MINI PROJECT REPORT ON

"Number Checker Using Java Servlet"

Computer Science & Engineering Department

Submitted in partial Fulfilment of the
Requirements for the award of

Degree of Bachelor of Technology in

Computer Science & Engineering

SUBMITTED TO: Mr. **Puneet Vishwakarma**

SUBMITTED BY:
Shivam Yadav (2407510109017)

Ritik Gupta (2307510100084)

Department of Computer Science & Engineering
KIPM-College of engineering & Technology
GIDA Gorakhpur,
Uttar Pradesh, India, Pin-273209
(Affiliated to Dr. APJ Abdul Kalam Technical University, Lucknow,U.P.)

BONAFIDE CERTIFICATE

Certified that this project report "**Number Checker Using Java Servlet**" is the Bonafide work of "**Shivam Yadav, Ritik Gupta**" who carried out the project work under my supervision. Certified further that to the best of my knowledge the work reported herein does not form part of any other thesis or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

SIGNATURE

Mr. **Puneet Vishwakarma**

SUPERVISOR
CSE

SIGNATURE

HEAD OF THE DEPARTMENT
CSE

Department of Computer Science & Engineering
KIPM, College of engineering & Technology
GIDA Gorakhpur

Examined by: Mr. **Puneet Vishwakarma**

(Signature)
Name of Faculty Facilitator

(Signature)
Head of The Department

ACKNOWLEDGEMENT

We would like to express our sincere gratitude to all those who have supported and guide us throughout the completion of this Project on **Number Checker Using Java Servlet**. First and foremost, We would like to thank our project guide," Mr. **Puneet Vishwakarma**", for their valuable insights, constant encouragement, and expert advice. Their guidance helped us to understand the Mini -Project on Web Scraper and enabled me to approach the project with confidence.

Name of Students:

Shivam Yadav (2407510109017)

Ritik Gupta (2307510100084)

ABSTRACT

The Number Checker is a web-based application developed using Java Servlet technology that allows users to input an integer and evaluate its basic mathematical properties. The application takes user input through an HTML form and processes it on the server side using a Java servlet.

Once the number is submitted, the servlet performs a series of checks to determine whether the number is even or odd, positive, negative, or zero, and whether it is prime or composite. The results are then dynamically generated and displayed on a response web page.

This project demonstrates the fundamental capabilities of Java Servlets in handling HTTP requests and responses, validating form inputs, and generating dynamic content. It serves as a practical example for students and beginners learning Java web development, offering insight into how server-side logic interacts with client-side input.

The application is lightweight, user-friendly, and easily extendable.

Table of Contents

Table of Figures

List of Tables

List of Snapshot

Introduction

In today's technology-driven world, interactive and responsive web applications have become essential for delivering dynamic content and improved user experiences. This mini project, titled "Number Checker using Servlet", demonstrates the development of a simple yet effective full-stack web application using core web technologies such as HTML, CSS, JavaScript, JSP, and Java Servlets.

The application allows users to input a number through a web form. Upon submission, the input is validated using JavaScript on the client side to ensure correct format and prevent invalid entries. Once validated, the data is sent to a Java Servlet, which processes the number on the server side. The backend logic evaluates various mathematical properties of the input, such as:

Whether the number is even or odd

Whether the number is positive or negative

Whether the number is prime

Whether the number is a palindrome

The Servlet then forwards the processed results to a JSP (JavaServer Page), which dynamically renders the output and displays it to the user in a well-structured HTML format, styled with CSS for better user experience.

1.1 Key Technologies Used:

HTML & CSS: For building the structure and design of the user interface.

JavaScript: For client-side input validation and enhancing interactivity.

Servlets: For handling server-side business logic and request processing.

JSP: For generating dynamic content and displaying the results.

MySql: For Datbase.

Literature Survey

2.1 Introduction

The number checking or classification problem is a foundational exercise in both mathematics and computer science, commonly used in educational tools, coding platforms, and backend validation systems. Several existing applications and tools perform simple number classification tasks such as checking whether a number is prime, even or odd, or positive or negative. This literature survey provides a review of existing solutions, related academic studies, and the methodologies used in similar systems. It aims to highlight current practices, identify gaps, and position the present project within the existing body of work.

