Auto Growth Suggestion API for Financial Modeling & Forecasting

"An API-Driven Approach to Financial Forecasting, Growth Analysis, and Decision Support"

Prepared For: Finance professionals, financial analysts, investment researchers, and decision-makers seeking API-driven solutions for automated financial growth forecasting and modeling.

Prepared By: Rahul Muneshwar Bhoyar

Date: December 2025

Project Description Summary

The AI-Based Auto Growth Suggestion System for Financial Modeling is an intelligent, API-driven financial decision-support platform designed to automate growth analysis and forecasting within financial models. The project aims to address the limitations of traditional financial modeling methods that rely heavily on static assumptions, manual calculations, and spreadsheet-centric workflows, which often fail to adapt efficiently to changing financial conditions.

In real-world financial environments, growth indicators such as revenue expansion, cost efficiency, and profitability are influenced by multiple dynamic factors including market trends, operational performance, and macro-economic conditions. Conventional financial models typically apply fixed growth rates and require frequent manual updates, increasing the risk of inconsistency, human error, and delayed decision-making. This project introduces a modular and reusable solution that encapsulates growth logic within a centralized API, enabling standardized and automated growth suggestions across financial models.

The core component of the system is a FastAPI-based Auto Growth Suggestion Engine, which exposes a RESTful endpoint capable of generating growth insights in a structured JSON format. Financial inputs and assumptions are processed through predefined analytical logic to derive growth indicators and strategic recommendations. By abstracting growth analysis into an API layer, the system allows financial models, dashboards, and planning tools to consume consistent growth suggestions without embedding complex logic directly into spreadsheets or front-end applications.

The system leverages Swagger (OpenAPI) documentation to provide transparency, testability, and ease of integration. Analysts and stakeholders can interactively validate the API outputs, ensuring that growth suggestions remain explainable and auditable. This approach enhances confidence in the financial modeling process while reducing dependency on opaque calculations.

From an architectural perspective, the project follows a clean separation of concerns. Data inputs, growth computation, and output recommendations are handled independently, enabling scalability and future enhancement. The API-driven design also allows seamless integration with enterprise financial planning systems, business intelligence dashboards, or Excel-based models.

Overall, the AI-Based Auto Growth Suggestion System for Financial Modeling demonstrates how automation and intelligent API design can modernize financial analysis workflows. The project highlights the practical application of AI-inspired decision logic to improve forecasting accuracy, reduce manual effort, and support strategic financial decision-making in corporate finance, investment analysis, and business planning environments.

Problem Statement

Financial modeling plays a critical role in corporate planning, investment analysis, budgeting, and strategic decision-making. Organizations rely on financial models to forecast revenue growth, estimate future performance, evaluate investment opportunities, and assess financial risk. Despite its importance, traditional financial modeling practices remain heavily dependent on static assumptions, manual calculations, and spreadsheet-driven workflows.

In most conventional financial models, growth rates are predefined and remain constant across forecasting periods. These assumptions are often based on historical averages or subjective estimates, making the models inflexible to changing market conditions. As financial environments evolve due to competition, economic cycles, operational changes, and regulatory factors, static growth assumptions quickly become outdated. This leads to inaccurate forecasts, delayed responses, and poor strategic decisions.

Another significant challenge is the lack of standardization in growth analysis. Financial growth logic is typically embedded directly within spreadsheets or individual applications, resulting in inconsistent methodologies across departments or projects. Analysts frequently duplicate formulas, modify assumptions manually, and apply different growth rules for similar scenarios. This increases the risk of calculation errors, reduces transparency, and makes validation or audit of financial models difficult.

Manual financial modeling also limits scalability. As organizations expand and analyze multiple scenarios, business units, or investment options, maintaining and updating individual models becomes time-consuming and inefficient. Spreadsheet-based approaches do not provide a centralized mechanism to manage growth logic, leading to fragmented financial planning processes.

