User Assistant

Design Document

**Table of Contents**

[1. Purpose 3](#_Toc305063666)

[2. Use Case 3](#_Toc305063667)

[3. Flow Chart 3](#_Toc305063668)

[4. Database Design 5](#_Toc305063669)

[5. Required Objects, Services, and Libraries 6](#_Toc305063670)

[6. Technical Specifications 8](#_Toc305063673)

[6.1 UI Layer 8](#_Toc305063674)

[6.2 Class Diagram: 10](#_Toc305063677)

[6.3 Sequence Diagram: 11](#_Toc305063678)

[6.5 Business Facade 12](#_Toc305063679)

[6.6 Business Object 13](#_Toc305063680)

[6.7 Persistence Layer 14](#_Toc305063686)

[6.8 Singleton 16](#_Toc305063687)

[7. Log, Error Message and Configuration Requirements 19](#_Toc305063688)

# Purpose

The document illustrates the low level design details of batch management module of the **User Assistant** application. The document can be used by developers to implement the following functionalities,

* The administrator can add new user details
* The administrator can get new user details
* The administrator can update user details
* The administrator can delete user details

# Flow Chart

The following are flow charts illustrate the process flow of the two use cases.

* **Add User Details**

User

Add User Details

Get User Details

Update User Details

Delete User Details

Validate the User

Display the Success message

Stop

No

Yes

Invalid credentials

CRUD Done Successfully

Add/delete/update/post-User-Role

# Database Design

|  |  |  |
| --- | --- | --- |
| **user** | **role** | **User\_role** |
| CREATE TABLE `user` (  `user\_id` int(11) NOT NULL,  `user\_code` varchar(50) NOT NULL,  `user\_phone` varchar(200) NOT NULL,  `user\_email` varchar(200) NOT NULL,  `pan\_no` varchar(50) NOT NULL,  `user\_password` varchar(200) NOT NULL,  `is\_active` tinyint(4) NOT NULL,  ) | CREATE TABLE `role` (  `role\_id` int(11) NOT NULL,  `role\_name` varchar(100) NOT NULL,  `role\_description` varchar(100) NOT NULL,  `is\_active` tinyint(4) NOT NULL DEFAULT 1,  `created\_by` bigint(20) DEFAULT 1,  `updated\_by` bigint(20) NOT NULL DEFAULT 1,  `created\_on` datetime DEFAULT NULL,  `updated\_on` datetime DEFAULT NULL  ) | CREATE TABLE `user\_role` (  `user\_role\_id` int(11) NOT NULL,  `user\_id` int(11) NOT NULL,  `role\_id` int(11) NOT NULL,  `is\_active` tinyint(4) NOT NULL,  `created\_by` bigint(20) DEFAULT NULL,  `updated\_by` bigint(20) NOT NULL,  `created\_on` datetime DEFAULT NULL,  `updated\_on` datetime DEFAULT NULL  ) ; |

# Required Objects, Services, and Libraries

* 1. Common Components

|  |  |
| --- | --- |
| **Component Name** | **Component Description** |
|  |  |
| UserNotFoundException | This is the exception class which will be used to throw exception id user not exist in system |
| RoleIdNotFoundException | This is the exception class which will be used to throw exception Role Id not exist in system |

* 1. Value Objects

|  |  |
| --- | --- |
| **Value Object Name** | **Value Object Description** |
| UserVO | This VO is used for holding the user details from front end to back end during insertion and back end to front end during retrieval. |

* 1. UI Standard

The following are the UI standards

* There should be no java script alerts used. Rather it should be messages in screen.
* There should be no horizontal scroll bar.
* All labels should be in Times new roman font.
* The error messages should in Arial font, size 12, bold and color is red.
* The success messages should be in green color, bold, font arial, size 12.
* Table header should be in Calibri font, size 12, bold and centre aligned.
* Buttons text should be aligned centre.
* There should be consistent vertical and horizontal spacing between the form controls.
* All the pages should have a casestudy footer and header similar to the login pages.
  1. Package Details

The following are the package details of the objects

|  |  |
| --- | --- |
| **Objects** | **Package details** |
| Default | com.vivriti.assignment. |
| Controller | com.vivriti.assignment.controller |
| Service | com.vivriti.assignment.service |
| Service Implement | com.vivriti.assignment.serviceimpl |
| Exception | com.vivriti.assignment.exception |
| Model | com.vivriti.assignment.model |
| DAO | com.vivriti.assignment.repository |

# 6. Technical Specifications

.



