Development Process

Step-By-Step

Development Process

1. Run SQL Script that contains encrypted passwords

Development Process

Step-By-Step

1. Run SQL Script that contains encrypted passwords

   A. Modify DDL for password field, length should be 68

Development Process

Step-By-Step

1. Run SQL Script that contains encrypted passwords

   A. Modify DDL for password field, length should be 68

2. Modify database properties file to point to new database schema

Development Process

Step-By-Step

1. Run SQL Script that contains encrypted passwords

   A. Modify DDL for password field, length should be 68

2. Modify database properties file to point to new database schema

THAT'S IT … no need to change Java source code :-)

Spring Security Password Storage

- In Spring Security 5, passwords are stored using a specific format

```
{id}encodedPassword
```

Spring Security Password Storage

- In Spring Security 5, passwords are stored using a specific format

```
{id}encodedPassword
```

ID	Description

Spring Security Password Storage

- In Spring Security 5, passwords are stored using a specific format

```
{id}encodedPassword
```

ID	Description
noop	Plain text passwords

Spring Security Password Storage

- In Spring Security 5, passwords are stored using a specific format

```
{id}encodedPassword
```

ID	Description
noop	Plain text passwords
bcrypt	BCrypt password hashing

Spring Security Password Storage

- In Spring Security 5, passwords are stored using a specific format

```
{id}encodedPassword
```

ID	Description
noop	Plain text passwords
bcrypt	BCrypt password hashing
…	…

Spring Security Password Storage

- In Spring Security 5, passwords are stored using a specific format

```
{bcrypt}encodedPassword
```

username	password	enabled
john	{bcrypt}$2a$04$eFytJDGtjbThXa80FyOOBuFdK2IwjyWefYkMpiBEFlpBwDH.5PM0K	1
mary	{bcrypt}$2a$04$eFytJDGtjbThXa80FyOOBuFdK2IwjyWefYkMpiBEFlpBwDH.5PM0K	1
susan	{bcrypt}$2a$04$eFytJDGtjbThXa80FyOOBuFdK2IwjyWefYkMpiBEFlpBwDH.5PM0K	1

Spring Security Password Storage

- In Spring Security 5, passwords are stored using a specific format

```
{bcrypt}encodedPassword
```

username	password	enabled
john	{bcrypt}$2a$04$eFytJDGtjbThXa80FyOOBuFdK2IwjyWefYkMpiBEFlpBwDH.5PM0K	1
mary	{bcrypt}$2a$04$eFytJDGtjbThXa80FyOOBuFdK2IwjyWefYkMpiBEFlpBwDH.5PM0K	1
susan	{bcrypt}$2a$04$eFytJDGtjbThXa80FyOOBuFdK2IwjyWefYkMpiBEFlpBwDH.5PM0K	1

Spring Security Password Storage

- In Spring Security 5, passwords are stored using a specific format

```
{bcrypt}encodedPassword
```

username	password	enabled
john	{bcrypt}$2a$04$eFytJDGtjbThXa80FyOOBuFdK2IwjyWefYkMpiBEFlpBwDH.5PM0K	1
mary	{bcrypt}$2a$04$eFytJDGtjbThXa80FyOOBuFdK2IwjyWefYkMpiBEFlpBwDH.5PM0K	1
susan	{bcrypt}$2a$04$eFytJDGtjbThXa80FyOOBuFdK2IwjyWefYkMpiBEFlpBwDH.5PM0K	1

Spring Security Password Storage

- In Spring Security 5, passwords are stored using a specific format

```
{bcrypt}encodedPassword
```

Password column must be at least 68 chars wide

{bcrypt} - 8 chars
***encodedPassword* - 60 chars**

username	password	enabled
john	{bcrypt}$2a$04$eFytJDGtjbThXa80FyOOBuFdK2IwjyWefYkMpiBEFlpBwDH.5PM0K	1
mary	{bcrypt}$2a$04$eFytJDGtjbThXa80FyOOBuFdK2IwjyWefYkMpiBEFlpBwDH.5PM0K	1
susan	{bcrypt}$2a$04$eFytJDGtjbThXa80FyOOBuFdK2IwjyWefYkMpiBEFlpBwDH.5PM0K	1