Furthermore, existing tools offer limited explainability and real-time adaptability. Most systems generate outputs without clearly separating growth logic from presentation, making it difficult for stakeholders to understand how growth recommendations are derived. This lack of explainability reduces confidence in financial forecasts and hinders collaborative decision-making.

There is a clear need for an automated, modular, and reusable system that can generate consistent growth suggestions for financial models while remaining transparent, scalable, and easy to integrate. Such a system should allow financial growth logic to be centralized, exposed through a standardized interface, and consumed by multiple financial models and analytical tools.

This project addresses these challenges by proposing an **AI-Based Auto Growth Suggestion System** that leverages an API-driven architecture to automate growth analysis. By abstracting growth logic into a centralized FastAPI service, the system aims to reduce manual dependency, improve forecasting accuracy, ensure consistency across models, and enhance decision support in financial modeling.

Introduction

Financial modeling is a fundamental tool used by organizations to evaluate business performance, forecast future outcomes, and support strategic decision-making. It plays a crucial role in areas such as corporate finance, investment analysis, budgeting, valuation, and long-term planning. Financial models help decision-makers estimate growth trends, assess profitability, and understand the potential impact of strategic choices under different assumptions.

Despite its importance, traditional financial modeling techniques largely depend on static assumptions, manual spreadsheet calculations, and repetitive adjustments. Growth rates, revenue projections, and cost assumptions are often fixed based on historical averages or subjective judgement. While this approach may be adequate in stable environments, it becomes ineffective in dynamic financial conditions where market trends, operational performance, and economic factors change frequently. As a result, many financial models fail to reflect real-time business realities and require continuous manual intervention.

Another limitation of conventional financial modeling is the lack of modularity and standardization. Growth logic is commonly embedded directly within spreadsheets or individual applications, making it difficult to reuse, validate, or scale across multiple financial scenarios. Analysts often replicate similar formulas across models, leading to inconsistencies, increased error rates, and reduced transparency. These challenges become more pronounced as organizations grow and manage multiple business units, projects, or investment portfolios.

With the increasing adoption of automation and intelligent systems in finance, there is a growing need for solutions that can enhance the efficiency, consistency, and reliability of financial modeling processes. Modern financial systems demand approaches that separate business logic from presentation layers, enable centralized control of assumptions, and support integration with analytical tools and dashboards. Application Programming Interfaces (APIs) have emerged as a powerful mechanism to achieve these objectives by allowing standardized access to core analytical logic.

The **AI-Based Auto Growth Suggestion System for Financial Modeling** is introduced to address these challenges. The project proposes an API-driven framework that automates the generation of growth suggestions based on predefined analytical logic. Instead of embedding growth calculations directly into spreadsheets, the system centralizes growth analysis within a FastAPI-based service that can be accessed by multiple financial models and applications.

By leveraging an API-centric design, the system promotes consistency, scalability, and explainability in financial modeling. Growth recommendations are delivered in a structured format, making them easy to validate, integrate, and audit. This approach reduces dependency on manual processes while improving forecasting accuracy and decision support.

Overall, the project demonstrates how AI-inspired automation and modern API design can transform traditional financial modeling into a more adaptive, reliable, and efficient process. It provides a practical foundation for building intelligent financial systems capable of supporting data-driven decision-making in rapidly evolving business environments.

Content Quality Analysis

The Content Quality Analysis phase evaluates the reliability, consistency, and logical correctness of the data, calculations, and outputs produced by the **AI-Based Auto Growth Suggestion System for Financial Modeling**. Since the system is designed to support financial decision-making, ensuring the quality and integrity of inputs and generated growth recommendations is critical for maintaining trust and accuracy.

Financial growth analysis depends heavily on the correctness of assumptions, structured inputs, and proportional interpretation of trends. Any inconsistency or error in financial data can significantly affect forecasts and strategic decisions. Therefore, this project incorporates systematic checks to validate financial inputs, processing logic, and output recommendations before they are consumed by financial models or decision-support tools.