### Add User Screen:

**Description:** This screen is used by the administrator to add a new batch of medicines in the system.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Data Element** | **Control type** | **Default Values** | **Editable Field** | **Restrictions** | **Mandatory/**  **Non Mandatory** | **Data Type** | **Data Size** |
| User\_Id | Textbox | No | Yes | Numeric | Mandatory | Int | 11 |
| User\_Code | Textbox | No | YES | Numeric | Mandatory | Varchar | 50 |
| User\_phone | Textbox | No | Yes | Numeric | Mandatory | Int | 200 |
| User\_email | Textbox | No | Yes | Alpha Numeric | Mandatory | Varchar | 200 |
| Pan\_no | Textbox | No | Yes | Alpha Numeric | Mandatory | Varchar | 50 |
| User\_password | Textbox | No | yes | Alpha Numeric | Mandatory | Varchar | 200 |
| Ia\_active | Textbox | No | Yes | Numeric | Mandatory | Varchar | 4 |

On successful addition the success screen should be shown to the user with the following message “User Successfully Added in the System”

**Role Table:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Table Name** | **Column Name** | **Column Discription** | **Data Type** | **Data Size** | **Primary Key** | **Foreign Key** |
| Role\_Id | role\_id | Column is used for storing the Role-Id | Int |  | yes | - |
| Role\_Name | role\_name | Column is used for storing the Role-Name | varchar2 | 30 | - | - |
| Role\_Description | role\_description | Column is used for storing the Role-Description | varchar2 | 50 | - | - |
| Is\_Active | is\_active | Column is used for storing the Ia-Active | varchar2 | 30 | - | - |
| Created\_By | created\_by | Column is used for storing the Created-By | varchar2 | 30 | - | - |
| Updated\_By | updated\_by | Column is used for storing the Update-By | varchar2 | 30 | - | - |
| Created\_On | created\_on | Column is used for storing the Created-By | Date | - | - | - |
| Updated\_On | updated\_on | Column is used for storing the Update-on | Date | - | - | - |

**User\_Role Table:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Table Name | Column Name | Column Discription | Data Type | Data Size | Primary Key | Foreign Key |
| User\_Role\_Id | user\_role\_id | Column is used for storing the user\_role\_id | Int |  | Yes | - |
| User\_Id | user\_id | Column is used for storing the user\_id | Int |  | - | Yes |
| Role\_Id | role\_id | Column is used for storing the role\_id | Int |  | - | Yes |
| Is\_Active | is\_active | Column is used for storing the is\_active | Varchar | 50 | - | - |
| Created\_By | created\_by | Column is used for storing the created\_by | Varchar | 50 | - | - |
| Updated\_By | updated\_by | Column is used for storing the updated\_by | Varchar | 50 | - | - |
| Created\_On | created\_on | Column is used for storing the created\_on | Date | - | - | - |
| Updated\_On | updated\_on | Column is used for storing the updated\_on | Date | - |  |  |

## Class Diagram:

# Controller class:

|  |  |
| --- | --- |
| Class Name | UserController |
| Class Description | Request Controller for all request |

|  |  |
| --- | --- |
| Method Name | addUser/update User/delete user/create new user |
| Method Description | Methods are used to perform CRUD operation on UserRole and role |
| Method Input | UserVO |
| Method Output | Objects/ boolean |
| Algorithm | This invokes the add user method/ update user method  Delete user method/ Create user method |
| Validations | Not Null |
|  |  |

# Service Layer:

|  |  |
| --- | --- |
| Class Name | ServiceLayer |
| Class Description | This service is used to content business logic |

|  |  |  |
| --- | --- | --- |
| Method Name | | addDetails,GetDetails,UpdateDetails,DeleteDetails |
| Method Description | | Used for storing the roles details in the system as a new record |
| Method Input | | UserVo |
| Method Output | | Boolean |
| Validations | | NA |
| Exception | Throws SQL Exception – when any underlying exception arises. | |

# DAO Layer:

|  |  |
| --- | --- |
| Class Name | DaoLayer |
| Class Description | This dao is used for adding new batches into the system. |

|  |  |  |
| --- | --- | --- |
| Method Name | | addDetails |
| Method Description | | Used for storing the roles details in the system as a new record |
| Method Input | | Role add |
| Method Output | | Boolean |
| Validations | | NA |
| Exception | Throws SQL Exception – when any underlying exception arises. | |