Modify DDL for Password Field

```
CREATE TABLE `users` (
  `username` varchar(50) NOT NULL,
  `password`     char(68) NOT NULL,
  `enabled` tinyint(1) NOT NULL,

  PRIMARY KEY (`username`)

) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

Modify DDL for Password Field

```
CREATE TABLE `users` (
  `username` varchar(50) NOT NULL,
  `password`     char(68) NOT NULL,
  `enabled` tinyint(1) NOT NULL,

  PRIMARY KEY (`username`)

) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

Password column must be at least 68 chars wide

{bcrypt} - 8 chars
***encodedPassword* - 60 chars**

Step 1: Develop SQL Script to setup database tables

```
INSERT INTO `users`
VALUES
('john','{bcrypt}$2a$04$eFytJDGtjbThXa80FyOOBuFdK2IwjyWefYkMpiBEFlpBwDH.5PM0K',1),
```

Step 1: Develop SQL Script to setup database tables

The encrypted password: fun123

```
INSERT INTO `users`
VALUES
('john','{bcrypt}$2a$04$eFytJDGtjbThXa80FyOOBuFdK2IwjyWefYkMpiBEFlpBwDH.5PM0K',1),
```

Step 1: Develop SQL Script to setup database tables

The encoding algorithm id

The encrypted password: fun123

```
INSERT INTO `users`
VALUES
('john','{bcrypt}$2a$04$eFytJDGtjbThXa80FyOOBuFdK2IwjyWefYkMpiBEFlpBwDH.5PM0K',1),
```

Step 1: Develop SQL Script to setup database tables

The encoding algorithm id

The encrypted password: fun123

```
INSERT INTO `users`
VALUES
('john','{bcrypt}$2a$04$eFytJDGtjbThXa80FyOOBuFdK2IwjyWefYkMpiBEFlpBwDH.5PM0K',1),
```

Let's Spring Security know the passwords are stored as encrypted passwords: bcrypt

Step 1: Develop SQL Script to setup database tables

The encoding algorithm id

The encrypted password: fun123

```
INSERT INTO `users`
VALUES
('john','{bcrypt}$2a$04$eFytJDGtjbThXa80FyOOBuFdK2IwjyWefYkMpiBEFlpBwDH.5PM0K',1),

('mary','{bcrypt}$2a$04$eFytJDGtjbThXa80FyOOBuFdK2IwjyWefYkMpiBEFlpBwDH.5PM0K',1),

('susan','{bcrypt}$2a$04$eFytJDGtjbThXa80FyOOBuFdK2IwjyWefYkMpiBEFlpBwDH.5PM0K',1);
```

Step 2: Point to New Database Schema

File: src/main/resources/persistence-mysql.properties

```
#
# JDBC connection properties
#
jdbc.driver=com.mysql.jdbc.Driver
jdbc.url=jdbc:mysql://localhost:3306/spring_security_demo_bcrypt?useSSL=false
jdbc.user=springstudent
jdbc.password=springstudent

#
# Connection pool properties
#
connection.pool.initialPoolSize=5
connection.pool.minPoolSize=5
connection.pool.maxPoolSize=20
connection.pool.maxIdleTime=3000
```