Logical Consistency of Financial Inputs

The system verifies that all financial inputs follow logical relationships and accepted financial principles. For example, revenue growth trends must align with historical performance patterns and should not exhibit abrupt, unexplained fluctuations. Cost structures and margin assumptions are evaluated to ensure that growth suggestions do not contradict fundamental financial logic.

The analysis ensures that increases in projected growth are supported by corresponding changes in input indicators rather than arbitrary assumptions. This prevents unrealistic or misleading growth recommendations.

Data Integrity and Validation

Data integrity is a key focus of the content quality process. The system validates that all required financial inputs are present, correctly formatted, and within acceptable ranges. Missing, incomplete, or contradictory values are identified and handled gracefully to avoid inaccurate output generation.

Invalid or inconsistent data inputs are either corrected using default assumptions or rejected with appropriate error responses. This prevents unreliable data from influencing growth analysis and ensures that API outputs remain dependable.

Accuracy of Growth Calculation

The growth calculation logic is assessed to ensure proportional and explainable results. Growth suggestions generated by the API must reflect realistic financial behavior. For instance, minor variations in input values should result in gradual changes in growth recommendations rather than extreme or unstable outputs.

The system confirms that growth indicators increase or decrease in a controlled and interpretable manner. This allows financial analysts to understand how changes in input assumptions influence final recommendations.

Recommendation Relevance and Applicability

The content quality analysis verifies that growth recommendations are contextually relevant and practically applicable. The system ensures that recommendations align with financial modeling objectives such as revenue expansion, cost optimization, or risk mitigation.

Growth suggestions are designed to support strategic planning rather than provide rigid or prescriptive outputs. This flexibility allows analysts to incorporate API recommendations into broader financial models and decision frameworks.

Consistency Across API Responses

To maintain reliability, the system checks for consistency across multiple API calls using similar input scenarios. Repeated requests with comparable data should yield consistent growth suggestions, ensuring stability and predictability in financial modeling.

This consistency is essential for maintaining confidence in automated financial analysis and supports the system's use in scenario planning and comparative evaluation.

Explainability and Transparency

Explainability is a key quality criterion of the system. The API outputs are structured in a clear and readable format, enabling users to trace growth suggestions back to input assumptions and processing logic.