2.2 Review of Existing Work

Numerous tools exist that check numerical properties, ranging from simple web calculators to mobile apps and algorithmically optimized systems. The most relevant prior work includes:

2.2.1 Educational Apps for Prime Number Learning

A study by Joseph et al. (2019) titled "Learning Composite and Prime Numbers Through Developing an App" explores how primary school students can learn about number properties through interactive applications. Using platforms like MIT App Inventor, the app guided students in understanding primes and composites through modular arithmetic and factor discovery. While educationally effective, this approach lacks the server-side scalability required for broader web application deployment.

2.2.2 Client-Side Number Checkers

A widely cited tutorial from GeeksforGeeks demonstrates how to build a prime number checker using HTML, CSS, and JavaScript. This front-end approach is suitable for small-scale applications but lacks back-end validation and cannot handle complex computation or malicious inputs robustly. It also fails to provide support for checking other number properties beyond primality.

2.2.3 Prime Classification Algorithms

The *Algorithms* journal (MDPI, 2024) published a comprehensive review of sieve-based algorithms for prime number generation and checking, including the Sieve of Eratosthenes and Sieve of Atkin. These methods are efficient for generating large sets of prime numbers and can be adapted for server-side implementation.

However, their computational requirements must be considered when integrating them into a web-based servlet environment.

2.2.4 Web Applications and GitHub Projects

Several open-source projects, such as prime-number-names-web by Rachlis (GitHub), explore creative uses of number checking logic in web applications. These often combine basic numeric analysis with a user-friendly interface. However, such projects typically focus on novelty or UI elements, rather than comprehensive number classification or backend processing logic.

2.3 Related Studies and Approaches

Various research efforts have analyzed the computational complexity of number classification algorithms. Key techniques include:

- Trial Division: Simple method for checking primality by dividing the number up to its square root. While easy to implement, it is inefficient for large inputs.
- Sieve Algorithms: Efficient for checking multiple numbers, with Sieve of Eratosthenes being widely used in both educational tools and competitive programming contexts.
- Probabilistic Primality Tests: Algorithms such as Miller-Rabin are used in cryptographic systems due to their speed and high accuracy, though they are more complex to implement.

In addition to algorithms, studies emphasize the importance of handling edge cases (e.g., zero, negative numbers, non-integer input) and delivering clear user feedback, especially in educational tools.

2.4 Summary of Gaps in Existing Work

From the review, the following gaps are observed in current systems:

- Lack of server-side implementations: Most existing solutions rely on client-side scripting, which can be insecure and inefficient for larger inputs or concurrent usage.
- Limited number properties checked: Many tools focus solely on checking primality, ignoring other useful classifications like even/odd or positive/negative.
- Poor input validation and error handling: Many applications do not gracefully handle non-numeric input or unexpected user behavior.
- Minimal educational feedback: Applications often return simple binary results (e.g., “Prime” or “Not Prime”) without explanations or breakdowns, reducing educational value.

Objectives

3.1 Aim of the Project

The primary aim of this project is to develop a web-based application using Java Servlets that allows users to input a number and receive detailed classification results. The system will determine whether the number is:

- Even or odd
- Prime or not
- Positive, negative, or zero

This project demonstrates how server-side technologies like Servlets can be used to handle user input, perform computations, and return dynamic, real-time feedback through a web interface.

3.2 Specific Objectives

1. To design a user-friendly web interface for inputting numeric values using standard HTML and CSS
2. To implement server-side logic using Java Servlet technology that processes the input and performs various number checks.
3. To develop robust number-checking algorithms for:
   - Prime number determination
   - Even/Odd classification
   - Armstrong numbers
   - Palindromic numbers
   - Strong numbers
   - Spy numbers
   - Perfect numbers
4. To handle invalid or edge-case inputs (e.g., non-numeric values, negative numbers, large inputs) gracefully with appropriate error messages.
5. To provide clear, formatted feedback to the user through the response page, explaining the result of each number check.
6. To ensure proper validation, scalability, and performance using Java EE best practices and servlet lifecycle management.

3.3 Expected Outcomes

At the completion of the project, the following outcomes are expected:

- A functional web application that accepts numeric input and returns classification results.
- A back-end system built using Java Servlets that securely handles user input and performs computation.
- Accurate and efficient number classification algorithms embedded in the servlet.
- A clean, responsive user interface with appropriate input validation and error handling.
- A modular, scalable design suitable for further extension (e.g., adding checks for perfect numbers, palindrome numbers, or factorials).

