LEVELS OF SECURITY FOR WEBAPPS

By Jasveer Singh & Parshant Khichi

Level 1

Storing usernames and passwords in plain text.

Level 1

```
{
    “email” : abc@g.com,
    “password” : “123”
}
```

Risks

Employees can access any user’s password.

If someone hacks into our server than this could be pretty good loot for them.

Level 2 Encryption

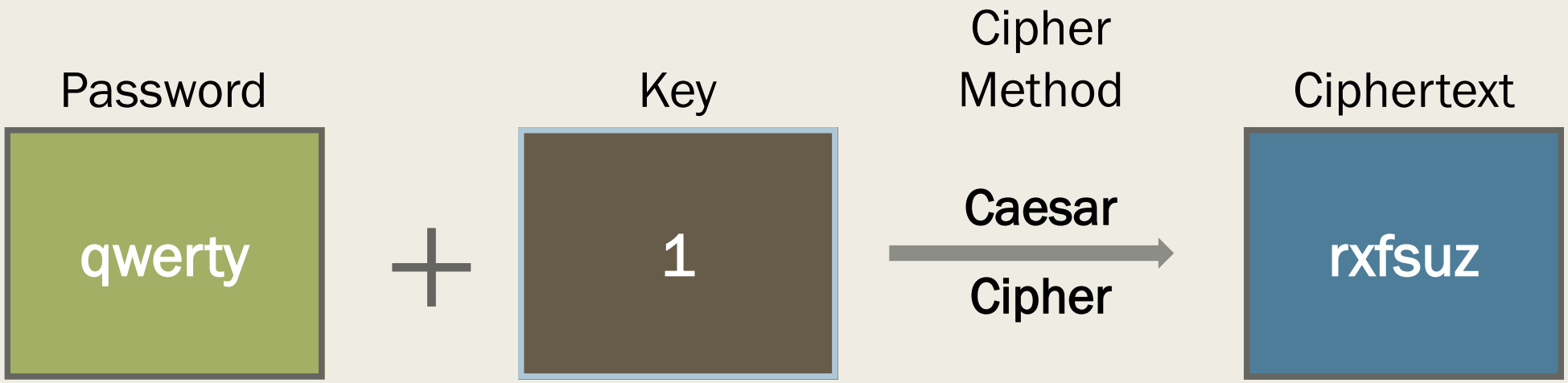
Encryption

Converting information into secret code that hides the information's true meaning

