# Title

# Prime Number Generate

# ABSTRACT

This project aims to develop a robust and efficient prime number generator and a server application that allows users to generate prime numbers within a specified range via a REST API. The system supports multiple prime generation strategies, records usage data, and ensures performance and reliability. This documentation outlines the project’s objectives, scope, modules, and detailed descriptions of each component, ensuring clarity for future maintenance and enhancement.

Fig 1. Use Case Diagram **8**

Fig 2: E-R Diagram **9**

Fig 3: API for Register User **11**

Fig 4: API for Login User **11**

Fig 5: API for Log out user **12**

Fig 6: API for Generating Prime Number **12**

# TABLE OF CONTENTS

### Chapter 1: Introduction 4

* 1. Aim of the Project 4
  2. Project Scope 4
  3. Project Objective 4
  4. Project Modules 4
  5. Project basic requirements 5

### Chapter 2: Analysis, Design Methodology, and Implementation Strategy 6

* 1. Detailed Modules Description 6
  2. Project SRS 7
     1. Use Case Diagrams 7
     2. Entity Relation Diagram 8
  3. Data Base Design 10
     1. User Table 10
     2. Execution Table 10

### Chapter 3: Implementation 11

* 1. Snap shot of API 11
  2. How to Run Application 13

### Chapter 4: Conclusion 14

* 1. Conclusion 14

### Chapter 5: References 15

# Chapter 1: Introduction

## Aim of the project

* The primary aim of this project is to create a versatile and high-performance prime number generator that can be accessed via a REST API. It will offer multiple generation strategies, record usage metrics, and ensure robust error and resource management for long-term server operation.

## Project Scope

* Prime number generator with multiple generation strategies.
* Command-line interface for generating primes within a specified range.
* REST API server to provide prime generation services over HTTP.
* In-memory database to log usage data, including timestamps, ranges, time elapsed, chosen algorithm, and number of primes returned.

## Project Objective

Develop a prime number generator supporting different algorithms. Implement a command-line interface for prime generation. Create a REST API server for remote prime generation requests. Record each API request in a database with detailed metrics. Ensure the system is maintainable, efficient, and reliable for long-term use.

## Project Modules

#### Prime number generator module

* + - * Nive approach
      * Optimized approach
      * Sieve of Eratosthenes
      * Optimized of sieve of Eratosthenes
      * Segmented sieve
      * Sieve of Sundaram

#### Rest API server user registration / login module

* + - * Registration
      * Login
      * Logout

#### REST API prime generation Server Module

* + - * API endpoints
      * Choose algorithm, request handling, response formulating

#### Database logging module

* + - * Record execution

#### Error handling

* + - * Exception handling.

## Project basic requirements

#### Software Requirement

* Python
* MongoDB
* Flask

#### Hardware Requirement

* Processor - i5
* Hard Disk – 512GB
* Memory – 4GB RAM

**Chapter 2: Analysis OF Project**

## Detailed Modules Description

## Prime Number Generator Module

## This module includes the implementation of different algorithms to generate prime numbers within a given range. The algorithms include:

## Basic Iterative Method: Checks each number in the range for primality.

## Sieve of Eratosthenes: Efficiently marks non-prime numbers in the range.

## Rest API server user registration / login

* This module is responsible for managing user accounts and enabling new users to sign up for the carpool service. Existing user can log in. Other functionality accessible if user is authenticated.

## REST API server module

* Implements a RESTful API using a web framework (e.g., Flask). It provides endpoints for generating primes and returns results in JSON format.

## Database logging module

* Uses an in-memory database to log each API request's details, such as timestamp, range, algorithm used, time taken, and number of primes returned.

## Error handling

* Manages exceptions and ensures proper resource cleanup to maintain server stability.

## Project SRS

### Use Case Diagrams

* + - * Use case diagrams are a popular method for communicating the main functions of a software system. At its most basic, a use case diagram is a depiction of a user's engagement with the system that depicts the relationship between the user and the many use cases in which the user is involved. A use case diagram may identify distinct sorts of system users and use cases, and it is frequently supplemented by other types of diagrams.
      * Use cases are simply the system's functionality written in an organized fashion. The actors are another factor that is crucial to the use cases. Actors are any objects that interact with the system.

- In summary, use case diagrams serve the following purposes:

* Used to collect system requirements.
* Used to acquire a bird's-eye perspective of a system.
* Determine the external and internal elements that influence the system.
* Display the criteria interacting as actors.