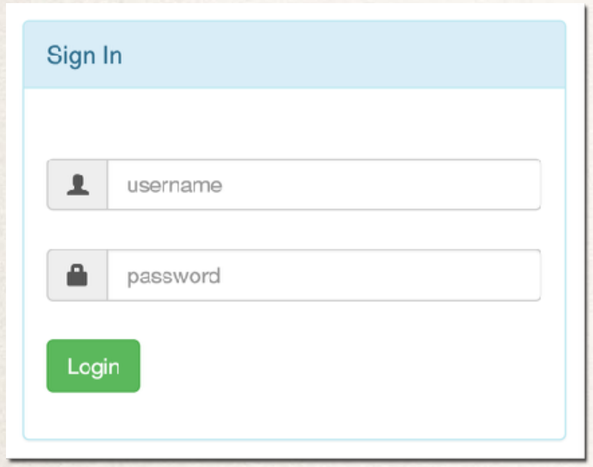
Step 2: Point to New Database Schema

File: src/main/resources/persistence-mysql.properties

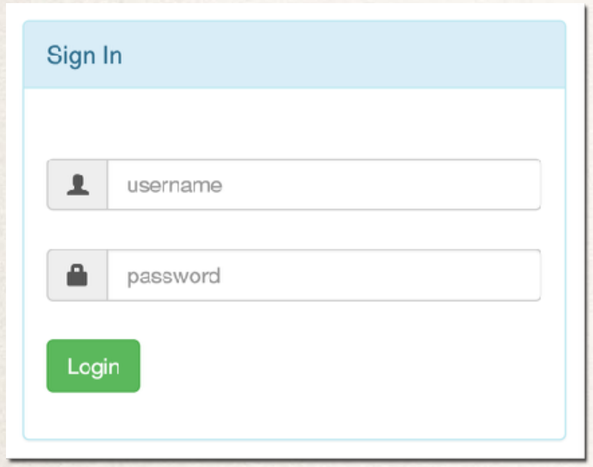
```
#
# JDBC connection properties
#
jdbc.driver=com.mysql.jdbc.Driver
jdbc.url=jdbc:mysql://localhost:3306/spring_security_demo_bcrypt?useSSL=false
jdbc.user=springstudent
jdbc.password=springstudent

#
# Connection pool properties
#
connection.pool.initialPoolSize=5
connection.pool.minPoolSize=5
connection.pool.maxPoolSize=20
connection.pool.maxIdleTime=3000
```

Spring Security Login Process

Spring Security Login Process

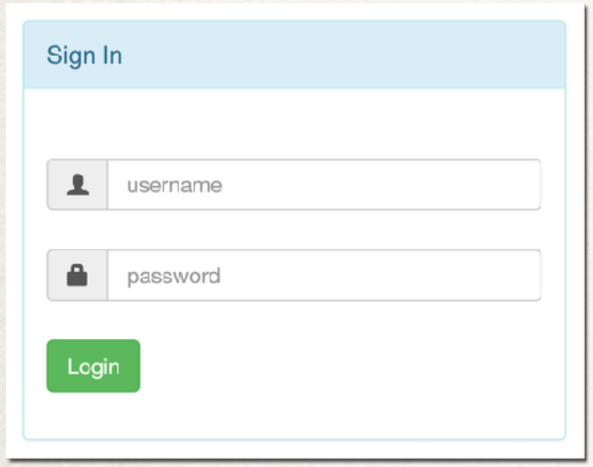


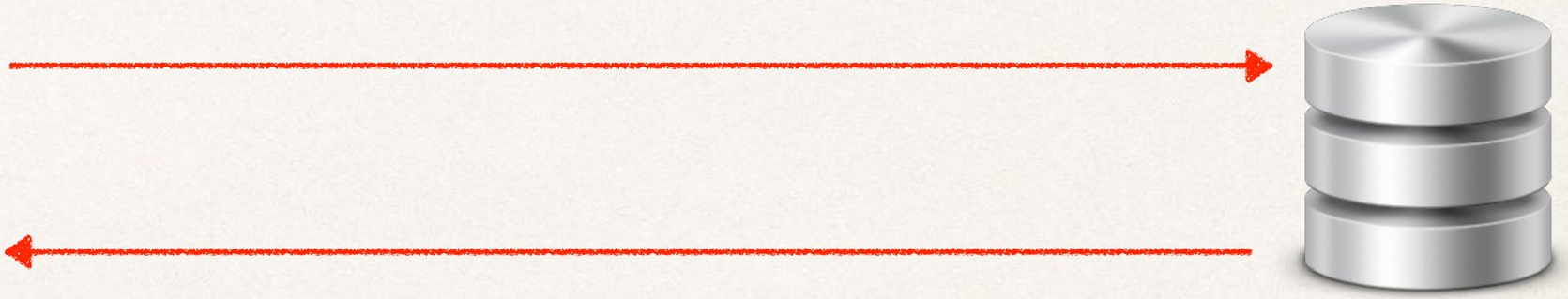
Spring Security Login Process



username	password	enabled
john	{bcrypt}$2a$04$eFytJDGtjbThXa80FyOOBuFdK2IwjyWefYkMpiBEFlpBwDH.5PM0K	1
mary	{bcrypt}$2a$04$eFytJDGtjbThXa80FyOOBuFdK2IwjyWefYkMpiBEFlpBwDH.5PM0K	1
susan	{bcrypt}$2a$04$eFytJDGtjbThXa80FyOOBuFdK2IwjyWefYkMpiBEFlpBwDH.5PM0K	1

