

Spam Detection System – API Usage Guide

1. Prerequisites

Before running the project, ensure you have the following installed:

- **Java 17**
- **Maven 3.6+**
- **MySQL**

Database Setup

1. Open MySQL and create a new schema using the following command:
`CREATE SCHEMA spam_detection;`
- 2.
3. Update your `application.properties` file with your MySQL credentials:
`spring.datasource.url=jdbc:mysql://localhost:3306/spam_detection`
4. `spring.datasource.username=<your-username>`
5. `spring.datasource.password=<your-password>`
- 6.

2. Running the Application

1. **Download the project:** Since you are not using GitHub, extract the project files in your IDE.
2. **Navigate to the project directory.**
3. **Build and run the application:**
`mvn clean install`
4. `mvn spring-boot:run`
- 5.
6. **Access the application:** The API will be available at:
`http://localhost:8080/`
- 7.
8. **Database Initialization:** The database will be automatically populated using `data.sql`.

3. API Endpoints & Usage

User Registration

Endpoint: POST /api/register

Description: Registers a new user.

Request Body (JSON):

```
{
  "name": "Prabhat Kumar",
  "phoneNumber": "9876543210",
  "email": "prabhat@example.com",
  "password": "Prabhat12@"
}
```

Authentication:  Not Required

User Login

Endpoint: POST /auth/login

Description: Authenticates a user and returns a JWT token.

Request Body (JSON):

```
{
  "phoneNumber": "9876543210",
  "password": "securepassword"
}
```

Authentication:  Not Required

Response: JWT Token (to be used in Authorization header)

Update User Profile

Endpoint: PUT /api/update

Description: Updates the user's name or email.

Request Header:

Authorization: Bearer <JWT-TOKEN>

Request Body (JSON):

```
{
  "name": "New Name",
  "email": "newemail@example.com"
}
```

Authentication:  Required

Search User by Name

Endpoint: GET /search/searchByName

Description: Searches users by name.

Request Example:

GET /search/searchByName?name=Prabhat Kumar

Authentication:  Required

Search User by Phone Number

Endpoint: GET /search/searchByNumber

Description: Searches for a phone number in the global database.

Request Example:

GET /search/searchByNumber?phone_number=9876543210

Authentication:  Required

View User Profile

Endpoint: GET /search/displayProfile

Description: Displays the full profile of a user.

Request Example:

GET /search/displayProfile?phone_number=9876543210

Authentication:  Required

Report a Number as Spam

Endpoint: POST /spam/report

Description: Marks a number as spam.

Request Example:

POST /spam/report?phone_number=9876543210

Authentication:  Required

4. Authentication

For all endpoints that require authentication, include the JWT Token in the request header:

Authorization: Bearer <JWT-TOKEN>

```
mysql> show tables;
```

```
+-----+
| Tables_in_callerid |
+-----+
| contacts           |
| spam_reports       |
| users              |
+-----+
```

```
prabhatkumar — mysql -u root -p — 204x51

[mysql> select * from users;

+----+-----+-----+-----+-----+
| id | email                | name    | password                                                                 | phone_number |
+----+-----+-----+-----+-----+
| 51 | johndoe@example.com  | John    | $2a$12$0asnJBYPJV4RVLgbFAsU6uiyz1UCfzPvkJqJiQ0bzMxiLRVwF7Xu2      | 1234567890   |
| 52 | janesmith@example.com | Jane    | $2a$12$RM6xOwk1wS11ch3RgbE1PeisZeDouay65ACOnCdVgs0uxA4zNwvwi       | 9876543210   |
| 53 | alice@example.com    | Alice   | $2a$12$xGNMmhYhnxakbdYvac1Ji02Zu0F0sz4I.100/a.N4c35raeSK9T0e       | 1112223333   |
| 54 | bob@example.com      | Bob     | $2a$12$Qm1mf800CG3VESSene0YOpunUxIYRwfoQG6lw5ICn34enivy.8oa       | 4445556666   |
| 55 | charlie@example.com  | Charlie | $2a$12$N76.Ie1ClxCXxqZCYIfba.gNJWmWZ/GbuMfmk1IzB1L0Q4dp3exxi      | 7778889999   |
| 56 | david@example.com    | David   | $2a$12$YXDBALjZ.ho04v1Rqops8.tXPW1ynF/OC0H2N/re7xGPHscsb0bbu      | 8889990000   |
| 57 | ava@example.com      | Ava     | $2a$12$0asnJBYPJV4RVLgbFAsU6uiyz1UCfzPvkJqJiQ0bzMxiLRVwF7Xu2      | 1234567812   |
| 58 | raj@example.com      | Raj     | $2a$12$RM6xOwk1wS11ch3RgbE1PeisZeDouay65ACOnCdVgs0uxA4zNwvwi       | 9876543238   |
| 59 | maya@example.com     | Maya    | $2a$12$xGNMmhYhnxakbdYvac1Ji02Zu0F0sz4I.100/a.N4c35raeSK9T0e       | 1112223377   |
| 60 | john@example.com     | John    | $2a$12$Qm1mf800CG3VESSene0YOpunUxIYRwfoQG6lw5ICn34enivy.8oa       | 4445556611   |
| 61 | sophia@example.com  | Sophia  | $2a$12$N76.Ie1ClxCXxqZCYIfba.gNJWmWZ/GbuMfmk1IzB1L0Q4dp3exxi      | 7778882999   |
| 62 | manoj@example.com   | Manoj   | $2a$12$YXDBALjZ.ho04v1Rqops8.tXPW1ynF/OC0H2N/re7xGPHscsb0bbu      | 8889996005   |
| 75 | prabhat12@example.com | Prabhat Kumar | $2a$10$hCQCmWfuou.ZfMSemm/Oa.2tpuAOMKYVv364qEJJzDM8rosa18Mje    | 5556667777   |
+----+-----+-----+-----+-----+

13 rows in set (0.01 sec)

[mysql> select * from spam_reports;

+----+-----+
| id | phone_number |
+----+-----+
| 1  | 1234567890   |
| 2  | 9876543210   |
| 3  | 3332221111   |
| 4  | 5554443333   |
| 5  | 7778889999   |
| 6  | 7778889549   |
| 7  | 1234567890   |
| 8  | 9876543210   |
| 9  | 3332221111   |
| 10 | 5554443333   |
| 11 | 7778889999   |
| 12 | 1234567890   |
| 13 | 9876543210   |
| 14 | 3332221111   |
| 15 | 5554443333   |
| 16 | 7778889999   |
| 17 | 7778889549   |
| 18 | 1234567890   |
| 19 | 9876543210   |
| 20 | 3332221111   |
| 21 | 5554443333   |
| 22 | 7778889999   |
| 23 | 5556667777   |
+----+-----+

23 rows in set (0.01 sec)
```