S E C R E T → E R T E C S

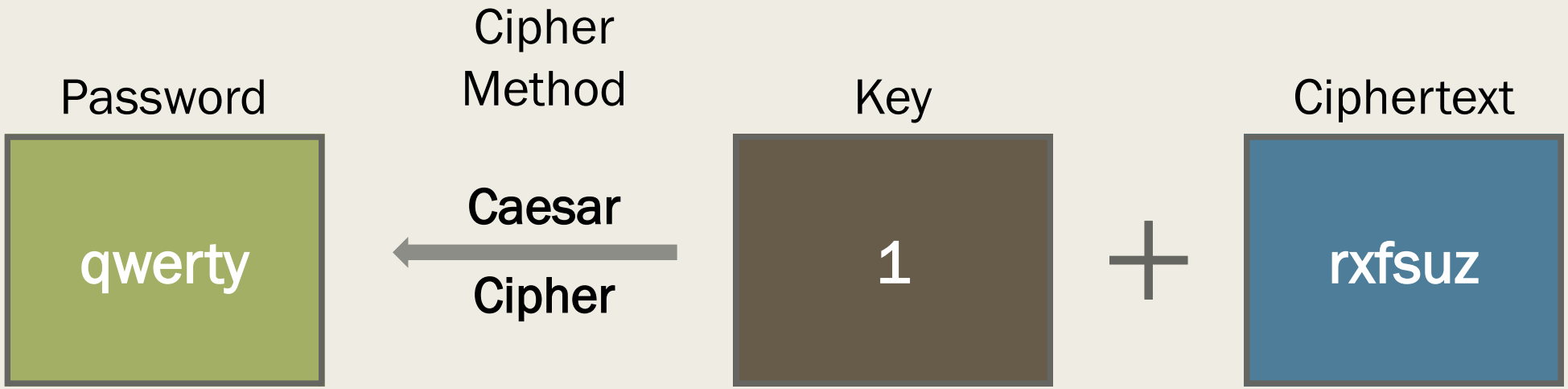
Level 2

Encryption



Level 2

Decryption



Implementing level 2 security

mongoose-encryption

Code

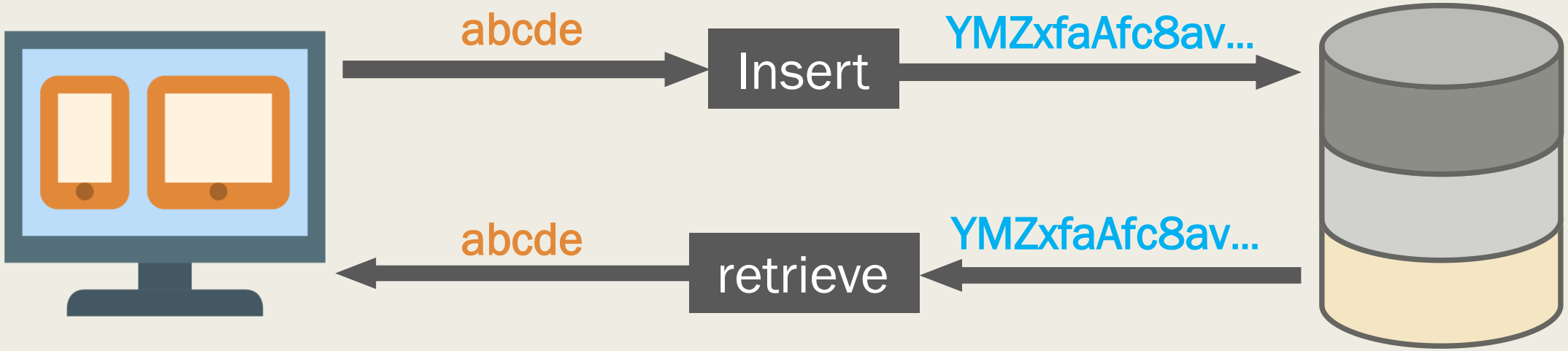
```
secret = “Thisisourlittlesecret”
userSchema.plugin(encrypt, { secret: secret , encryptedField: [“password”] })
```

Database

```
{
    “email” : abc@g.com,
    “_ct” : “YMZxfaAfc8avUNFnd8ga6YFb54KB4avH7bsBubJBbklusda==”
}
```

