



# SYSTEM REQUIREMENTS ANALYSIS FOR PYTHON FRAMEWORKS

PRESENTATION





# 01


# SYSTEM REQUIREMENTS ANALYSIS

Understanding system requirements is crucial for ensuring that software meets user and stakeholder expectations.

- Before choosing a Python framework, clearly identify:
  - **What the system needs to accomplish** (functional requirements).
  - **The conditions under which it must operate** (non-functional requirements).
- This foundational understanding helps in selecting a framework that matches the project's goals and constraints, ensuring an effective and efficient development process.



## 02 COLLECTING DETAILED SYSTEM REQUIREMENTS

- **Functional requirements** outline what the **system should do**, detailing the necessary functions and features such as **user authentication**, **data processing**, **reporting**, and **user interface design**.
  - **Non-functional requirements** describe **how the system should perform**, covering aspects like **response time**, **data security measures**, **scalability**, and **user experience**.
- 

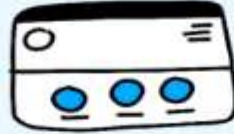


## FUNCTIONAL REQUIREMENTS

examples of

## NON-FUNCTIONAL REQUIREMENTS

the website will have a homepage



the homepage should load within 1.5 seconds

the website will store customer data



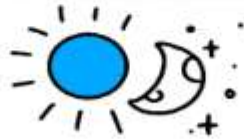
customer data will be encrypted to level 4 encryption standards

website customers can log into their accounts



the website has the capacity for 5.000 logged-in users at any one time

the website will always be available to customers



the website must have a 99.7% uptime

customers can access the websites on their phones



the website will be compatible with iOS 12.8 and above



# REQUIREMENT GATHERING TECHNIQUES

Effective requirement gathering involves **collecting and documenting both functional and non-functional requirements** to ensure the final product meets user and stakeholder needs.

Techniques include:

- **Interviews:** One-on-one or group discussions to gain deep insights.
- **Surveys:** Questionnaires for a broader audience to gather quantitative data.
- **Workshops & Focus Groups:** Collaborative sessions for consensus-building.
- **Prototyping:** Building early versions of the system for feedback.
- **Document Analysis:** Reviewing existing documents for historical context.



# DOCUMENTING REQUIREMENTS

Once requirements are gathered, clear and comprehensive documentation is essential.

Methods include:

- **Requirement Specifications:** Detailed descriptions of functional and non-functional requirements.
- **User Stories:** Simple descriptions of features from an end-user perspective.
- **Use Cases:** Detailed descriptions of user interactions to achieve specific goals.
- **Process Diagrams & Wireframes:** Visual representations of workflows and user interfaces.





# 03 MATCHING SYSTEM REQUIREMENTS TO PYTHON FRAMEWORKS

Choosing the right Python framework is essential for meeting both functional and non-functional system requirements. Frameworks like Django, Flask, and FastAPI offer unique features that cater to different system needs:

- **Django:** Ideal for large-scale, full-stack web applications with built-in security and scalability.
- **Flask:** Lightweight and flexible, suited for microservices and prototyping.
- **FastAPI:** High performance, asynchronous, and great for API-first designs and real-time data processing.