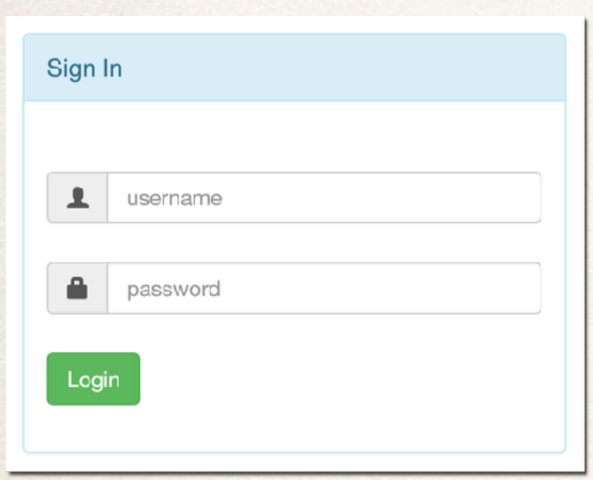
Spring Security Login Process

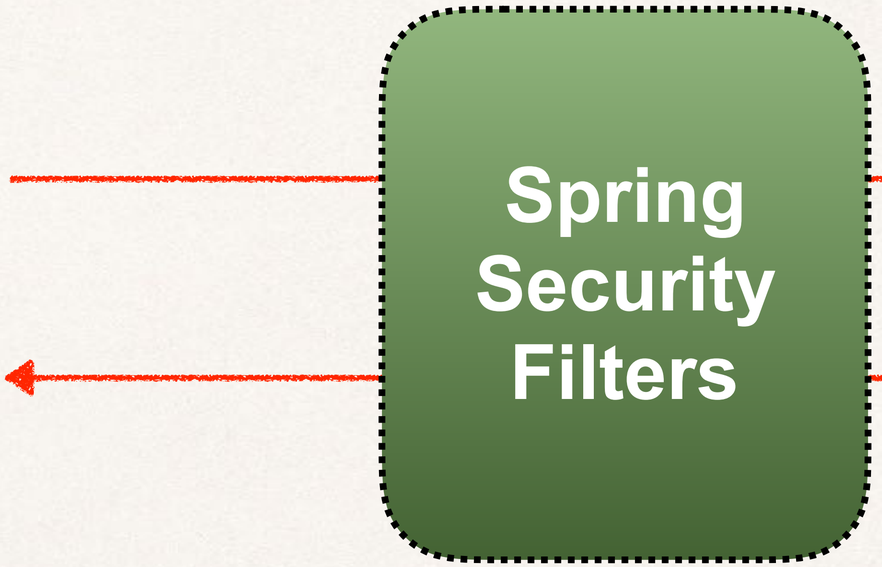




username	password	enabled
john	{bcrypt}$2a$04$eFytJDGtjbThXa80FyOOBuFdK2IwjyWefYkMpiBEFlpBwDH.5PM0K	1
mary	{bcrypt}$2a$04$eFytJDGtjbThXa80FyOOBuFdK2IwjyWefYkMpiBEFlpBwDH.5PM0K	1
susan	{bcrypt}$2a$04$eFytJDGtjbThXa80FyOOBuFdK2IwjyWefYkMpiBEFlpBwDH.5PM0K	1