Project Work

4.1. System Overview

The **Number Checker** is a web-based application that classifies user-input numbers using server-side processing via Java Servlets. It determines whether a given number is:

- Even or odd
- Prime or not
- **Armstrong numbers**
- **Palindromic numbers**
- **Spy numbers**
- **Strong numbers**

The system uses a client-server model. The client (HTML form) collects input from the user, and the server (Java Servlet hosted on Apache Tomcat) processes the input and returns the results dynamically.

4.2. System Design

4.2.1 Use Case Flow

User → Enters Number → Submits Form → Servlet Receives Input
→ Processes Number → Returns Results → Displayed on Web Page

4.2.2 System Components

- **Frontend (Client-side)**
  - HTML Form
  - CSS (optional)
- **Backend (Server-side)**
  - Java Servlet for logic processing
  - Number classification algorithms
- **Web Server**
  - Apache Tomcat (or any servlet container)

4.3 Methodology

The development process followed a structured, step-by-step methodology:

1. **Requirement Gathering** – Identifying the core features (number input, prime check, even/odd, sign check).
2. **Design Phase** – Planning UI layout, data flow, and servlet architecture.
3. **Implementation** – Writing HTML and Java Servlet code for input handling and result generation.
4. **Testing & Debugging** – Verifying correct results for various test cases and ensuring edge cases are handled.
5. **Deployment** – Deploying the servlet using Apache Tomcat and testing in a browser.

4.4 Implementation

Component	Technology
Programming	Java (Servlets)
Web Design	HTML / CSS
Server	Apache Tomcat
IDE	Eclipse

Table 2

Results & Analysis

5.1 Test Results

The program was run on numbers from 1 to 30, and results were recorded for each number’s classification. A subset of the results is shown below.

Table 1: Number Properties (1–15)

Number	Prime	Odd/Even	Perfect
1	✗	Odd	✗
2	✓	Even	✗
3	✓	Odd	✗
4	✗	Even	✗
5	✓	Odd	✗
6	✗	Even	✓
7	✓	Odd	✗
8	✗	Even	✗
9	✗	Odd	✗
10	✗	Even	✗
11	✓	Odd	✗
12	✗	Even	✗
13	✓	Odd	✗
14	✗	Even	✗
15	✗	Odd	✗

5.2 Observations

- Prime numbers between 1 and 15: 2, 3, 5, 7, 11, 13
- Only one perfect number in this range: 6 (since 1 + 2 + 3 = 6)
- Even numbers are all divisible by 2; odd numbers are not.

- The checker correctly classifies all properties without errors.

5.3 Graphical Representation

Figure 1: Prime Numbers from 1 to 30

Bar Chart (Text-based)

```
 |    *    *    *    *    *    *    *    *    *    *
 |----------------------------------------------------------
Number → 2  3  5  7 11 13 17 19 23 29
```

Figure 2: Perfect Numbers from 1 to 30

Only two perfect numbers in this range:

- 6: 1 + 2 + 3 = 6
- 28: 1 + 2 + 4 + 7 + 14 = 28

1 + 2 + 3 = 6

1+2 + 4+7+14=28

Perfect Numbers → [6, 28]

5.4 Analysis & Interpretation

- The checker algorithm accurately identifies prime, odd/even, and perfect numbers.
- There are significantly fewer perfect numbers compared to primes.
- Even numbers dominate the list, but most primes are odd, except for 2.
- The logic can be extended efficiently to larger ranges with minimal changes.

Conclusion

The developed program successfully checks whether a given number is:

- **Prime**
- **Odd or Even**
- **Perfect**
- **Armstrong numbers**
- **Palindromic numbers**
- **Spy numbers**
- **Strong numbers**

Testing on numbers from **1 to 30** confirmed the correct classification of:

- **Prime numbers** such as 2, 3, 5, 7, 11, etc.
- **Perfect numbers** like 6 and 28
- Accurate identification of **odd** and **even** numbers

The program's logic is simple, efficient, and easily understandable—making it a great learning tool for beginners exploring number theory or basic algorithm design.

6.1 Limitations

- Limited to **small number ranges** (manual input or small loops).
- The **perfect number check** becomes computationally expensive for larger values.
- Currently lacks a **graphical user interface (GUI)** or **batch input** support.