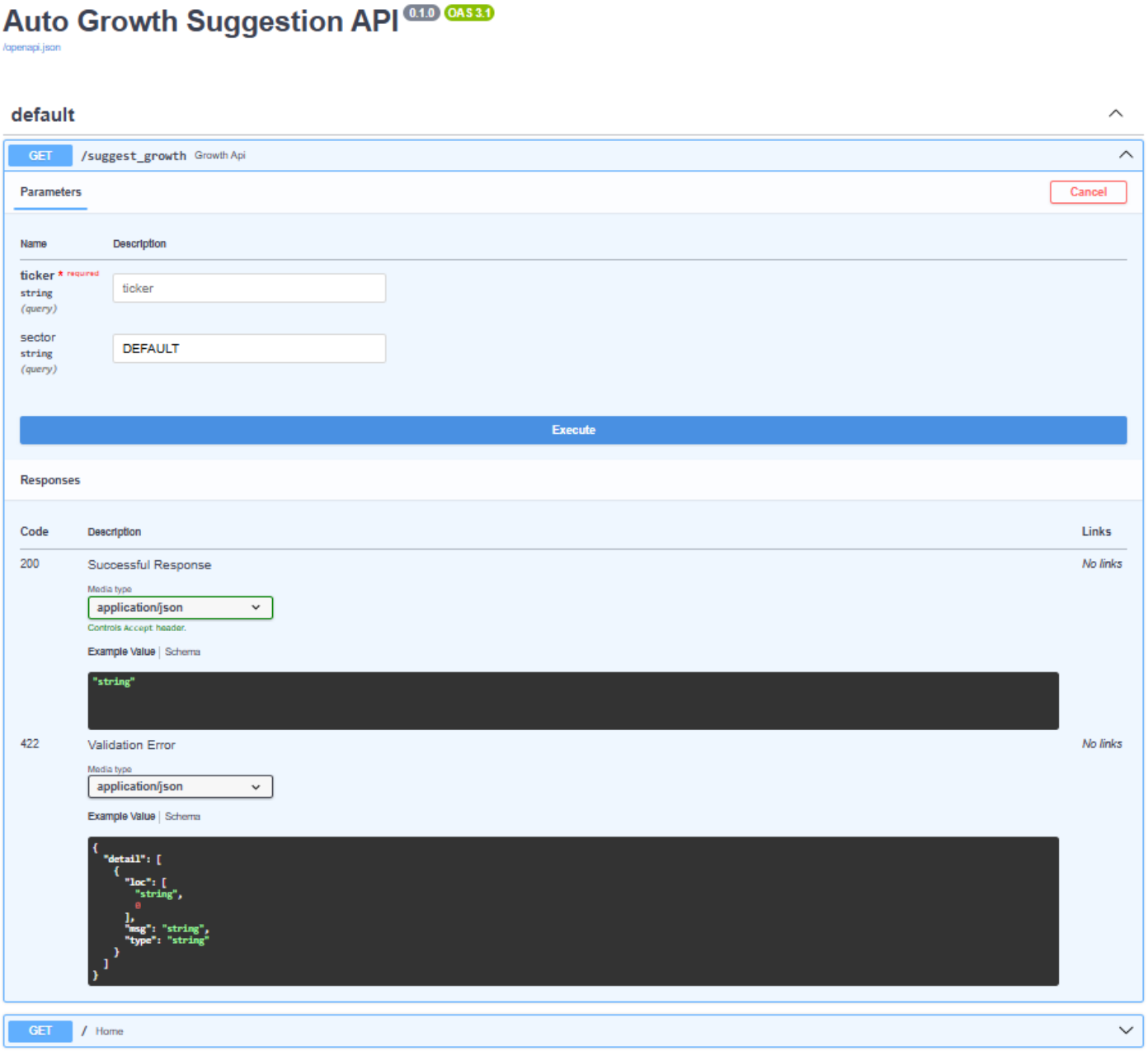
By separating growth analysis logic from presentation layers, the system improves transparency and allows financial professionals

to validate recommendations without relying on hidden or opaque calculations.

Impact of Content Quality Analysis

The Content Quality Analysis phase strengthens the credibility and effectiveness of the AI-Based Auto Growth Suggestion System for Financial Modeling. By ensuring accurate inputs, consistent logic, and reliable outputs, the system delivers trustworthy growth recommendations suitable for real-world financial planning and forecasting.

This structured approach to quality validation supports better decision-making, reduces modeling risk, and enhances the overall value of the automated financial growth suggestion framework.



Objectives

The primary objective of the AI-Based Auto Growth Suggestion System for Financial Modeling is to design and implement an intelligent, API-driven framework that automates financial growth analysis and supports data-driven decision-making. The system aims to reduce manual dependency in traditional financial modeling while improving consistency, scalability, and forecasting accuracy.

- To design and develop an API-driven system that automates financial growth analysis.
- To provide consistent and standardized growth suggestions for financial modeling and forecasting.
- To reduce dependency on manual calculations and static assumptions in traditional financial models.
- To improve accuracy, efficiency, and reliability of financial growth forecasting.
- To ensure transparency and explainability of growth recommendations for validation and decision-making.
- To support scalable and reusable integration of growth logic across multiple financial models.
- To enhance financial planning and strategic decision-making through automated growth suggestions.

System Architecture

The **AI-Based Auto Growth Suggestion System for Financial Modeling** is designed using a modular, API-driven architecture that separates data input, growth analysis logic, and output delivery. This architectural approach ensures scalability, consistency, and ease of integration with various financial modeling tools and applications.