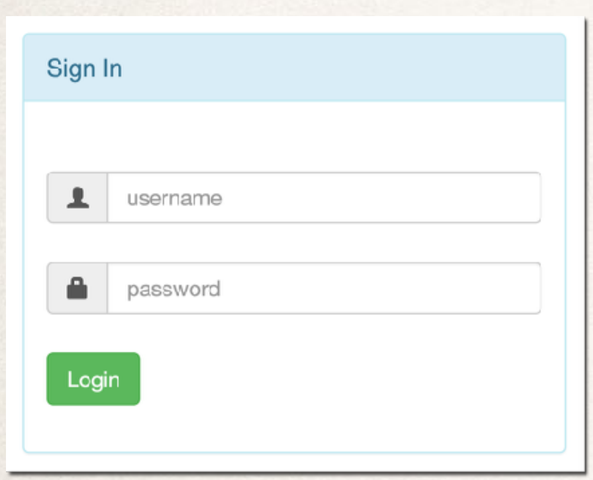
Spring Security Login Process





username	password	enabled
john	{bcrypt}$2a$04$eFytJDGtjbThXa80FyOOBuFdK2IwjyWefYkMpiBEFlpBwDH.5PM0K	1
mary	{bcrypt}$2a$04$eFytJDGtjbThXa80FyOOBuFdK2IwjyWefYkMpiBEFlpBwDH.5PM0K	1
susan	{bcrypt}$2a$04$eFytJDGtjbThXa80FyOOBuFdK2IwjyWefYkMpiBEFlpBwDH.5PM0K	1

Spring Security Login Process

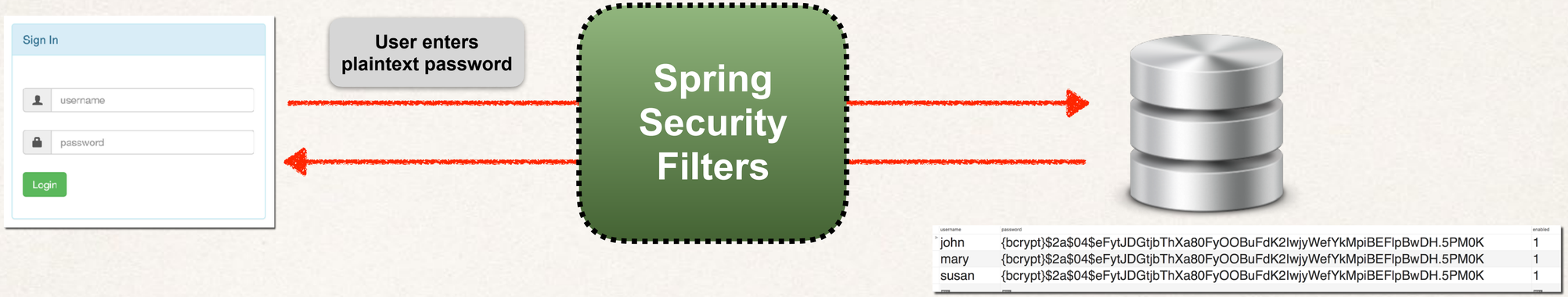


User enters plaintext password

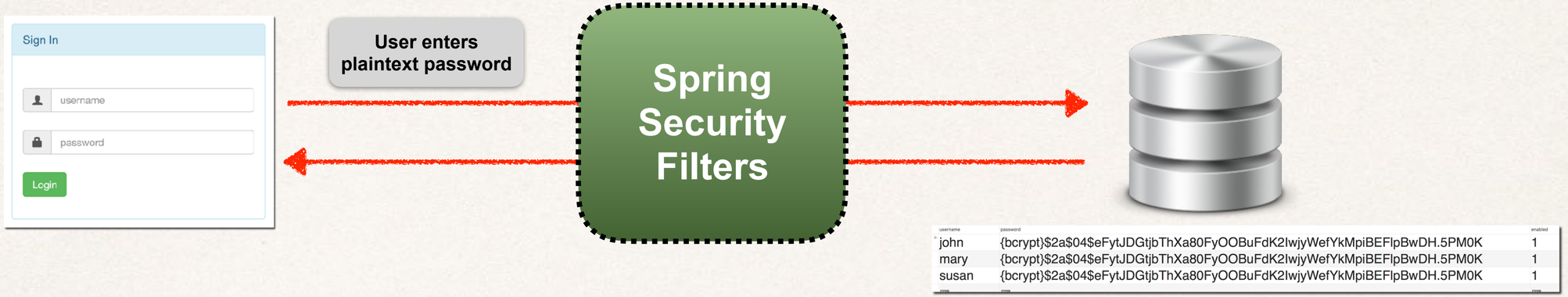
Spring Security Filters

username	password	enabled
john	{bcrypt}$2a$04$eFytJDGtjbThXa80FyOOBuFdK2IwjyWefYkMpiBEFlpBwDH.5PM0K	1
mary	{bcrypt}$2a$04$eFytJDGtjbThXa80FyOOBuFdK2IwjyWefYkMpiBEFlpBwDH.5PM0K	1
susan	{bcrypt}$2a$04$eFytJDGtjbThXa80FyOOBuFdK2IwjyWefYkMpiBEFlpBwDH.5PM0K	1