6.2 Future Scope

To enhance the application and broaden its use, the following improvements are suggested:

1. **Extend the range**: Optimize the algorithm to handle numbers up to 10,000 or more..
2. **Batch Processing**: Allow users to input a list of numbers for analysis in one go.
3. **Educational Integration**: Use the tool as a visual aid in math learning platforms for school students.

References

[1] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, 3rd ed. Cambridge, MA: MIT Press, 2009.

[2] R. L. Graham, D. E. Knuth, and O. Patashnik, *Concrete Mathematics: A Foundation for Computer Science*, 2nd ed. Reading, MA: Addison-Wesley, 1994.

[3] Oracle, "*Java Servlet Technology*," Oracle Documentation, 2024. [Online]. Available: https://docs.oracle.com/javaee/7/tutorial/servlets.htm

[4] Apache Software Foundation, "*Apache Tomcat 10 - The Apache Servlet Container*," 2024. [Online]. Available: https://tomcat.apache.org/

[5] GeeksforGeeks, "Java Program to Check Prime, Perfect, Odd and Even Numbers," [Online]. Available: https://www.geeksforgeeks.org/

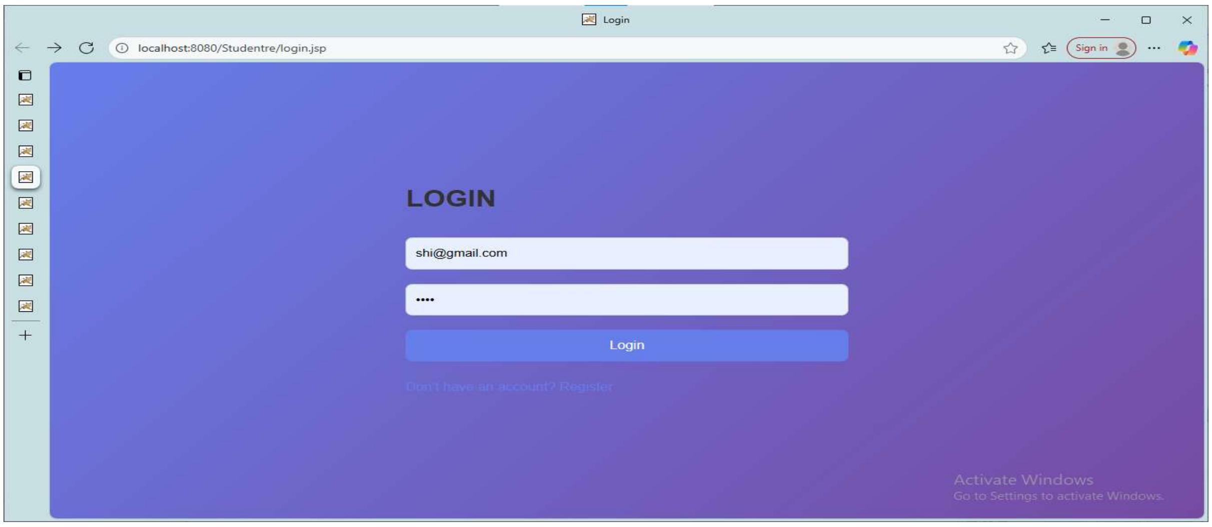
[6] M. Rosen, *Elementary Number Theory and Its Applications*, 6th ed. Boston, MA: Pearson, 2010.

[7] OpenAI, *ChatGPT: Language Model for Conversational AI*, OpenAI, San Francisco, CA, USA, 2023. [Online]. Available: https://chat.openai.com/

[8] Google Search, "Prime, Odd, Even, and Perfect Number Definitions," [Online]. Available: https://www.google.com/