Level 2

mongoose-encryption



Using environment variables

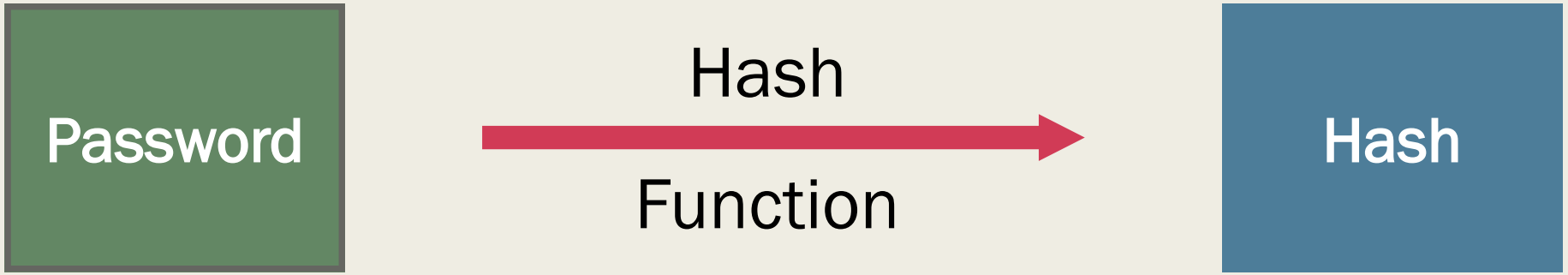
.env

.env

```
SECRET=Thisisourlittlesecret
API_KEY=shkanivphoibnvkhghf
```

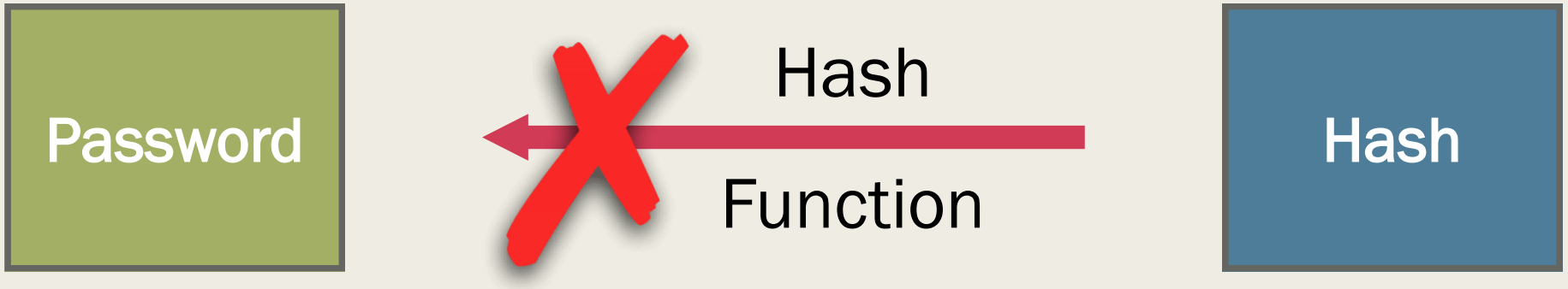
Level 3 Hashing

Hashing is simply passing some data through a formula that produces a result, called a **hash.**



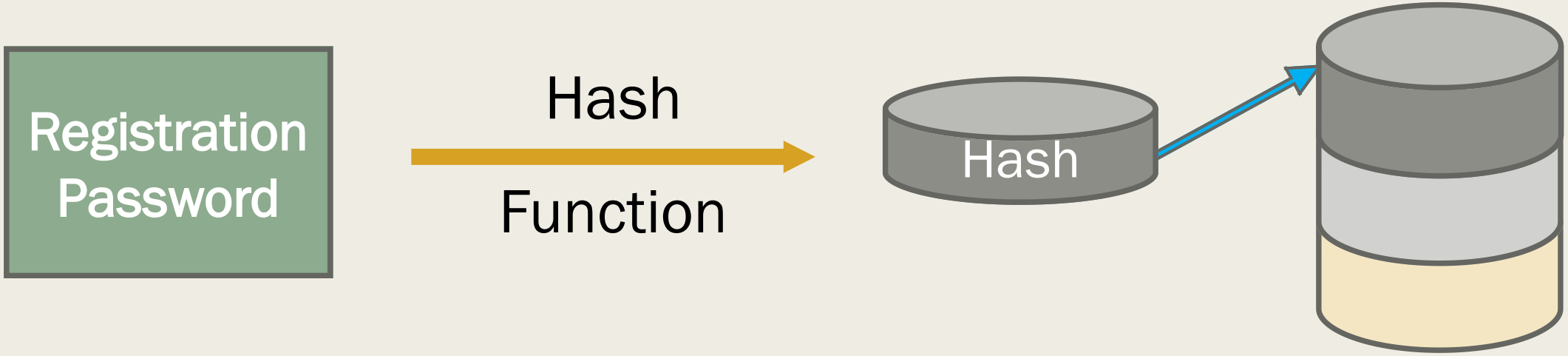
Level 3

- It is almost impossible to reverse this process.
- For each different string a unique hash is generated



