



# UMS Documentation

Technical Doc & Guide

Abdul Mobeen  
MOBEENDEV@GMAIL.COM

# UMS - Documentation

## Contents

1. Contents.....	
2. Introduction & System Design .....	
a. Domain Model	
b. ERD	
c. System Architecture	
3. Web Application Screens .....	
4. APIs.....	
5. APIs Snapshots	
6. System Implementation & Tools/Technologies Used .....	
7. Setup Guide.....	

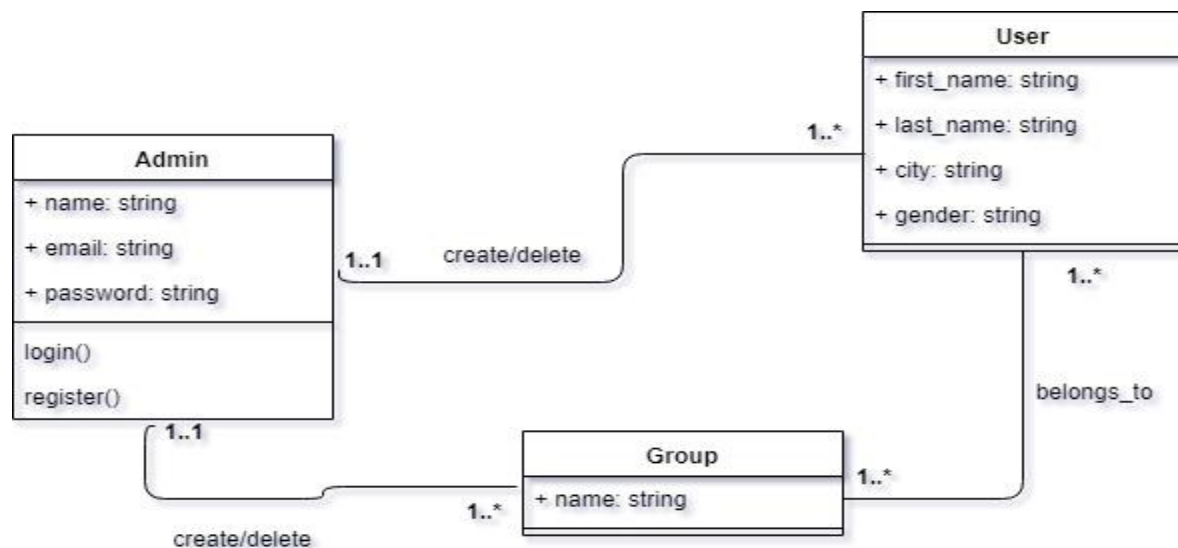
## Introduction

This is a Web Application which is a **User Management System** with the following stories:

- A person can register as Admin
- Admin can login into the system
- Admin can add users
  - a user has following attributes
    - first\_name.
    - last\_name
    - gender
    - city
- Admin can view the User
- Admin can delete users from groups.
  - When admin deletes a user, his/her joined group will be detached but user will not be removed from the DB.
- Admin can attach users to a group they aren't already part of.
- Admin can remove users from a group.
- Admin can create groups.
  - A group has 'name' attribute
- Admin can view the groups with all the user's in that group on detail screen
- Admin can delete groups when they no longer have any users/members

