Why is hashing used?

A hashing algorithm transforms a stream of data into a string of characters of fixed length.
For example the hash of password is **5f4dcc3b5aa765d61d8327deb882cf99**
But if output string is changed even slightly the hash value changes completely.
The hash of password1 is **7c6a180b36896a0a8c02787eeafb0e4c**

Hashing is a one way encryption, usually used in passwords. When a password is registered it is hashed and on login the password is checked with the hash value. If the match is found login is successful.

Since the attacker doesn't have the actual value but the hash value, he cannot go backward and get the original password.

However if a commonly known password is used, it is easy to get the original password from the hash value using the rainbow table. A rainbow table is a database of commonly used passwords and their corresponding hash values. So hashing by itself it isn't enough to protect passwords.

So more secure techniques are used:

SALT:

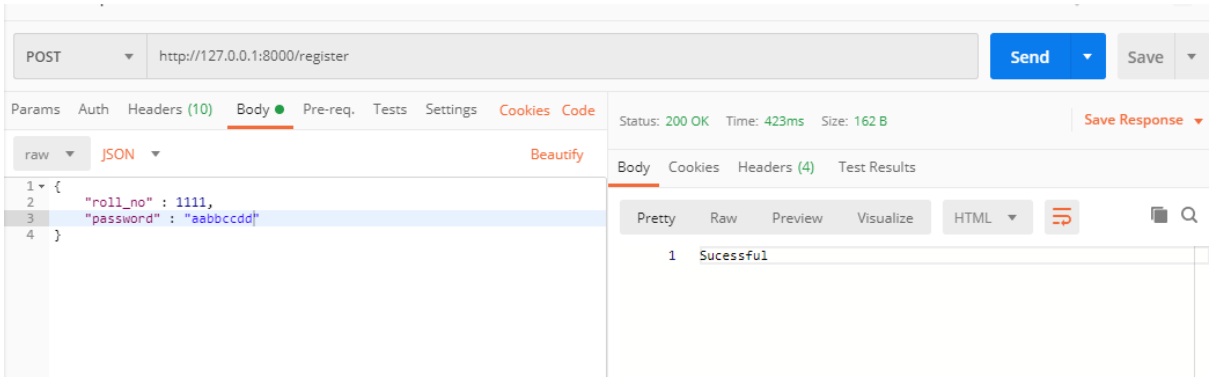
Salt is a short string of random characters that is appended to the password before they are hashed. This helps in preventing rainbow table attacks.

For e.g.: Password: **qwerty** may be in the rainbow table.

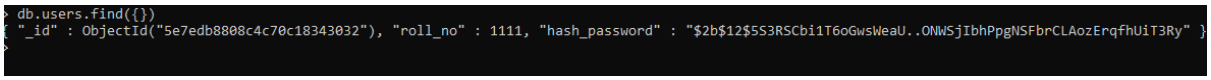
> However **qwertyP#)!z** is not likely to exist in the database. Here **P#)!z** is the salt used.

Salts are usually stored as plaintext and are added to the password, before storing in the database. The users are usually unaware of the salt.

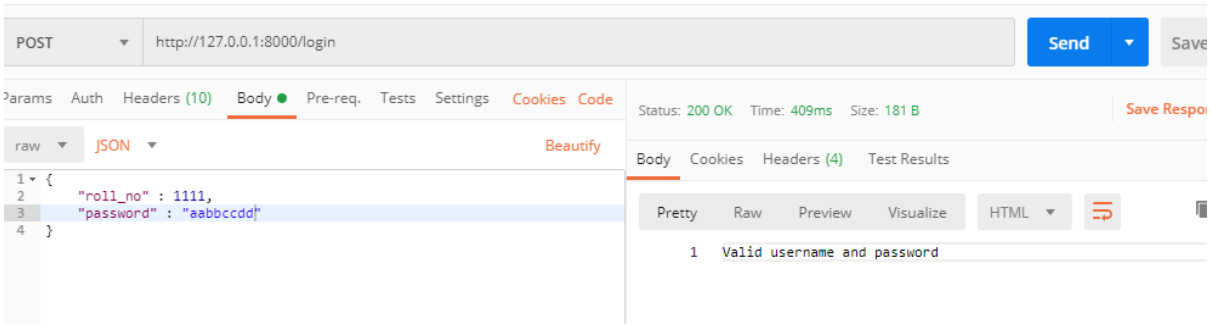
Hash (password + salt)