Spring Security Login Process



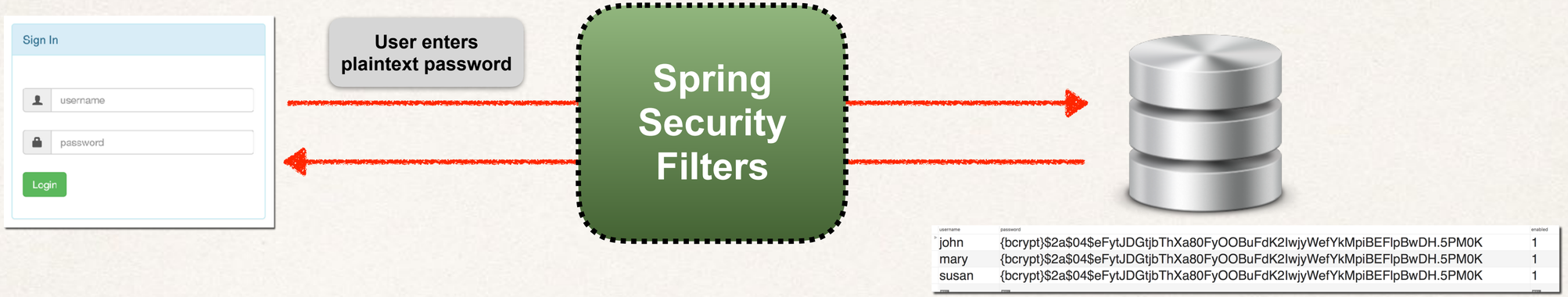
Spring Security Login Process



username	password	enabled
john	{bcrypt}$2a$04$eFytJDGtjbThXa80FyOOBuFdK2IwjyWefYkMpiBEFlpBwDH.5PM0K	1
mary	{bcrypt}$2a$04$eFytJDGtjbThXa80FyOOBuFdK2IwjyWefYkMpiBEFlpBwDH.5PM0K	1
susan	{bcrypt}$2a$04$eFytJDGtjbThXa80FyOOBuFdK2IwjyWefYkMpiBEFlpBwDH.5PM0K	1