The system follows a layered architecture in which each component performs a distinct role while interacting seamlessly with other components. By centralizing financial growth logic within an API, the system avoids duplication of calculations and improves standardization across financial models.

1. Input Layer

The Input Layer is responsible for receiving financial data and assumptions required for growth analysis. These inputs may include historical financial indicators, projected values, or predefined assumptions used in financial modeling.

This layer accepts data in a structured format, ensuring that inputs are standardized before processing. Input validation checks are applied to verify data completeness, correct formatting, and logical consistency. Any missing or invalid data is flagged to prevent incorrect growth recommendations.

2. Processing and Analysis Layer

The Processing and Analysis Layer forms the core of the system. It is responsible for applying financial growth logic to the validated input data. This layer evaluates growth indicators, trends, and assumptions to generate meaningful growth suggestions.

The analytical logic is implemented using predefined rules and AI-inspired decision logic that mimic how financial analysts evaluate growth scenarios. By isolating this logic from the presentation layer, the system ensures that growth calculations remain consistent and reusable.

3. Auto Growth Suggestion Engine

The Auto Growth Suggestion Engine is the decision-making component of the architecture. It interprets processed financial data and produces growth recommendations based on analytical outcomes.

The engine evaluates growth patterns and generates structured recommendations such as suggested growth rates, expansion indicators, or strategic guidance. These recommendations are designed to support financial forecasting, planning, and decision-making rather than replace human judgement.

4. API Layer (FastAPI)

The API Layer exposes the growth suggestion logic through a RESTful interface built using **FastAPI**. This layer acts as the communication bridge between the growth engine and external systems such as financial models, dashboards, or analytical tools.

The API returns growth suggestions in a standardized JSON format, making it easy to integrate with spreadsheets, business intelligence tools, or web applications. The use of Swagger (OpenAPI) documentation enables easy testing, validation, and explainability of API responses.

5. Output Layer

The Output Layer delivers growth recommendations to consuming systems or users. Outputs are structured, readable, and designed for easy interpretation within financial models.

This layer ensures that growth suggestions are clearly separated from raw calculations, allowing financial analysts to review, validate, and apply recommendations within broader financial frameworks.

6. Validation and Error Handling Layer

To ensure reliability, the system includes a validation and error-handling mechanism that manages incorrect inputs, logical inconsistencies, or processing failures. Clear error messages and fallback responses help maintain system stability and prevent misleading results.

7. Deployment and Access Layer

The system is deployed locally during development and testing using a local server environment. The API-based architecture allows easy extension to cloud or enterprise deployment in the future. Access to the system is controlled through API endpoints, enabling secure and scalable usage.

Architecture Layer	Technology Used
Backend Framework	FastAPI
API Documentation	Swagger (OpenAPI)
Programming Language	Python
Data Processing	Rule-based financial logic
Output Format	JSON
Development Environment	VS Code / Local Server

Features Implemented

The AI-Based Auto Growth Suggestion System for Financial Modeling includes several functional features designed to automate growth analysis, improve forecasting accuracy, and support financial decision-making. Each feature contributes to making the system modular, scalable, and easy to integrate with existing financial models.

1. API-Driven Growth Suggestion Engine

The system implements a centralized growth suggestion engine exposed through a RESTful API. This engine processes financial inputs and returns standardized growth recommendations in JSON format. By using an API-based approach, the growth logic can be reused across multiple financial models and applications.

2. Automated Financial Growth Analysis

The system automatically evaluates financial growth indicators using predefined analytical logic. This reduces reliance on manual calculations and static assumptions commonly used in traditional spreadsheet-based financial modeling.

3. Standardized Growth Recommendations

Growth suggestions are generated in a consistent and structured format. This ensures uniformity across financial models and enables easy comparison between different growth scenarios and forecasting periods.