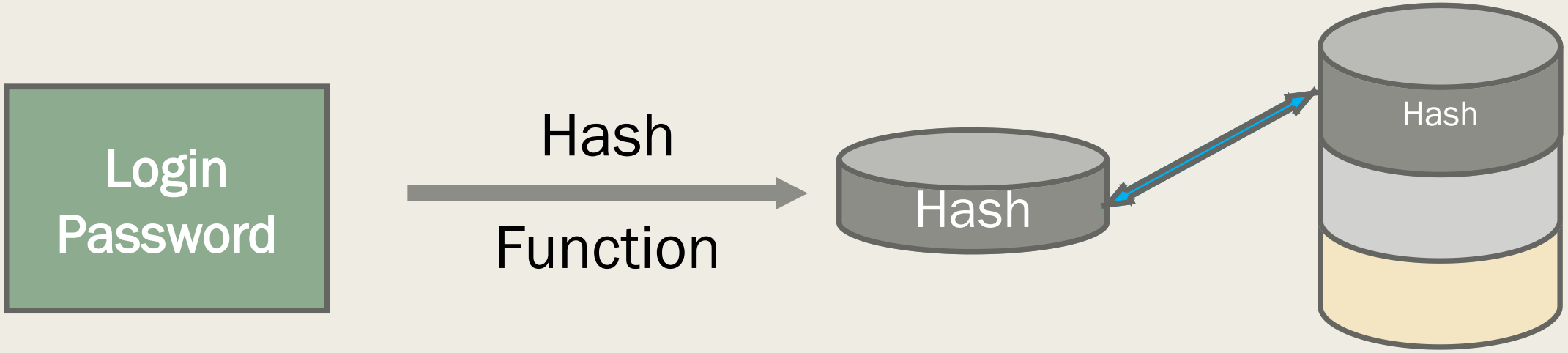
Level 3

While registering we'll store the hash into our database.



Level 3

While logging in we check if the generated hash matches with the one stored in the database



md5

The md5 message-digest algorithm is widely used hash function producing a 128-bit hash value.

md5(“qwerty”)

d8578edf8458ce06fbc5bb76a58c5ca4

Risks

Username	Hash
tony@gmail.com	d8578edf8458ce06fbc5bb76a58c5ca4
john@gmail.com	54b024e1f48df56a187ca7047ceb3751
karan@outlook.com	d8578edf8458ce06fbc5bb76a58c5ca4
rahul@yahoo.com	d8578edf8458ce06fbc5bb76a58c5ca4