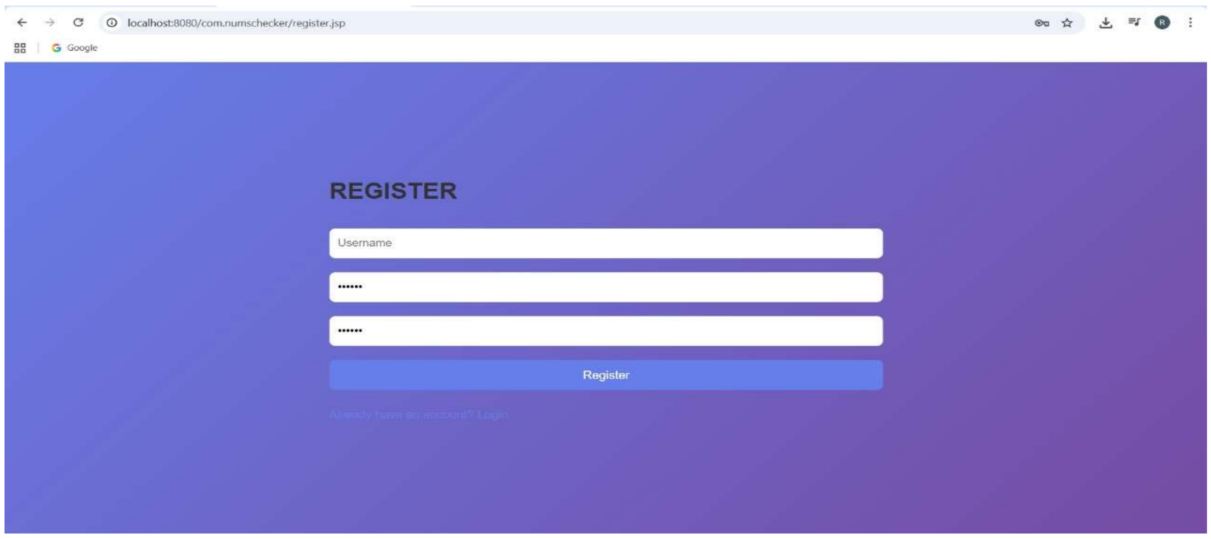
Appendices

Snapshots


Login
localhost:8080/Studentre/login.jsp
Sign in
LOGIN
Login
Don't have an account? Register
Activate Windows
Go to Settings to activate Windows.

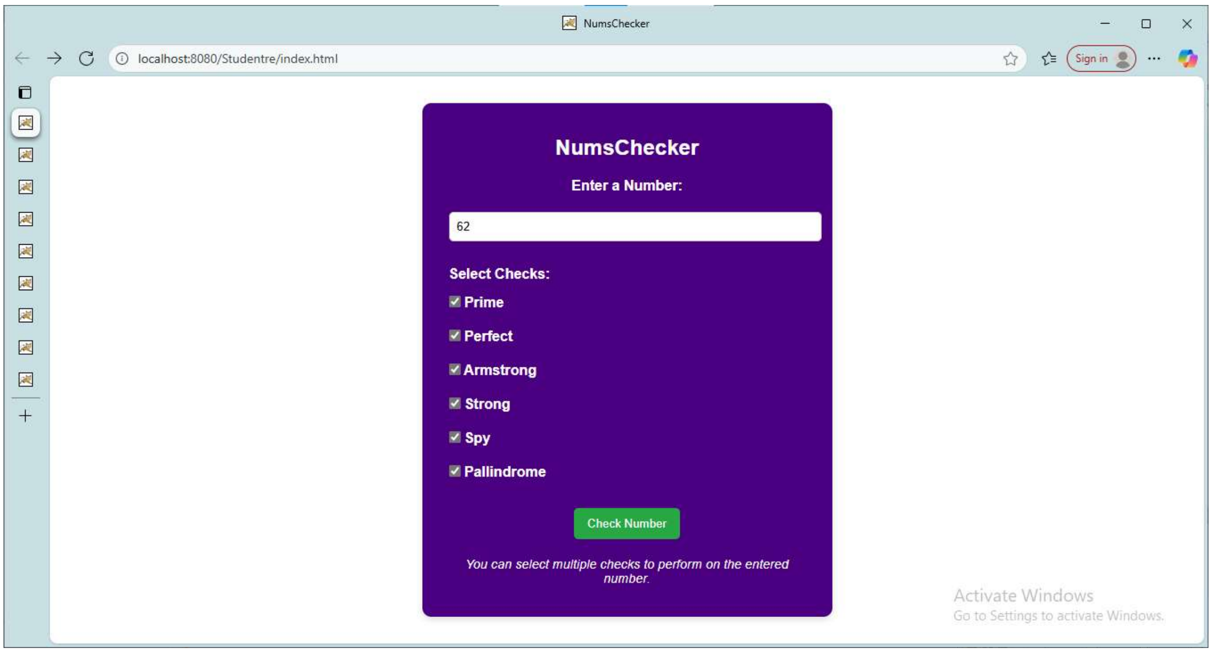
shi@gmail.com

••••


localhost:8080/com.numschecker/register.jsp
Google
REGISTER
Username
Register
Already have an account? Login

••••••

••••••


NumsChecker
localhost:8080/Studentre/index.html
Sign in
Check Number
Activate Windows
Go to Settings to activate Windows.