## UML

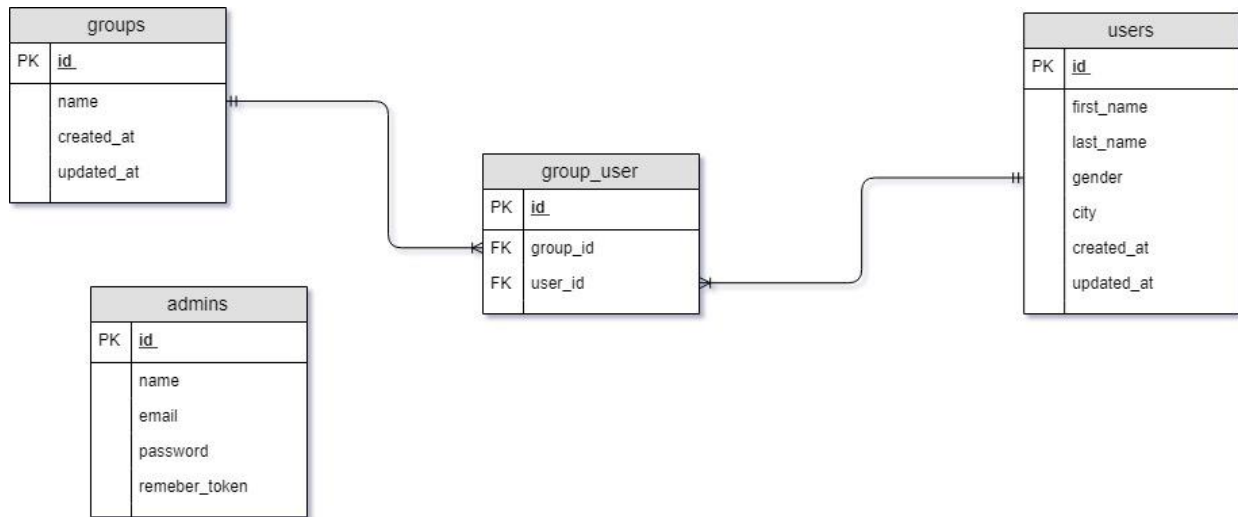
A domain model in problem solving and software engineering is a conceptual model of all the topics related to a specific problem. It describes various entities, their attributes, roles, and relationships, plus the constraints that govern the problem domain



## ERD

As this web application uses **user** and **groups** data, so for this task I have used relational database, to store and retrieve the data, figure below shows tables and relationships among the tables which are used for this application.

I have assumed that there's **many to many** relationships between **users** <-> **groups**. i.e. a user can be in multiple groups and a group can have many users.



## System Architecture

For this task, I have used three tier architecture i.e MVC or Model View Controller. I have also used the repository pattern where possible to create the abstraction for data access layer and to handle the domain objects (users, groups).

Next you can see the screen for above stories.

## Web Application Screens

### Login Screen:

The screenshot shows a web browser window with the URL `localhost/ums/public/login`. The page has a header with the text "InterNations" on the left and "Login Register" on the right. The main content area contains a "Login" form. The form has two input fields: "E-Mail Address" with the value "admin@admin.com" and "Password" with masked characters "\*\*\*\*\*". Below the password field is a blue "Login" button.

InterNations Login Register

Login

E-Mail Address admin@admin.com

Password \*\*\*\*\*

Login

### Register Screen

The screenshot shows a web browser window with the URL `localhost/ums/public/register`. The page has a header with the text "InterNations" on the left and "Login Register" on the right. The main content area contains a "Register" form. The form has four input fields: "Name", "E-Mail Address", "Password", and "Confirm Password". Below the "Confirm Password" field is a blue "Register" button.

InterNations Login Register

Register

Name

E-Mail Address

Password

Confirm Password

Register

## Dashboard

The screenshot shows a web application interface for a dashboard. The browser's address bar displays 'localhost/ums/public/admin'. On the left, a sidebar menu lists 'InterNations', 'Dashboard', 'Users', and 'Groups'. The main content area features a large header with the text 'This is your Dashboard' and a subtext 'This example is a quick exercise to illustrate how the bottom navbar works.' Below this is a 'Sample Button'.

Below the header, there are two summary cards:

- Groups**: Shows '4 Groups' with links for 'Create More' and 'Add Users to Groups'.
- Users**: Shows '6 Users' with links for 'Create More' and 'Attach Users with Groups'.

The bottom of the image shows a Windows taskbar with various application icons and a system clock indicating 3:49 PM on 4/6/2019.

## User Listing

The screenshot shows the 'User Listing' page of the web application. The browser's address bar displays 'localhost/ums/public/users'. The sidebar menu on the left has 'Users' selected. The main content area is titled 'Users List' and contains a table of user records. Each record includes a user ID, name, gender, group information, and a 'Delete' button. A pagination bar at the bottom of the table shows '1' and '2' as active page numbers.

On the right side of the table, there is an 'Add User' button. The system clock at the bottom right indicates 3:49 PM on 4/6/2019.