4. Swagger-Based API Documentation

The project integrates Swagger (OpenAPI) documentation to provide an interactive interface for testing and validating API endpoints. This improves transparency, usability, and explainability of growth recommendations for analysts and stakeholders.

5. Input Validation and Data Consistency Checks

The system validates all incoming financial inputs for correctness, completeness, and logical consistency. Invalid or missing data is handled gracefully to prevent inaccurate growth recommendations.

6. Modular and Scalable Architecture

The architecture is designed with clear separation between input handling, growth logic, and output delivery. This modular structure allows easy extension of the system with additional financial metrics, advanced logic, or external integrations.

7.Explainable and Transparent Outputs

Growth suggestions are designed to be interpretable and easy to understand. The structured output format allows users to trace recommendations back to the input assumptions and analytical logic used.

8. Error Handling and System Stability

The system includes basic error handling mechanisms to manage invalid inputs, processing issues, or unexpected failures. This ensures stable API behavior and reliable responses.

9. Easy Integration with Financial Models

The JSON-based API responses can be easily consumed by financial models, dashboards, or analytical tools. This makes the system suitable for integration into corporate finance workflows and decision-support systems.

Working Mechanism

The AI-Based Auto Growth Suggestion System for Financial Modeling operates through a structured, step-by-step mechanism that transforms financial inputs into automated growth recommendations using an API-driven approach. The system is designed to ensure accuracy, consistency, and transparency throughout the financial growth analysis process.

Step 1: Financial Data Input

The working process begins with the provision of structured financial data and assumptions. These inputs may include historical growth indicators, baseline assumptions, or predefined financial parameters required for growth analysis.

The system expects inputs in a standardized format to ensure compatibility across different financial models. This step ensures that the growth analysis is initiated with relevant and meaningful data.

Step 2: Input Validation and Consistency Check

Once the financial data is received, the system performs validation checks to ensure data completeness, correct formatting, and logical consistency. Missing, invalid, or contradictory values are identified at this stage.

If inconsistencies are detected, the system either applies predefined default rules or returns a clear error response. This step prevents unreliable data from influencing the growth recommendation process.

Step 3: Growth Analysis Processing

After successful validation, the data is passed to the processing layer where predefined financial growth logic is applied. This logic evaluates growth trends and patterns based on the supplied inputs.

The analysis is designed to simulate how financial analysts assess growth scenarios, ensuring that growth suggestions remain realistic and proportional to the input assumptions.

Step 4: Auto Growth Suggestion Generation

The processed data is then evaluated by the **Auto Growth Suggestion Engine**, which generates structured growth recommendations. These recommendations may include suggested growth rates or strategic guidance relevant to financial modeling and forecasting.

The system focuses on supporting decision-making rather than enforcing rigid outcomes, allowing analysts to interpret and apply suggestions as required.

Step 5: API Response Generation

The generated growth suggestions are returned to the user or consuming system through a RESTful API endpoint built using FastAPI. The response is delivered in a standardized JSON format, ensuring easy integration with financial models, dashboards, or analytical tools.

Swagger-based API documentation enables users to test and validate the API responses interactively.

Step 6: Consumption by Financial Models

The API output is consumed by external financial models or applications. Analysts can incorporate the growth suggestions into budgeting, forecasting, or valuation models to support data-driven decision-making.
This step completes the automated growth suggestion cycle while maintaining flexibility for human judgement.

Step 7: Error Handling and Stability Management

Throughout the working mechanism, error handling routines ensure system stability. Invalid inputs, processing failures, or unexpected conditions are managed gracefully, preventing system breakdowns and ensuring reliable output delivery.

.

Security Considerations

- The system processes structured financial data and does not handle personal or sensitive user information.
- Input validation is implemented to prevent invalid or malformed data from affecting growth analysis.
- The API architecture separates financial logic from client applications, reducing the risk of direct manipulation.
- Error handling mechanisms ensure safe and controlled system responses.
- The current implementation operates in a local development environment, minimizing external exposure.
- Future deployment can include authentication, authorization, and encrypted communication for enhanced security.