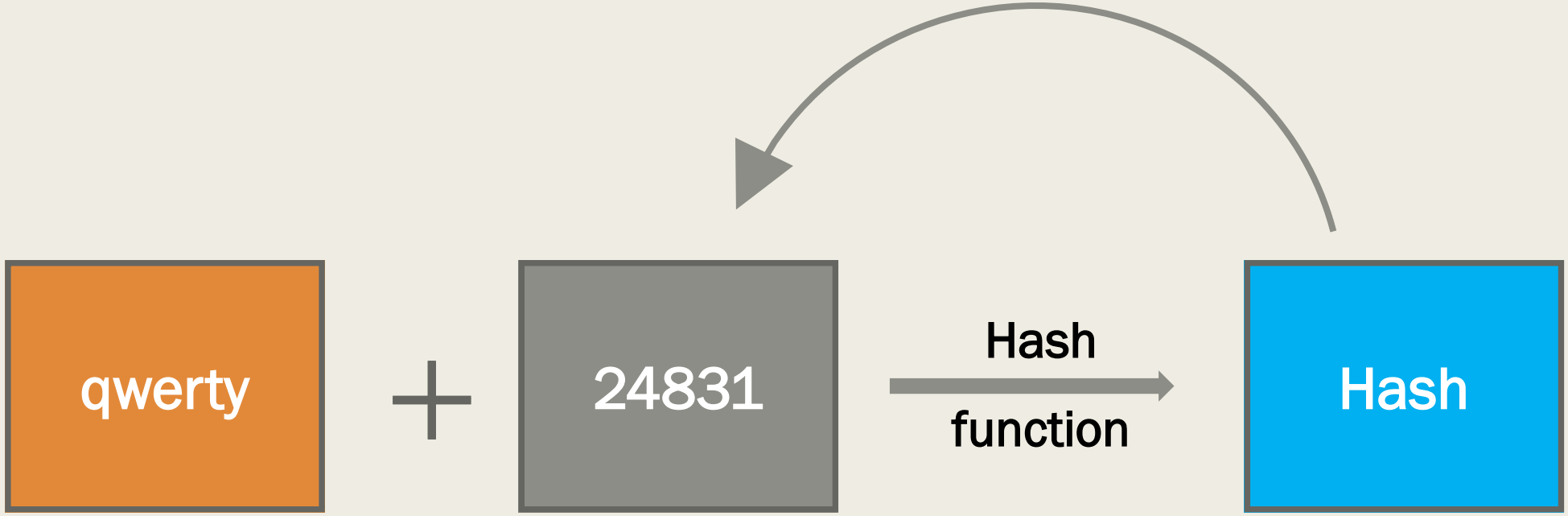
Same

Level 4 - Salting and Hashing

Salting

Level 4

Salt Rounds



bcrypt.js

bcrypt npm package is a JavaScript implementation of the bcrypt password hashing function that allows you to easily create a hash out of a password string.

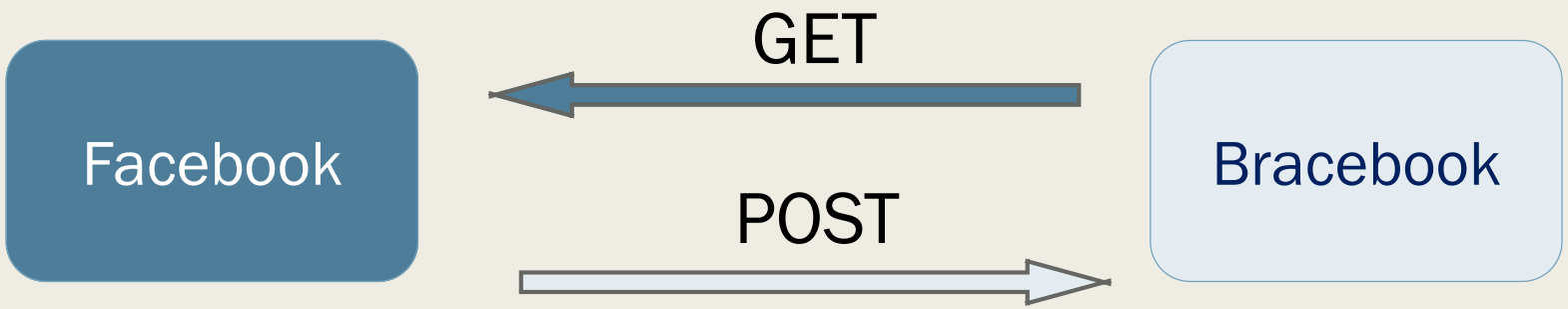
```
Const bcrypt = require(“bcrypt”)
Const saltrounds = 10
bcrypt.hash(password, saltrounds, function(err, hash) {
    .......
})
```

Level 5 OAuth

Open Authorization

OAuth is a way to get access to protected data from an application. It's safer and more secure than asking users to log in with passwords.

