How Authentication works:

The authentication is implemented in such a way that the secret information like password and api key are not transferred with the request openly. PhP’s inbuilt crypt(The blowfish hashing algorithm) function is used to hash the secret entities. In this application, there are five security tokens involved. They are username, password, api key id, api key and salt(user and system salt).

Publicly sharable – username, api key id and salt strings (both user and system)

Private Strings – password and api key

What User Knows - His username, password, api key id and api key, the hashing procedure and his own salt string.

What server knows – Everything what user knows except his salt string and the system salt which is used to hash his secret credentials to store in database.

The following steps illustrate how authentication is achieved.

1. User sends his username and api key id to the server.
2. Server verifies the username and api key Id against the database.
3. If the credentials are okay then the system returns the salt that is used to hash the user’s password and api key. If not, then invalid credential message is returned.
4. On receiving the salt, the user hashes the password and api key with the salt provided by the system. In that way, the server and user has same hash for their secret credentials.
5. Then the user forms a security string of format “hashed password|user salt string|hashed api key”.
6. Then the user hashes the above formed string with his own salt which we call it as request token.
7. Then the user sends his username, api key id, his own salt(we call it as request Salt) and request token with other required parameters to any of the functions defined in getProductAndCartDetails.php or purchaseProducts.php.
8. The server on receiving the request tries to authenticate the request token before actually executing the method it called.
9. The server tries to form the same request token of format “hashed password from database | user given salt | hashed api key from database”.
10. The above string is hashed with user given salt and compared with request token sent by the user.
11. If the hashes matches then the method is executed otherwise authentication failed message is returned.

Note:

1. For clarification, API key Id and API key is different. It is just like username and password. API key Id and username together is used to identify the API key for the user.
2. In request Token, string there is no space between the credential separator ('|') and the credentials.
3. Care should be taken while forming the salt string. As blowfish algorithm of crypt function accepts only 22 character length. Further recommended usage of the function can be seen in this link - http://php.net/manual/en/function.crypt.php.

Method Name	Description	Parameters Required	Output On Success	Failure Condition	Output On Failure
getAuthSalt	This method is used to get the salt, the server uses to hash user password and API key	**userName** – username of the user. **apiKeyId** – API Key Id of the user	Return the message – "Auth Salt Obtained Successfully". and the salt used .	When either of the **userName** or **apiKeyId** is invalid	Returns "Invalid credential" message.
				When either one or all of the parameter is missing.	Return message "Invalid Request userName or apiKeyId parameter is missing "

Method Name	Description	Parameters Required	Output On Success	Failure Condition	Output On Failure
Authenticate	This method is used to authenticate the user. The user does not call this method directly. The server call this method when the user calls any of the function defined in getProductAndCartDetails.php or purchaseProducts.php	**userName** – username of the user. **apiKeyId** – API Key Id of the user. **requestSalt** – The salt given by user **requestToken** – The security string	Return the message – "Authentication Successful".	When either of the **userName** or **apiKeyId** or **requestToken** is invalid	Returns "Invalid credential" message if **userName** or **apiKeyId** is invalid. Returns "Authentication unsuccessful" if **requestToken** is invalid.
				When either one or all of the parameter is missing	Returns "Authentication unsuccessful" message.

General Comments -

1. All outputs are json encoded.
2. When sending a json encoded string, please do url encode the json encoded string before sending it as request.
3. Only POST method is allowed for functions defined in authentication.php file.
4. All the security tokens should be sent through post request body for the calls accessing the methods described in this file.
5. The method name should be assigned as a value to action parameter for sending a request.
6. A sample access URL for the functions defined in this file http://localhost/api/authentication.php.
7. For accessing methods in getProductAndCartDetails.php, send the security credentials in http header. For accessing methods in purchaseProducts.php, send the security tokens in post request body.

8. The Following code shows how to make a request to get a system salt in PhP curl.

```
$userName = "admin";
 $apiKeyId = "adminKey";
 $apiKey = "ffd7fcc5-fad2-44e4-af28-c467c4c34cbd";
$password = "LwkPC&RgUe";
$url = "http://localhost/api/authentication.php";
$post =  array('action'=>'getAuthSalt','userName'=>$userName,'apiKeyId'=>$apiKeyId);
$curl = curl_init($url);
curl_setopt($curl, CURLOPT_TIMEOUT, 3);
curl_setopt($curl, CURLOPT_RETURNTRANSFER, 1);
curl_setopt($curl, CURLOPT_SSL_VERIFYHOST, 0);
curl_setopt($curl, CURLOPT_SSL_VERIFYPEER, 0);
curl_setopt($curl, CURLOPT_POST, 1);
curl_setopt($curl, CURLOPT_POSTFIELDS, $post);
$response = curl_exec($curl);
$httpCode = curl_getinfo($curl, CURLINFO_HTTP_CODE);
$authResponse = json_decode($response,true);
// $authResponse['salt'] will contain the salt if the call is successful.
```

9. The Following Code shows how to frame a request token and make a request to a method defined in getProductAndCartDetails.php

```
                    $hashedApiKey = crypt($apiKey,'$2a$10$'.$authResponse['salt'].'$');
                    $hashedPassword = crypt($password,'$2a$10$'.$authResponse['salt'].'$');
                    $requestSalt = "heyiamadminallowmetouse";
                    $requestToken = crypt($hashedPassword.'|'.$requestSalt.'|'.$hashedApiKey,'$2a$10$'.$requestSalt.'$');
                     $url = "http://localhost/api/getProductAndCartDetails.php?action=getAllProducts";
                    $curl = curl_init($url);
                     curl_setopt($curl, CURLOPT_TIMEOUT, 3);
                    curl_setopt($curl, CURLOPT_RETURNTRANSFER, 1);
                    curl_setopt($curl, CURLOPT_SSL_VERIFYHOST, 0);
                    curl_setopt($curl, CURLOPT_SSL_VERIFYPEER, 0);
                     curl_setopt($curl, CURLOPT_HTTPHEADER,
array('userName:'.$userName,'apiKeyId:'.$apiKeyId,'requestSalt:'.urlencode($requestSalt),'requestToken:'.urlencode($requestToken)));
      $resp = curl_exec($curl);
```

```
$httpCode = curl_getinfo($curl, CURLINFO_HTTP_CODE);
 $array = json_decode($resp,true);
 if($httpCode==200) {
        echo print_r(json_decode($array['productDetails'],true),true);
}
curl_close($curl);
```

How hash of password and api key is produced for user admin and john

For Admin,

The password is LwkPC&RgUe

The API key is ffd7fcc5-fad2-44e4-af28-c467c4c34cbd

The salt for admin is somerandomsaltforadmin

The hashed string is calculated through crypt function as follows:

```
echo crypt('LwkPC&RgUe','$2a$10$somerandomsaltforadmin$').PHP_EOL;
echo crypt('ffd7fcc5-fad2-44e4-af28-c467c4c34cbd','$2a$10$somerandomsaltforadmin$').PHP_EOL;
```

For John,

The password is hsdbfgvfwd

The API key is aff1f9b5-2ff5-45f5-99e1-2b1f5c0fda7c

The salt for john is donothavesaltlikethisy

The hashed string is calculated through crypt function as follows:

```
echo crypt('hsdbfgvfwd','$2a$10$donothavesaltlikethisy$').PHP_EOL;
echo crypt('aff1f9b5-2ff5-45f5-99e1-2b1f5c0fda7c','$2a$10$donothavesaltlikethisy$').PHP_EOL;
```