Limitations:

- The system uses predefined analytical logic and does not currently employ machine learning techniques.
- Growth recommendations depend on input assumptions and may not reflect sudden market changes.
- Real-time financial data integration is not included in the current implementation.
- The system is designed as a decision-support tool and not as an autonomous financial decision-maker.
- Advanced enterprise-level security features are not implemented in the prototype stage.

Future Enhancements:

- Integration of machine learning models for adaptive and predictive financial forecasting.
- Incorporation of real-time financial and market data sources.
- Development of a dashboard-based interface for visualizing growth trends and insights.
- Implementation of authentication, authorization, and secure API access.
- Extension of the API to support scenario analysis and sensitivity testing.
- Deployment on cloud platforms to improve scalability and availability.

SECTION 2: Main Code Components Used

This section describes the major code components implemented in the AI-Based Auto Growth Suggestion System for Financial Modeling and explains their functionality, purpose, and contribution to the overall system. Each component is designed to ensure modularity, reliability, and consistency in automated financial growth analysis.

2.1 FastAPI Application Initialization

The FastAPI framework is used to create the backend API for the Auto Growth Suggestion System. This component initializes the application and defines its basic configuration.

Code Purpose

- Acts as the entry point of the API
- Handles incoming requests
- Provides automatic Swagger documentation

Code Snippet

```python
from fastapi import FastAPI, HTTPException
import yfinance as yf
import math
import numpy as np

app = FastAPI(title="Auto Growth Suggestion API")

```

2.2 Sector-Based Growth Limits Configuration

This component defines sector-wise minimum and maximum growth limits. These limits ensure that suggested growth values remain realistic and aligned with industry behavior.

Code Purpose

- Applies sector intelligence
- Prevents unrealistic growth suggestions
- Improves financial modeling accuracy

Code Snippet

```python
# ----------------------------
SECTOR_LIMITS = {
    "ENERGY": {"min": 4, "max": 10},
    "IT": {"min": 6, "max": 15},
    "BANKING": {"min": 7, "max": 14},
    "FMCG": {"min": 5, "max": 12},
    "DEFAULT": {"min": 5, "max": 12}
}
```

2.3 Safe Float Handling Module

Financial data fetched from external sources may contain missing, infinite, or invalid values. This component ensures safe numeric conversion.

Code Purpose

- Cleans financial data
- Prevents runtime calculation errors
- Ensures stability of growth calculations

Code Snippet

```python
# ----------------------------
# SAFE FLOAT
# ----------------------------
def safe_float(x):  6 usages
    if x is None:
        return 0.0
    if isinstance(x, float) and (math.isnan(x) or math.isinf(x)):
        return 0.0
    return float(x)
```

2.4 CAGR Calculation Module

This component calculates the Compound Annual Growth Rate (CAGR), a core financial metric used in growth analysis.

Code Purpose

- Calculates long-term growth trend
- Used as a base indicator for growth suggestion
- Handles invalid financial scenarios safely

Code Snippet

```python
# CAGR
# ----------------------------
def calculate_cagr(start, end, years):  1 usage
    start = safe_float(start)
    end = safe_float(end)

    if start <= 0 or end <= 0 or years <= 0:
        return 0.0

    return ((end / start) ** (1 / years) - 1) * 100
```

2.5 Financial Data Fetching Module (Yahoo Finance)

This component fetches historical financial data such as **Revenue, EBITDA, and PAT** using the Yahoo Finance API.

Code Purpose

- Retrieves real financial statements
- Extracts and cleans historical data
- Provides fallback logic when data is missing

Code Snippet

```python
def fetch_financials(ticker: str):  1 usage
    stock = yf.Ticker(ticker)
    fin = stock.financials

    if fin is None or fin.empty:
        raise ValueError("Financial data not available")
```

2.6 Auto Growth Suggestion Engine

This is the **core financial intelligence engine** of the system. It combines long-term CAGR and recent performance to generate growth suggestions.