ID	Name	Gender	Group	Action
Abdul Mobeen8	- Doha	male	Groups ( SL-PHPBackend-V2,SL-967 )	<a href="#">Add to Group</a> <button>Delete</button>
sdfs asdfsdf	- sdfsdf	female	Groups ( SL-PHPBackend-V2 )	<a href="#">Add to Group</a> <button>Delete</button>
Abdul1 Mobeen	- Doha	male	Groups ( SL-PHPBackend-V2 )	<a href="#">Add to Group</a> <button>Delete</button>
Abdul345 Mobeen	- Doha	male	Groups ( No group assigned! )	<a href="#">Add to Group</a> <button>Delete</button>
Abdul Mobeen	- Doha	male	Groups ( No group assigned! )	<a href="#">Add to Group</a> <button>Delete</button>

## Add New User

Simple Sidebar - Start Bootstrap x +

localhost/ums/public/users/create

InterNations Toggle Menu Home Link Abdul Mobeen

Dashboard

Users

Groups

First Name Last Name

First Name Last Name

City Gender

--Select Here--

--Select Here--

Male

Female

Add Back

## Groups Listing and Add Group

Simple Sidebar - Start Bootstrap x +

localhost/ums/public/groups

InterNations Toggle Menu Home Link Abdul Mobeen

Dashboard

Users

Groups

Groups List

SL-PHPBackend-V2  
Total User :3 | View details Delete

SL-967  
Total User :1 | View details Delete

hello  
Total User :0 | View details Delete

hello  
Total User :0 | View details Delete

Group Name

Group Name

Add

Windows Taskbar: 3:52 PM 4/6/2019

## View Group Detail

The screenshot shows a web application interface for viewing group details. The browser address bar indicates the URL is `localhost/ums/public/groups/8`. The application has a sidebar menu with options: InterNations, Dashboard, Users, and Groups. The main content area is titled 'Group Details' and features a group icon of three people. The group name is 'SL-PHPBackend-V2' and it has 3 total users. Below this, a 'Users List' table displays the following data:

#	First Name	Last Name	Gender	City	Remove from Group
1	Abdul	Mobeen8	male	Doha	<a href="#">(X) Remove</a>
1	sdfs	asdfsdf	female	sdfsdf	<a href="#">(X) Remove</a>
1	Abdul1	Mobeen	male	Doha	<a href="#">(X) Remove</a>

The Windows taskbar at the bottom shows the time as 3:53 PM on 4/6/2019.

## APIs

I have only created following APIs just for the demonstration

- Admin Login
- Admin Registration
- Get all users with their joined groups
- Get all groups with all the user's in that group



## APIs snapshots

### Admin Registration API

POST ▼ http://localhost/ums/public/api/auth/register Send ▼

Params Authorization Headers **Body** Pre-request Script Tests Cookies Code

☐ none ☒ form-data ☐ x-www-form-urlencoded ☐ raw ☐ binary

KEY	VALUE	DESCRIPTION	..
Key	Value	Description	

**Body** Cookies Headers (10) Test Results Status: 404 Not Found Time: 1119 ms Size: 522 B Save

Pretty Raw Preview JSON ≡

```
1 {
2   "success": false,
3   "message": "Validation Errors",
4   "data": {
5     "email": [
6       "The email field is required."
7     ],
8     "name": [
9       "The name field is required."
10    ],
11    "password": [
12      "The password field is required."
13    ]
14  }
15 }
```

POST ▼ http://localhost/ums/public/api/auth/register

	KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/>	email	admin@admin.com	
<input checked="" type="checkbox"/>	password	12345678	
<input checked="" type="checkbox"/>	name	mobeen	
	Key	Value	Description

**Body** Cookies Headers (10) Test Results Status: 404 Not Found Time: 1113 ms Size: 441

Pretty Raw Preview JSON ≡

```
1 {
2   "success": false,
3   "message": "Validation Errors",
4   "data": {
5     "email": [
6       "The email has already been taken."
7     ]
8   }
9 }
```



