. Dependencies

## 8.1 Core

* Python 3.11–3.13
* Streamlit ≈1.36–1.39
* pandas, numpy, pyarrow
* matplotlib, plotly
* statsmodels, scikit-learn

## 8.2 Optional

* pmdarima (Auto-ARIMA)
* prophet, tbats
* xgboost, lightgbm
* pytorch + TFT (if deep learning enabled)

# 9. Testing Strategy

* Unit: data readers, imputation logic, feature builders.
* Integration: end-to-end on sample datasets per file type.
* UI smoke: app loads, controls render, minimal forecast runs without error.
* Regression: fixed datasets + expected metric snapshots.

# 10. Open Questions / Decisions

* Primary metric for best model (RMSE vs MAPE) — confirm baseline choice.
* Confidence/prediction intervals across models — include or omit?
* Enable deep models (TFT) by default — dependency trade-off?
* Project save/load beyond CSV downloads — scope?