Code Purpose

- Calculates weighted growth score
- Applies sector-based caps
- Produces realistic growth recommendations

Code Snippet

```
105         raw = 0.6 * cagr + 0.4 * recent_avg
106         limits = SECTOR_LIMITS.get(sector.upper(), SECTOR_LIMITS["DEFAULT"])
107
108         final = min(max(raw, limits["min"]), limits["max"])
109
110         return round(final, 2), round(cagr, 2), round(recent_avg, 2)
111
```

2.7 Growth Suggestion API Endpoint

This API endpoint integrates all components and returns structured growth recommendations for Revenue, EBITDA, and PAT.

Code Purpose

- Accepts company ticker and sector
- Executes growth engine
- Returns JSON-based analysis

Code Snippet

```
114     # API ENDPOINT
115     # ----------------------------
116     @app.get("/suggest_growth")
117     def growth_api(ticker: str, sector: str = "DEFAULT"):
```

2.8 Error Handling Mechanism

The system includes structured error handling to manage missing data, API failures, or invalid inputs.

Code Purpose

- Prevents system crashes
- Returns meaningful error messages
- Improves reliability

Code Snippet

```
140             }
141
142         except Exception as e:
143             raise HTTPException(status_code=400, detail=str(e))
144
145
```

2.9 Home API Health Check

A simple endpoint is implemented to confirm that the API is running successfully.

Code Purpose

- Verifies API availability
- Useful for deployment testing

Code Snippet

```
144
145
146     @app.get("/")
147     def home():
148         return {"status": "API running successfully"}
149
```

Conclusion

The **AI-Based Auto Growth Suggestion System for Financial Modeling** successfully demonstrates how API-driven automation can enhance traditional financial modeling practices. The project addresses key limitations of conventional spreadsheet-based forecasting by introducing a centralized growth suggestion engine capable of generating consistent, explainable, and data-driven growth recommendations.

By integrating real financial data through Yahoo Finance and applying structured financial logic such as CAGR analysis and recent performance trends, the system produces realistic growth suggestions for key financial metrics including Revenue, EBITDA, and Profit After Tax. The incorporation of sector-based growth limits further ensures that recommendations remain aligned with industry behavior, improving the reliability of financial forecasts.

The use of FastAPI enables a modular and scalable system architecture, allowing growth logic to be reused across multiple financial models and applications. Swagger-based API documentation enhances transparency, testability, and ease of integration, making the system suitable for both academic evaluation and practical financial analysis.

Overall, the project highlights the effectiveness of combining financial analytics with modern API design to support informed decision-making. While the current implementation serves as a robust prototype, it provides a strong foundation for future enhancements such as machine learning-based forecasting, real-time data integration, and enterprise-level deployment. The system successfully meets its objectives and demonstrates a practical approach to automated financial growth modeling.

References

- FastAPI Official Documentation – https://fastapi.tiangolo.com/
- Swagger (OpenAPI) Specification – https://swagger.io/
- Yahoo Finance (yfinance) Python Library – https://pypi.org/project/yfinance/
- Python Programming Language – https://www.python.org/
- Investopedia – CAGR (Compound Annual Growth Rate)
- Corporate Finance Institute – Financial Modeling Concepts

Appendices

- **Appendix A:** Technology Stack (Python, FastAPI, yfinance, Swagger)
- **Appendix B:** API Endpoints (/, /suggest_growth)
- **Appendix C:** Financial Metrics Used (Revenue, EBITDA, PAT, CAGR)
- **Appendix D:** Sample API JSON Output
- **Appendix E:** Abbreviations (API, CAGR, EBITDA, PAT)

Note : The system uses a predefined company name–to–ticker mapping for commonly traded Indian companies. This approach simplifies user input, reduces errors, and ensures reliable financial data retrieval. Full market-wide ticker search is proposed as a future enhancement.