Level 5



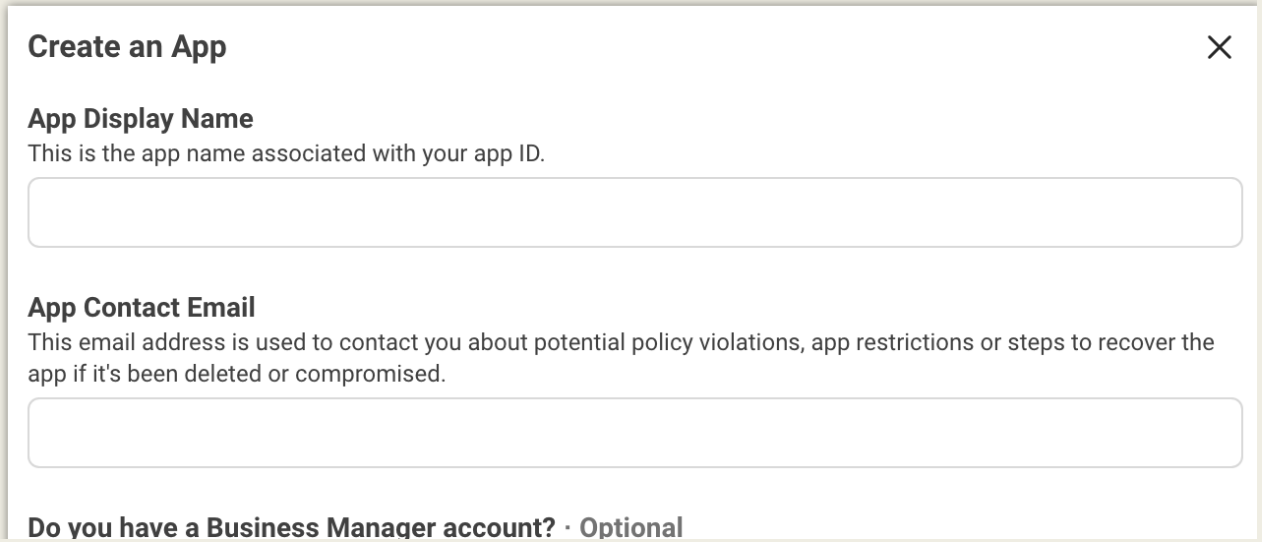
Users	
Name	Email
Ajay	aj@gmail.com
Angela	angela@gmail.com
Raju	raju@gmail.com

Users	
Name	Email
Ajay	aj@gmail.com
Angela	angela@gmail.com

Imlementing level-5 security

Step 1: ***Set up your app***

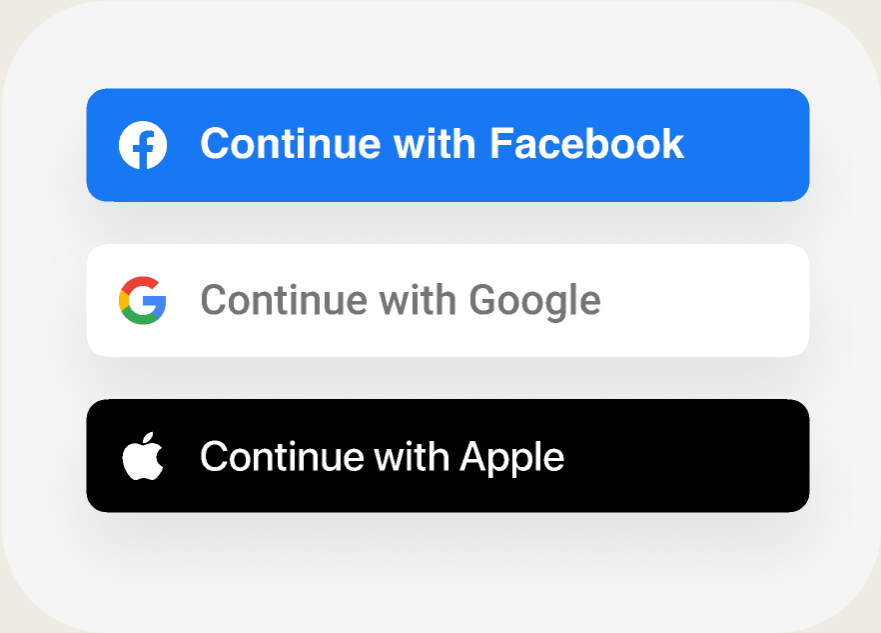
Telling these third parties about our app location. We have to set up our app their developer console and in return we'll get App Id.