1. Retrieve password from db for the user

Spring Security Login Process

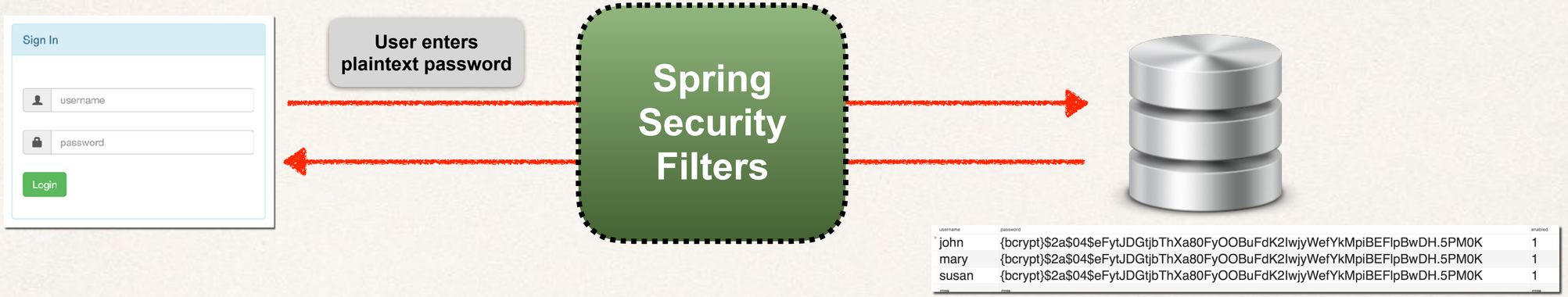


username	password	enabled
john	{bcrypt}$2a$04$eFytJDGtjbThXa80FyOOBuFdK2IwjyWefYkMpiBEFlpBwDH.5PM0K	1
mary	{bcrypt}$2a$04$eFytJDGtjbThXa80FyOOBuFdK2IwjyWefYkMpiBEFlpBwDH.5PM0K	1
susan	{bcrypt}$2a$04$eFytJDGtjbThXa80FyOOBuFdK2IwjyWefYkMpiBEFlpBwDH.5PM0K	1

1. Retrieve password from db for the user

2. Read the encoding algorithm id (bcrypt etc)

Spring Security Login Process

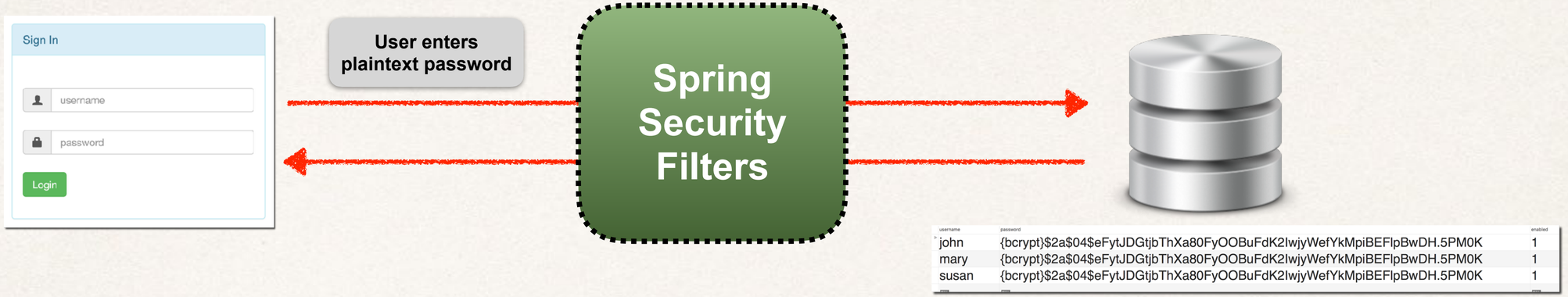


1. Retrieve password from db for the user

2. Read the encoding algorithm id (bcrypt etc)

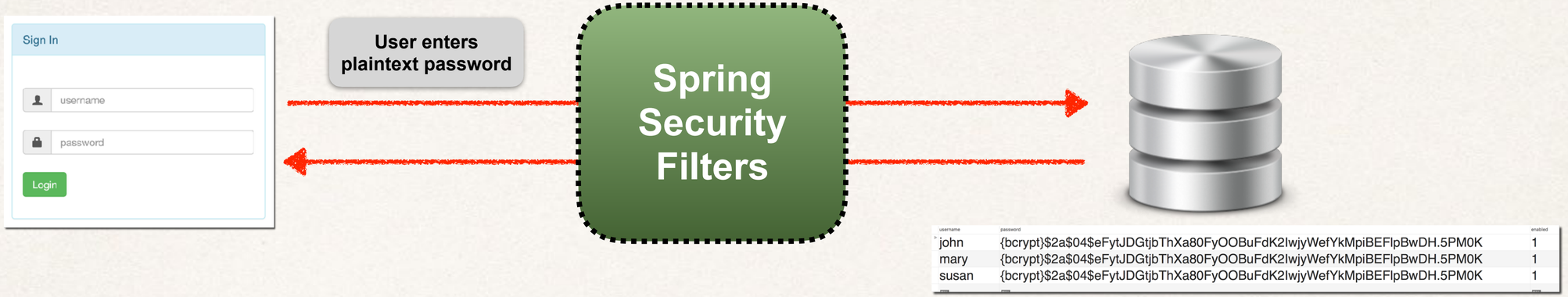
3. For case of bcrypt, encrypt plaintext password from login form (using salt from db password)