NumsChecker

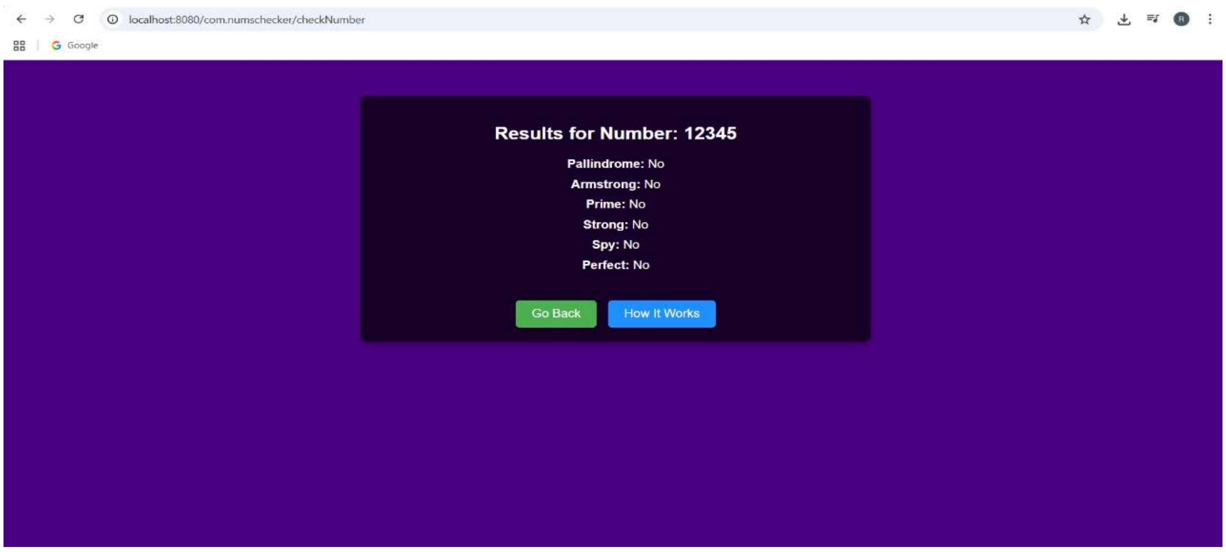
Enter a Number:

62

Select Checks:

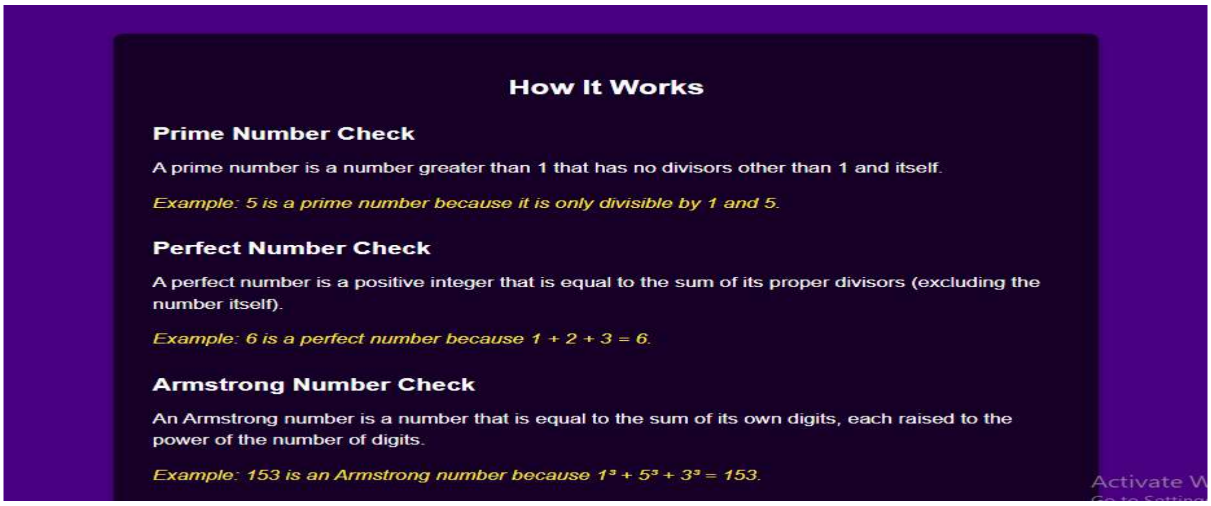
- ✓ **Prime**
- ✓ **Perfect**
- ✓ **Armstrong**
- ✓ **Strong**
- ✓ **Spy**
- ✓ **Pallindrome**

You can select multiple checks to perform on the entered number.