Implementing level 5 security

Step 2: ***Redirect to Authenticate***

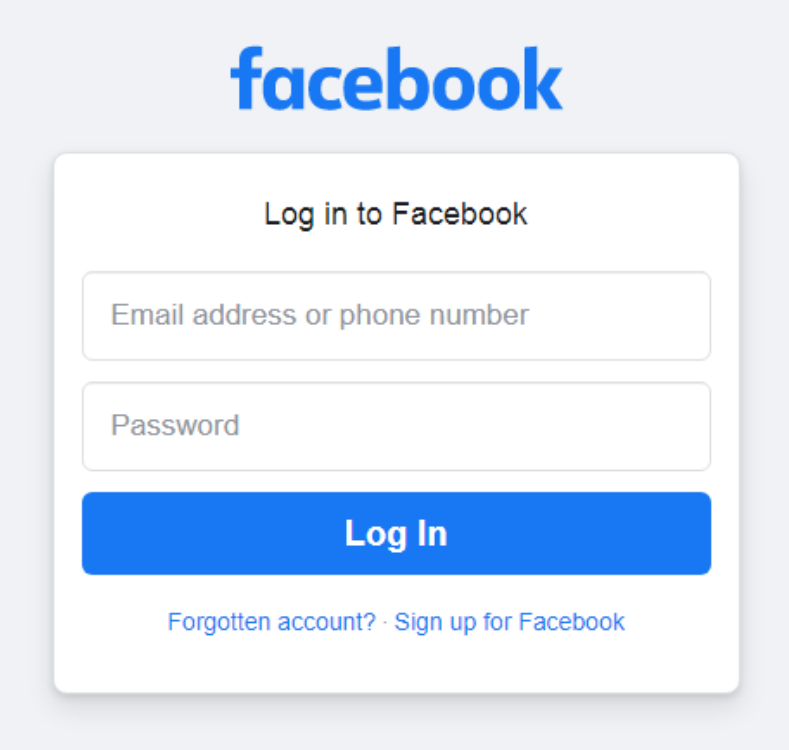
Giving the users the option of registering account using these third party app on our login page.



Implementing level 5 security

Step 3: ***User Logs In***

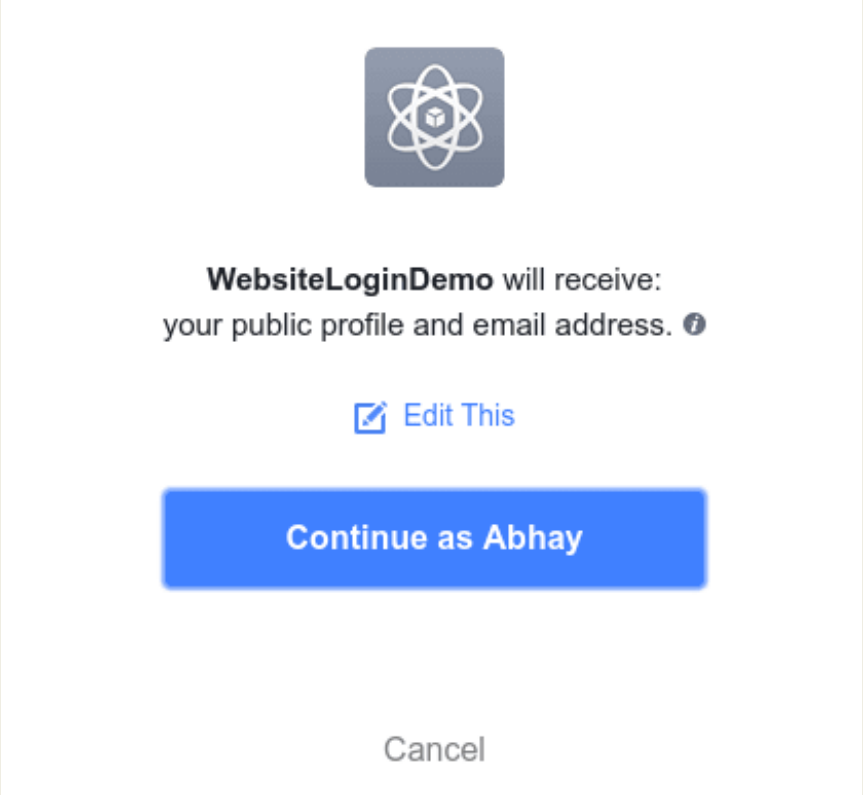
take users to the actual third party website(facebook, google, apple) and they login their using their actual credentials.



