**User Management Solution Guide**

This document describes how we are going to implement User Management Solution for MCB Project in Detail.

1. **Creating a User Management Tables**:

We need 3 tables to make the User Access for given services

* **MCB\_USER\_INFO** Table for maintaining User Information
* **MCB\_SERVICES\_INFO** Table for maintaining Services Info
* **MCB\_USER\_SERVICE\_MAP\_INFO** Table for maintaining User and Service Mappings.

in Oracle 11G database , create above 3 tables with given columns. we are attaching the Sql queries for each table creation and constraints and insert queries.

**MCB\_USER\_INFO**  Table Description

|  |  |  |  |
| --- | --- | --- | --- |
| **Column Name** | **Data type & Constraint** | **Description** | **Possible  Value example** |
| USER\_ID | Auto generated Primarykey column | Primary key for User Info table values start from 1 |  |
|  |  |  |  |
| USER\_NAME | VARCHAR(255) NOT NULL Primary Key | It is a MCB User , used as UserId to login, should be unique , it is a primary key column in DB | MCBADMIN |
| PASSWORD\_HASH | VARCHAR(255) ) NOT NULL | Password text after applying Hashing Algorithm | XXXXXXXX |
| USER\_ROLE | VARCHAR(20) | Role of the User | Administrator / other |
| ROLE\_DESCRIPTION | VARCHAR(255) | Describes the type of User Role | MCB Administrator,  ICPS User... |
|  |  |  |  |
|  |  |  |  |
| USER\_ACTIVE | VARCHAR(1) | Indicates this particular user is Active or Inactive | **Y(Yes) / N (NO)** |
| CREATED\_DATE | DATE | Date, on User Created initially | 16-Apr-2019 |
| UPDATED\_DATE | DATE | Date, on which User Modified recently | 17-Apr-2019 |

**MCB\_User\_Info** Table structure with sample data:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **USER\_ID (PK)** | **USER\_NAME** | **PASSWORD\_HASH** | **USER\_ROLE** | **ROLE\_DESCRIPTION** | **ACTIVE** | **CREATE\_DT** | **UPDATED\_DT** |
| 1 | MCBUSER | XXXXXX | Admin | Administrator | Y | 04/18/2019 | 04/20/2019 |

**MCB\_SERVICES\_INFO** Table Description

|  |  |  |  |
| --- | --- | --- | --- |
| **Column Name** | **Data type & Constraint** | **Description** | **Possible  Value example** |
| **SERVICE\_ID** | Auto generated PK |  | 1 |
| **SERVICE\_NAME** | Varchar(60) | Name of the Service |  |
| **SERVICE\_DESC** | Varchar(255) | Description of the Service |  |
|  |  |  |  |

**MCB\_SERVICES\_INFO** Table with sample data:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **SERVICE\_ID** | **SERVICE\_NAME** | **SERVICE\_DESC** | **CREATE\_DT** | **UPDATED\_DT** |
| **1** | **getDebitCreditCardInfo** | **XXXXXXXX** | 04/18/2019 | 04/20/2019 |
| **2** | **getCreditTransactionList** | **YYYYYYYYY** | 04/18/2019 | 04/20/2019 |

**USER-SERVICE Mapping** Tabel Description:

|  |  |  |  |
| --- | --- | --- | --- |
| **Column Name** | **Data type & Constraint** | **Description** | **Possible  Value example** |
| **USER\_ID** | Auto generated PK | USER ID from User Table | 1 |
| **SERVICE\_ID** | Varchar() | SERVICE\_ID from Service Table |  |
| **HAS\_ACCESS** | Varchar() | Given user has access to Service or not |  |
|  |  |  |  |

**USER-SERVICE Mapping** Tabel with sample data

|  |  |  |
| --- | --- | --- |
| **USER\_ID** | **SERVICE\_ID** | **HAS\_ACCESS** |
| **1** | **1** | **Y** |
| **1** | **2** | **N** |
|  |  |  |

1. **Convert Password Hash & insert User record in the Table:**

Use following link for sample Implementation of Hashing

<https://www.geeksforgeeks.org/sha-512-hash-in-java/>

after Hashing the password field, we need to insert it along with other user info in DB table.

Insert Query:

INSERT INTO MCB\_USER\_INFO

( USER\_NAME ,

PASSWORD\_HASH,

USER\_ROLE ,

ROLE\_DESCRIPTION ,

USER\_ACTIVE ,

CREATED\_DATE ,

UPDATED\_DATE) values('MCB\_USER1','XX\_HASHED\_PWD1','Admin1','MCB Administrator1','N',current\_timestamp,current\_timestamp);

1. **Retrieving User info records from DB and put them in Local Cache with expiry time.**

we have implemented Hashing for Password text, and deception with base64 algorithm <we are still working on Cache mechanism>

1. **Business logic to Validate Request header's user credentials with cache details before knocking the Web services.**

this can be done while final integration time with WS service & user management

1. **Error handling when validation fails.**

this can be taken care while integration time & user management