### Symbols used in use-case diagram:

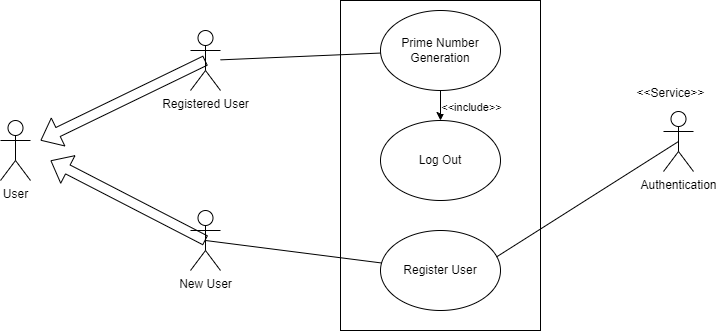
Use Case Include

Association Extend

Actor Dependency

System Generalization

**Use – Case Diagram for Prime number generate System**



**Figure 1: Use Case Diagram**

### Entity Relationship Diagrams

### Entity-Relationship model is used to represent a logical design of a database to be created. In ER model, real world objects (or concepts) are abstracted as entities, and different possible associa8ons among them are modeled as relationships. We represents the attributes, entities and rela8on using the ER diagram. Using this ER diagram, table structures are created, along with required constraints. Finally, these tables are normalized in order to remove redundancy and maintain data integrity. Thus, to have data stored efficiently, the ER diagram is to be drawn as much detailed and accurate as possible.

### Symbols used in ER diagram:

### 

**Figure 2: E-R Diagram**

## Database Design and Normalization

#### User Table:

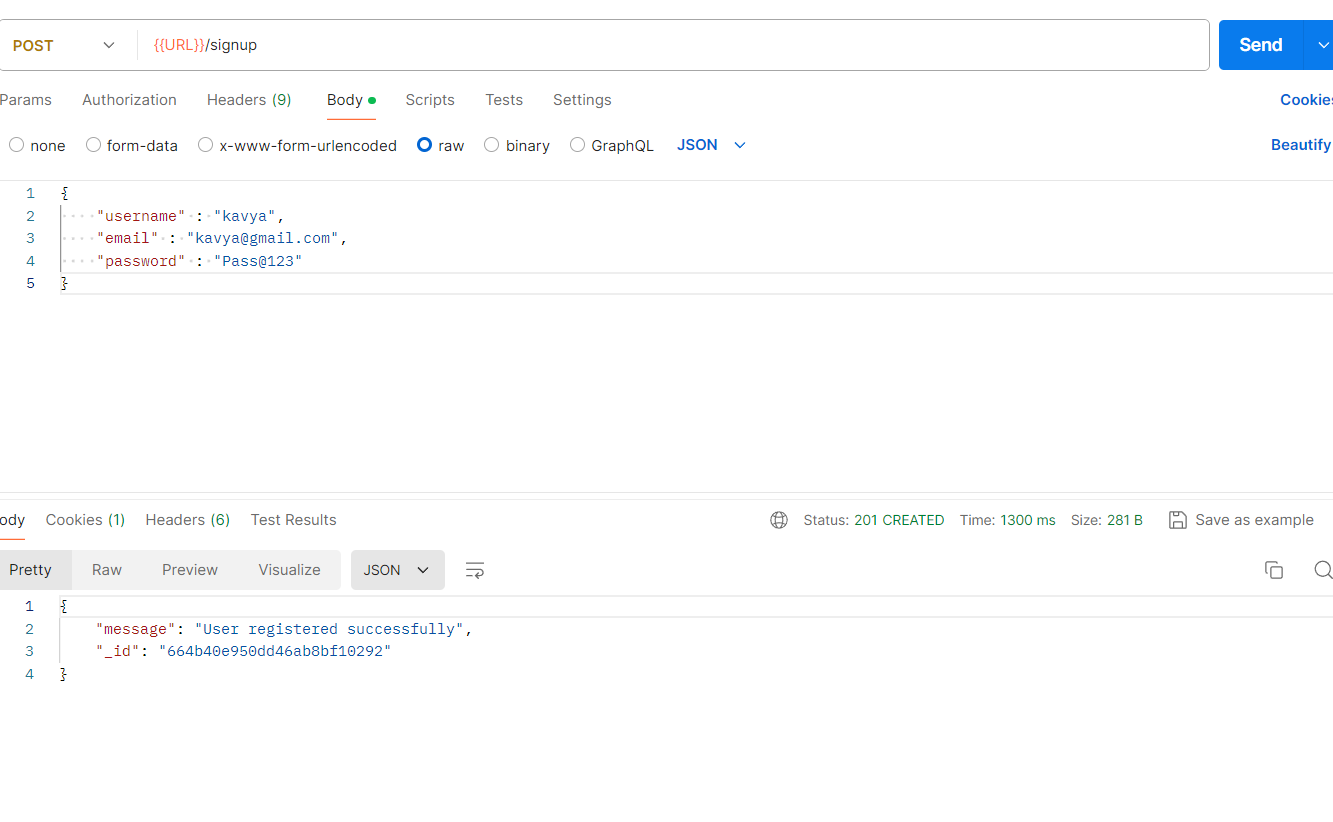
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sr.**  **No.** | **Name** | **Datatype** | **Constraint** | **Description** |
| 1. | User\_id | int(10) | Primary key | To store the user id |
| 2. | Username | varchar(10) | Not null | To store the username |
| 3. | Password | varchar(10) | Not null | To store the password |
| 4. | Email | varchar(20) | Not null | To store email |