Implementing level 5 security

Step 4: ***User Grants Permissions***

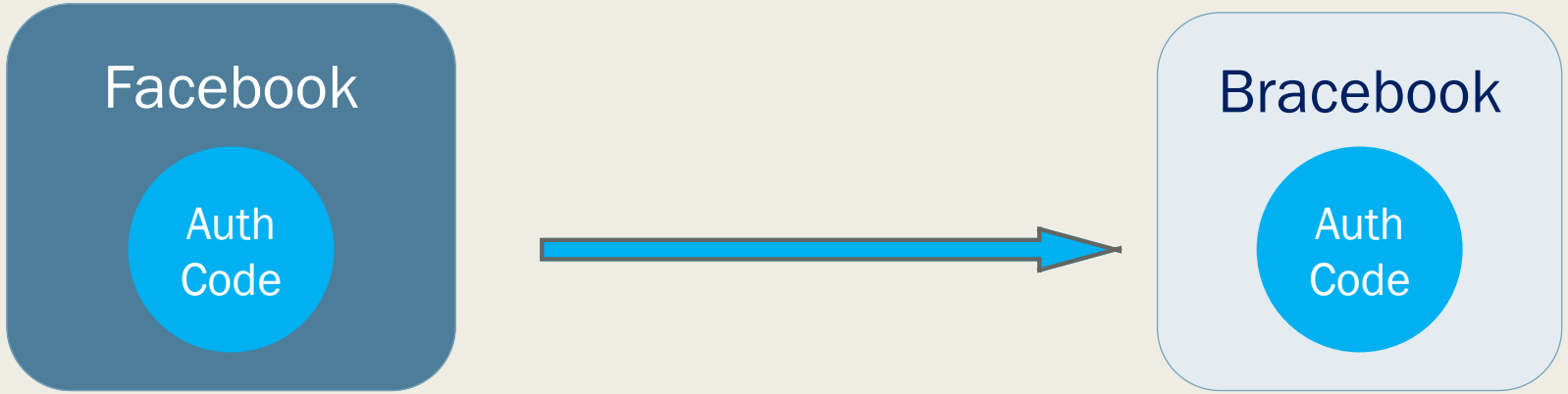
User reviews the permissions that our website is asking for.



Implementing level 5 security

Step 5: ***Receive Authorization code***

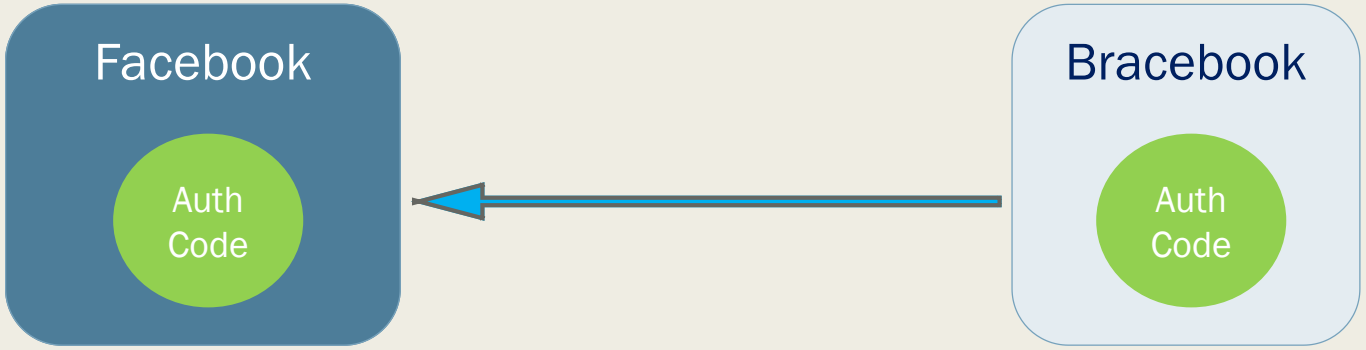
After successful login our app will receive authorization code. Which allows us to check that the user actual signed in to Facebook.



Implementing level 5 security

Step 6: ***Exchange AuthCode for Access Token***

Allows us to check that the user actual signed to Facebook.



Implementing level 5 security

Pros

- Highest level of security.
- Provides users a familiar interface and fast registration process.
- We don’t need to store the passwords, so we don’t need to take care of their security.

THANKS

BY:

Jasveer Singh
(190280060)

Parshant Khichi
(190280089)