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

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).

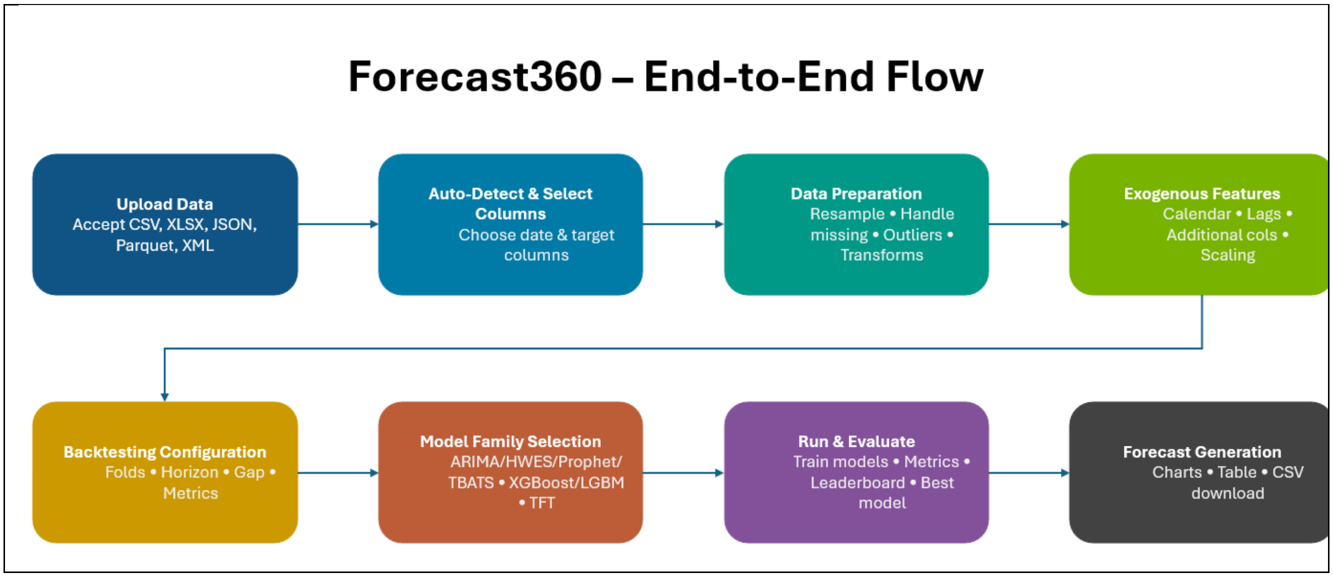
1.3 Out of Scope

- 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


Forecast360 – End-to-End Flow
Upload Data
Accept CSV, XLSX, JSON, Parquet, XML
Auto-Detect & Select Columns
Choose date & target columns
Data Preparation
Resample • Handle missing • Outliers • Transforms
Exogenous Features
Calendar • Lags • Additional cols • Scaling
Backtesting Configuration
Folds • Horizon • Gap • Metrics
Model Family Selection
ARIMA/HWES/Prophet/ TBATS • XGBoost/LGBM • TFT
Run & Evaluate
Train models • Metrics • Leaderboard • Best model
Forecast Generation
Charts • Table • CSV download

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 R^2; 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 $O(N^2)$ 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?