Get all users with their joined groups

```
GET http://localhost/ums/public/api/users?page=1 with pagination Send

Pretty Raw Preview JSON

{
  "data": [
    {
      "id": 8,
      "first_name": "Abdul",
      "last_name": "Mobeen8",
      "city": "Doha",
      "gender": "male",
      "created_at": "2019-04-02 18:11:37",
      "updated_at": "2019-04-02 18:11:37",
      "groups": [
        {
          "id": 8,
          "name": "SL-PHPBackend-V2",
          "created_at": "2019-04-06 10:42:42",
          "updated_at": "2019-04-06 10:42:42",
          "pivot": {
            "user_id": 8,
            "group_id": 8
          }
        }
      ]
    },
    {
      "id": 9,
      "name": "SL-967",
      "created_at": "2019-04-06 10:42:47",
      "updated_at": "2019-04-06 10:42:47",
      "pivot": {
        "user_id": 8,
        "group_id": 9
      }
    }
  ]
}
```

```
JSON
{
  "success": true,
  "message": "Users retrieved successfully.",
  "data": {
    "current_page": 1,
    "data": [
      {
        "id": 8,
        "first_name": "Abdul",
        "last_name": "Mobeen8",
        "city": "Doha",
        "gender": "male",
        "created_at": "2019-04-02 18:11:37",
        "updated_at": "2019-04-02 18:11:37",
        "groups": [
          {
            "id": 8,
            "name": "SL-PHPBackend-V2",
            "created_at": "2019-04-06 10:42:42",
            "updated_at": "2019-04-06 10:42:42",
            "pivot": {
              "user_id": 8,
              "group_id": 8
            }
          }
        ]
      },
      {
        "id": 9,
        "name": "SL-967",
        "created_at": "2019-04-06 10:42:47",
        "updated_at": "2019-04-06 10:42:47",
        "pivot": {
          "user_id": 8,
          "group_id": 9
        }
      }
    ]
  },
  "from": 1,
  "last_page": 2,
  "next_page_url": "http://localhost/ums/public/api/users?page=2",
  "path": "http://localhost/ums/public/api/users",
  "per_page": 5,
  "prev_page_url": null,
  "to": 5
}
```

Get all groups with all the user's in that group

GET <http://localhost/ums/public/api/groups?page=1> Send

Pretty Raw Preview JSON

```
1 {
2   "success": true,
3   "message": "Groups retrieved successfully.",
4   "data": {
5     "current_page": 1,
6     "data": [
7       {
8         "id": 8,
9         "name": "SL-PHPBackend-V2",
10        "created_at": "2019-04-06 10:42:42",
11        "updated_at": "2019-04-06 10:42:42",
12        "users": [
13          {
14            "id": 8,
15            "first_name": "Abdul",
16            "last_name": "Mobeen8",
17            "city": "Doha",
18            "gender": "male",
19            "created_at": "2019-04-02 18:11:37",
20            "updated_at": "2019-04-02 18:11:37",
21            "pivot": {
22              "group_id": 8,
23              "user_id": 8
24            }
25          },
26          {
27            "id": 10,
28            "first_name": "sdfs",
```

Not secure | jsonviewer.stack.hu

Viewer Text

JSON

- success: true
- message: "Groups retrieved successfully."
- data
  - current\_page: 1
  - data
    - 0
      - id: 8
      - name: "SL-PHPBackend-V2"
      - created\_at: "2019-04-06 10:42:42"
      - updated\_at: "2019-04-06 10:42:42"
      - users
        - 0
          - id: 8
          - first\_name: "Abdul"
          - last\_name: "Mobeen8"
          - city: "Doha"
          - gender: "male"
          - created\_at: "2019-04-02 18:11:37"
          - updated\_at: "2019-04-02 18:11:37"
          - pivot
            - group\_id: 8
            - user\_id: 8
  - 1
  - 2
- from: 1
- last\_page: 2
- next\_page\_url: "http://localhost/ums/public/api/groups?page=2"
- path: "http://localhost/ums/public/api/groups"
- per\_page: 3
- prev\_page\_url: null
- to: 3
- total: 4

Sample response objects for Users & Groups API are attached as well.

I have not applied the token check on the GET API calls for ease of testing the APIs.

### System Implementation & Tools Technologies Uses

The system is developed using PHP, MySQL as main technologies and Apache is used as the web server to host this web application.

Other details are as follows

Laravel Framework, Bootstrap, Xdebug, Composer, XAMPP

Postman, Workbench, Chrome Browser

### Setup Guide

Extract the Zip/Rar File.

Place the “ums” folder into XAMPP (htdocs) or WAMP (www) respective directory.

Open the Project in any code editor.

open the .env file and provide the DB details if yours are different e.g.

DB\_CONNECTION=mysql

DB\_HOST=127.0.0.1

DB\_PORT=3306

DB\_DATABASE=ums // this should remain same

DB\_USERNAME=root

DB\_PASSWORD=

Import the **ums\_db.sql** file, it will create the database “ums” in your MySQL along with the required tables.

And hit this url: <http://localhost/ums/public/>

And you will see the login screen. As you are using it for the first time, just create an admin user using the Register Link and login.

If there is any issue in setting up the application, please let me know.

**Thanks.**