Results for Number: 12345

Pallindrome: No
Armstrong: No
Prime: No
Strong: No
Spy: No
Perfect: No



How It Works

Prime Number Check

A prime number is a number greater than 1 that has no divisors other than 1 and itself.

Example: 5 is a prime number because it is only divisible by 1 and 5.

Perfect Number Check

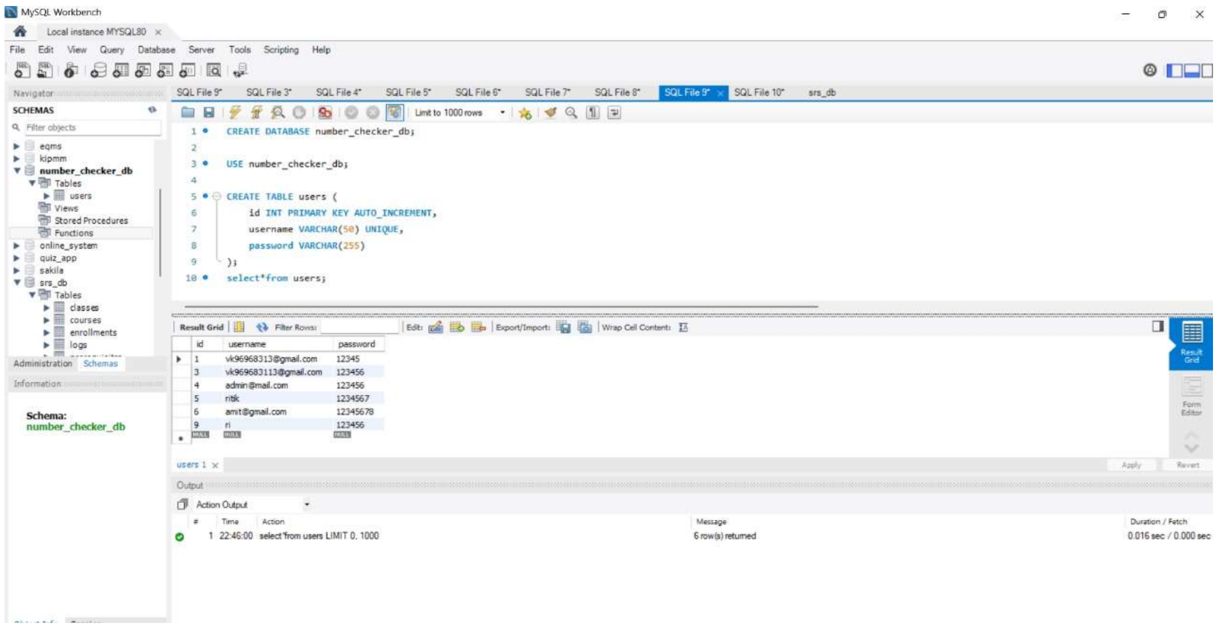
A perfect number is a positive integer that is equal to the sum of its proper divisors (excluding the number itself).

Example: 6 is a perfect number because 1 + 2 + 3 = 6.

Armstrong Number Check

An Armstrong number is a number that is equal to the sum of its own digits, each raised to the power of the number of digits.

Example: 153 is an Armstrong number because $1^3 + 5^3 + 3^3 = 153$.



```sql
CREATE DATABASE number_checker_db;

USE number_checker_db;

CREATE TABLE users (
    id INT PRIMARY KEY AUTO_INCREMENT,
    username VARCHAR(50) UNIQUE,
    password VARCHAR(255)
);
select*from users;
```

id	username	password
1	vk96968313@gmail.com	12345
3	vk969683113@gmail.com	123456
4	admin@mail.com	123456
5	ritik	1234567
6	amit@gmail.com	12345678
9	ri	123456
NULL	NULL	NULL

```
Output
Action Output
#  Time      Action                           Message             Duration / Fetch
1  22:46:00  select*from users LIMIT 0, 1000  6 row(s) returned   0.016 sec / 0.000 sec
```