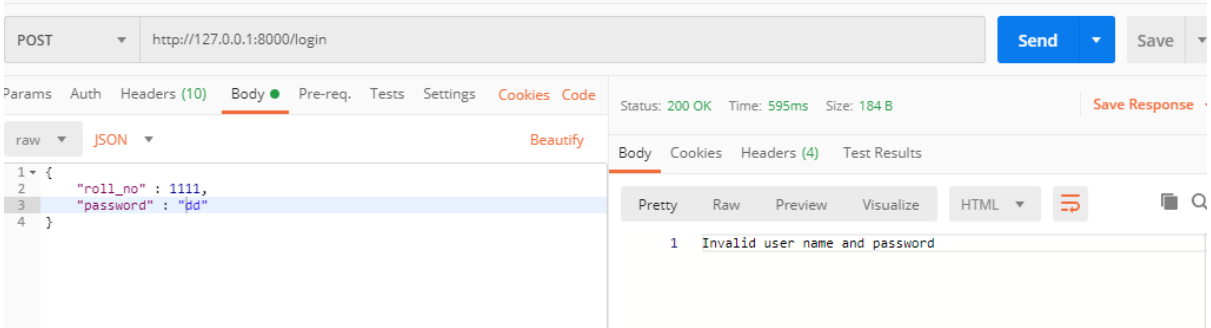
Register



Database



Successful login



Unsuccessful login

Advantage:

Salt aims at avoiding the issue with rainbow tables.

Disadvantage:

However the user can still obtain the password, if he gets the salt value and the location to add in the string.

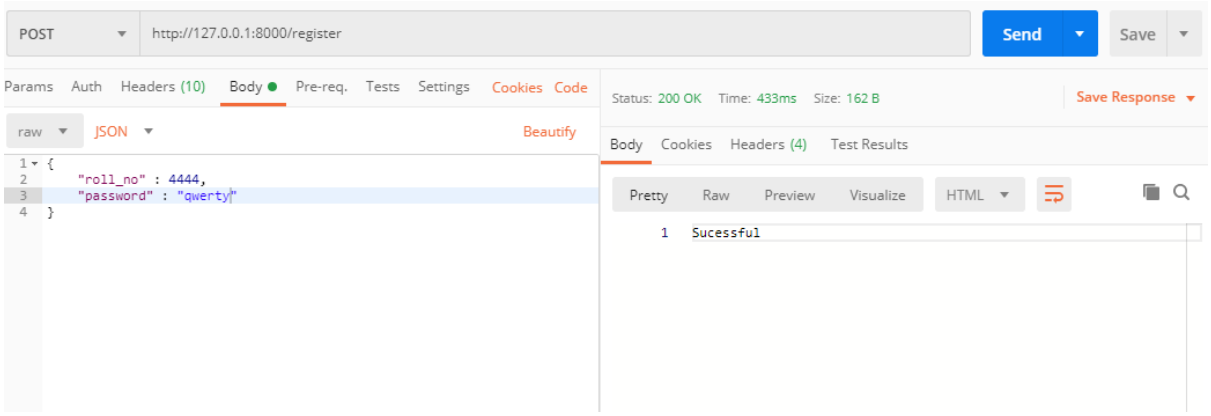
PEPPER:

Pepper is a short string or character appended at the end of the passwords. Peppers are random and different for each password.