#### Execution Table:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sr.**  **No.** | **Name** | **Datatype** | **Constraint** | **Description** |
| 1. | Data\_id | int(10) | Primary key | To store the user id |
| 2. | Email | Varchar(20) | Not null | To store email |
| 3. | Start\_range | varchar(10) | Not null | To store the start range |
| 4. | End\_range | varchar(10) | Not null | To store the end range |
| 5. | Algorithm\_choosen | int(10) | Not null | To store algorithm choosen |
| 6. | Time\_elapsed | DateTime | Not null | To store time need for execution of algorithm |
| 7. | Prime\_returns | String | Not null | Prime number returns by algorithm |

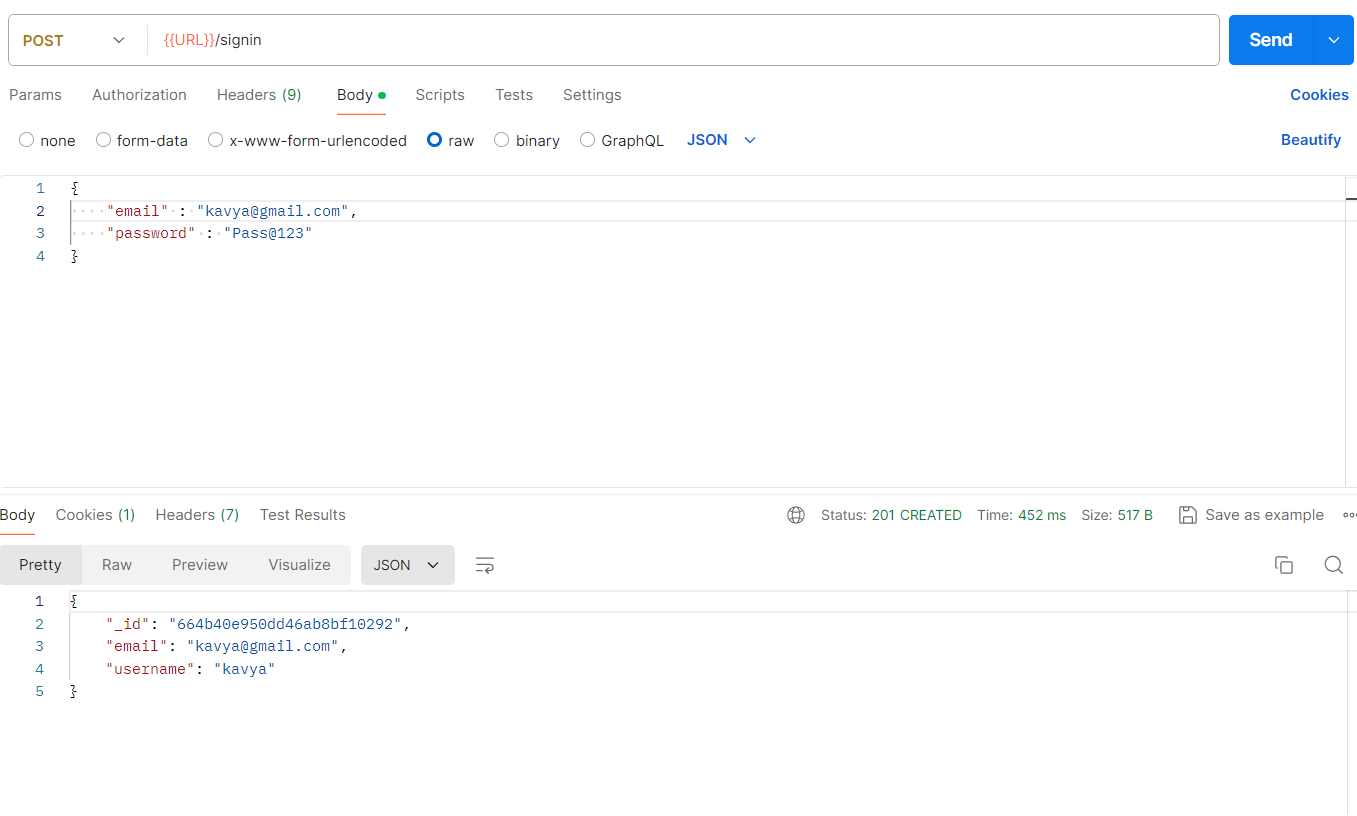