Spring Security Login Process



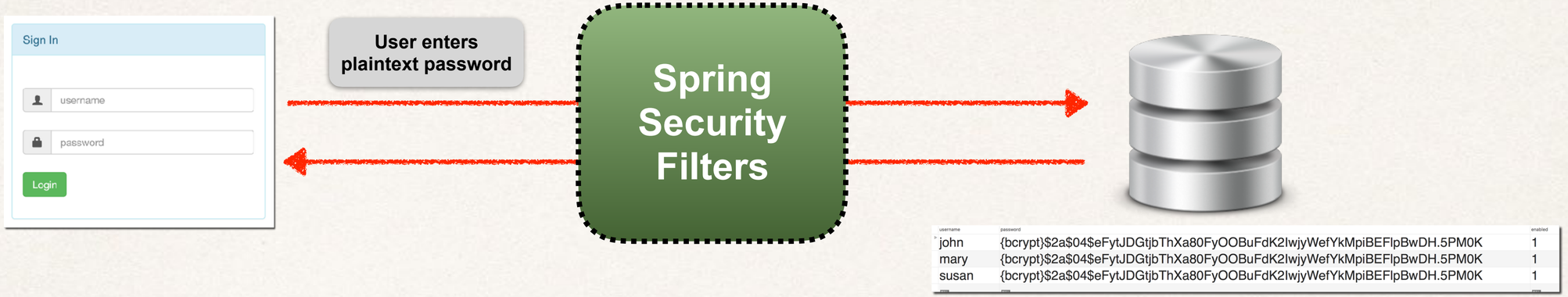
1. Retrieve password from db for the user
2. Read the encoding algorithm id (bcrypt etc)
3. For case of bcrypt, encrypt plaintext password from login form (using salt from db password)
4. Compare encrypted password from login form WITH encrypted password from db

Spring Security Login Process



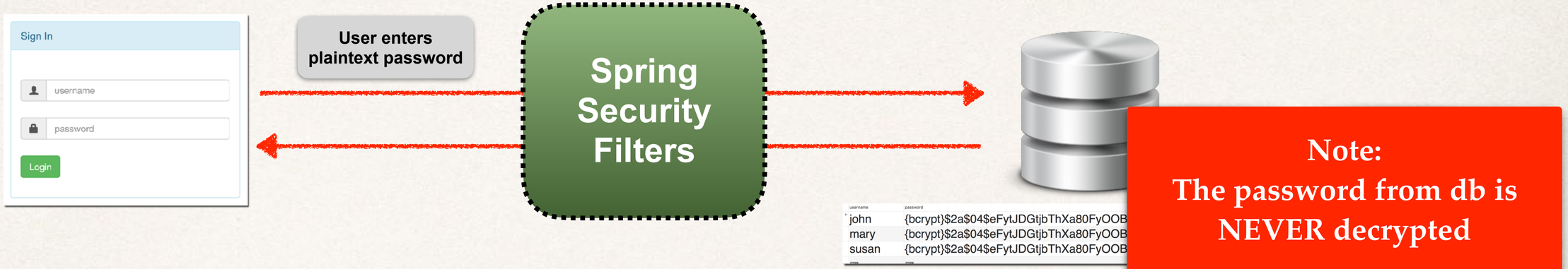
1. Retrieve password from db for the user
2. Read the encoding algorithm id (bcrypt etc)
3. For case of bcrypt, encrypt plaintext password from login form (using salt from db password)
4. Compare encrypted password from login form WITH encrypted password from db
5. If there is a match, login successful

Spring Security Login Process



1. Retrieve password from db for the user
2. Read the encoding algorithm id (bcrypt etc)
3. For case of bcrypt, encrypt plaintext password from login form (using salt from db password)
4. Compare encrypted password from login form WITH encrypted password from db
5. If there is a match, login successful
6. If no match, login NOT successful

Spring Security Login Process



1. Retrieve password from db for the user
2. Read the encoding algorithm id (bcrypt etc)
3. For case of bcrypt, encrypt plaintext password from login form (using salt from db password)
4. Compare encrypted password from login form WITH encrypted password from db
5. If there is a match, login successful
6. If no match, login NOT successful

Note:
The password from db is NEVER decrypted

Because bcrypt is a one-way encryption algorithm