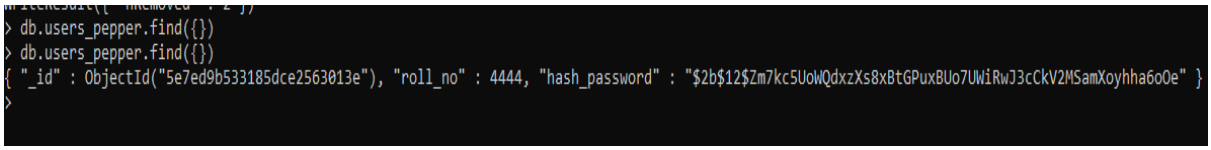
For e.g.: Password: **qwerty**

Pepper is letter '**e**'.

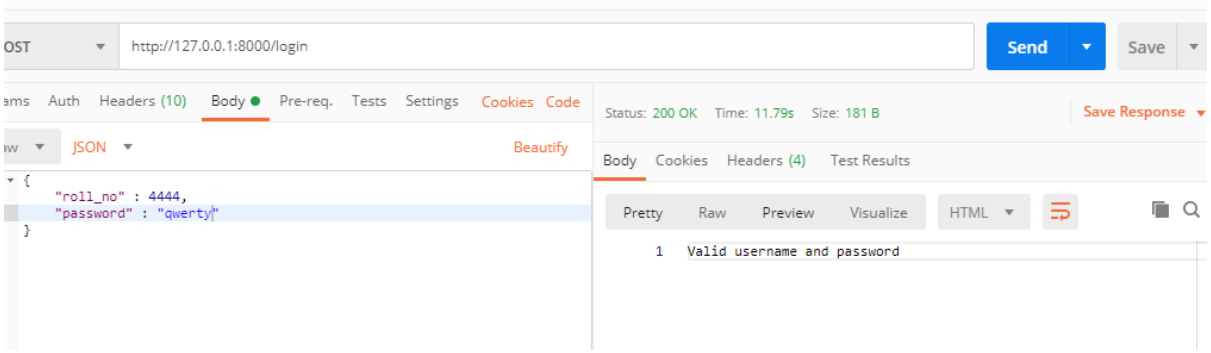
Hash (Password + Pepper)



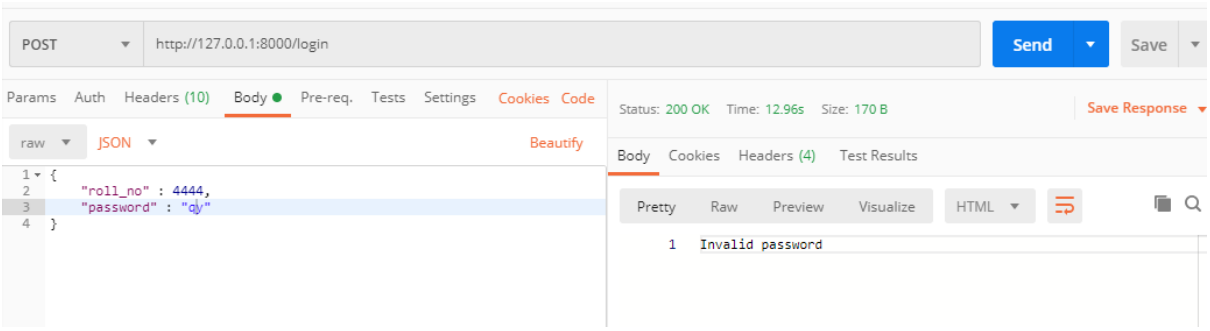
Register user



Database



Successful login



Unsuccessful login

Similar to salt the users are unaware of the Pepper. The pepper value is not stored. So when user enters a password the website cycles through all possible peppers until it matches with the hash.

Advantage:

This method is more secure since no one knows what the pepper is even the website, until it goes through all the possible options to find it.