## Chapter 3: Implementation

1. **Snap Shot of API Run:**
   * 1. **Register User**



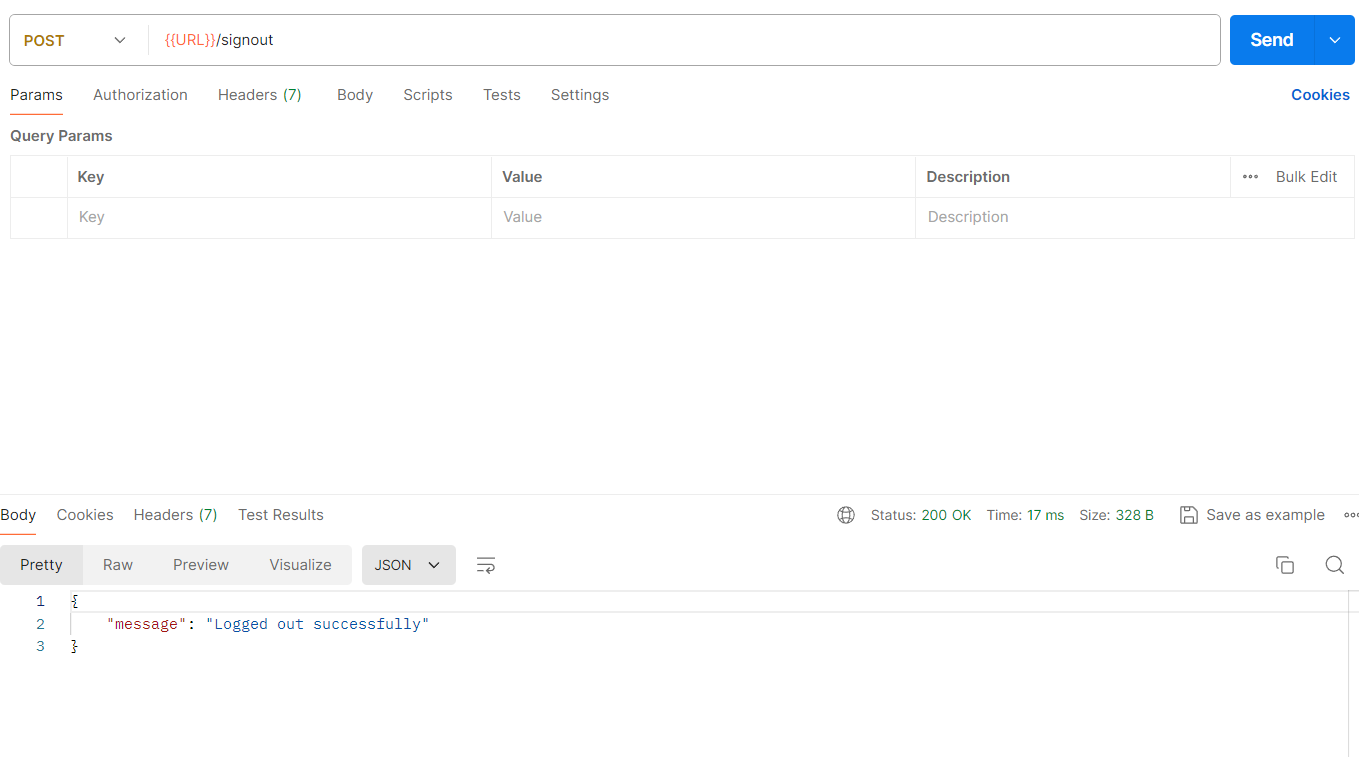
**Fig 3: API for Register User**

* + 1. **Login User**

****

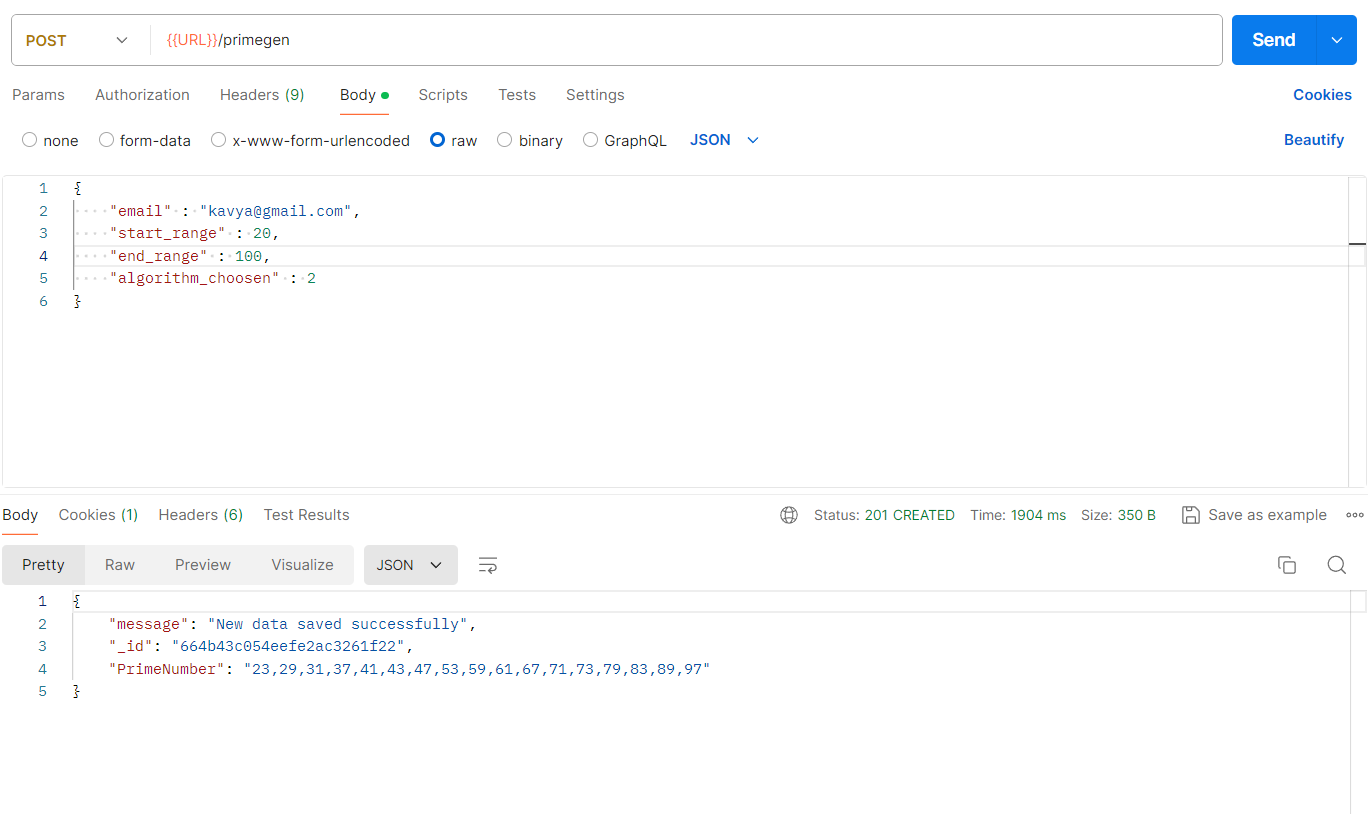
**Fig 4 API for Login User**

* + 1. **Log out User**

****

**Fig 5 API for Log out user**

* + 1. **Prime Number generator**

****

**Fig 6 API for Generating Prime Number**

1. **How To Run Application:**

* **Run Application:**
  + cd server
  + python -m venv env
  + env\Scripts\activate
  + pip install -r requirements.txt
  + python app.py

## Chapter 4: Conclusion

* 1. **Conclusion:**

This project delivers a comprehensive solution for generating prime numbers using multiple algorithms, accessible via both a command-line interface and a REST API. It includes detailed logging and robust error management, making it suitable for long-term, reliable use. The documentation and modular design ensure the system is maintainable and extensible for future enhancements.

## Chapter 5: References

* <https://www.baeldung.com/cs/prime-number-algorithms>
* <https://www.geeksforgeeks.org/prime-numbers/>
* <https://en.wikipedia.org/wiki/Sieve_of_Sundaram>
* <https://en.wikipedia.org/wiki/Sieve_of_Eratosthenes>
* <https://www.geeksforgeeks.org/segmented-sieve-print-primes-in-a-range/>
* <https://flask.palletsprojects.com/en/3.0.x/>