11/28/2018

FITNESS FACTORY

IDC30BT DOCUMENTATION

UML Documentation and data scripts

Contents

[DOCUMENTATION (UML) 2](#_Toc531171923)

[Use case diagram 2](#_Toc531171924)

[Domain class diagram 3](#_Toc531171925)

[Full Use case Description 4](#_Toc531171926)

[Design class diagram 9](#_Toc531171927)

[System sequence diagram 13](#_Toc531171928)

[Multilayer Sequence Diagram 16](#_Toc531171929)

[Package diagram 18](#_Toc531171930)

[CRC Cards 20](#_Toc531171931)

[Data Script 23](#_Toc531171932)

[ERD 27](#_Toc531171933)

# DOCUMENTATION (UML)

## Use case diagram



## Domain class diagram



## Full Use case Description

|  |  |  |
| --- | --- | --- |
| **USE CASE NAME:** | Register member | |
| **SCENARIO:** | The user will enter his/her personal information in the system in order to register then the system will check if the member exist or not if the member already exist the message will be sent to the user noting that the member already exist and if the member don’t exist then the information will be saved in a database. | |
| **TRIGGERRING EVENT:** | The member wants to have an account | |
| **BRIEF DESCRIPTION:** | The member must be able to get access to system to insert his/her personal details to the system | |
| **ACTORS:** | Member | |
| **RELATED USE CASE:** | None | |
| **STAKEHOLDERS:** | Member, Admin | |
| **PRECONDTION:** | The Member must not be existing member from the database | |
| **POSTCONDITION:** | The message will be displayed to the screen and the member’s information will be saved in the database. | |
| **FLOW OF ACTIVITY:** | ACTOR | SYSTEM |
| 1. Click on registration tab   2. the member insert his/her information into the fields and clicks register button | 1.1 The system display the registration page  2.1the system will check if all fields have data  2.2 check if member doesn’t exist |
| **EXCEPTION CONDITION:** | The member must not be the existing member on the system | |

|  |  |  |
| --- | --- | --- |
| **USE CASE NAME:** | Book session | |
| **SCENARIO:** | The member will login into the system to make bookings for the session they would like, but the member is not to book two sessions for the same date. | |
| **TRIGGERRING EVENT:** | The member wants to book a session | |
| **BRIEF DESCRIPTION:** | The member must be able to get access to system and make bookings | |
| **ACTORS:** | Member | |
| **RELATED USE CASE:** | None | |
| **STAKEHOLDERS:** | Member, Admin | |
| **PRECONDTION:** | NONE | |
| **POSTCONDITION:** | The message will be displayed to the screen and the member’s booking will be saved in the database. | |
| **FLOW OF ACTIVITY:** | ACTOR | SYSTEM |
| 1. Click on booking tab   2. the member selects the session and the date she wants to do the session | 1.1 The system display the booking page  2.1the system will check if the member hasn’t booked the selected date and display the appropriate message |
| **EXCEPTION CONDITION:** |  | |

|  |  |  |
| --- | --- | --- |
| **USE CASE NAME:** | delete member | |
| **SCENARIO:** | The admin will login into the system to delete an existing member. | |
| **TRIGGERRING EVENT:** | The member don’t want an account anymore | |
| **BRIEF DESCRIPTION:** | The admin must be able to get access to system so that he can delete an existing member | |
| **ACTORS:** | Admin | |
| **RELATED USE CASE:** | None | |
| **STAKEHOLDERS:** | Member, Admin | |
| **PRECONDTION:** | The Member must be existing member from the database | |
| **POSTCONDITION:** | The message will be displayed to the screen and the member’s information will removed in the database. | |
| **FLOW OF ACTIVITY:** | ACTOR | SYSTEM |
| 1. Click on deleteMember tab   2. the admin insert clicks delete button next to the name of the member | 1.1 The system display the deleteMember page  2.1the system will display the message and remove data from the database. |
| **EXCEPTION CONDITION:** |  | |

|  |  |  |
| --- | --- | --- |
| **USE CASE NAME:** | Update Profile | |
| **SCENARIO:** | The user will view his/her personal profile and select update tab in order to update personal information that he/she would like to change. | |
| **TRIGGERRING EVENT:** | The member wants to update personal details | |
| **BRIEF DESCRIPTION:** | The member must be able to get access to system to view and update personal information | |
| **ACTORS:** | Member | |
| **RELATED USE CASE:** | None | |
| **STAKEHOLDERS:** | Member, Admin | |
| **PRECONDTION:** | The Member profile must exist already | |
| **POSTCONDITION:** | The message will be displayed on the form after new information is saved | |
| **FLOW OF ACTIVITY:** | ACTOR | SYSTEM |
| 1. Click on view profile tab   2.The member updates information and press update button | 1.1 The system will display profile details  2.1 System allows you to edit profile information  2.2 display the message and changes data from the database |
| **EXCEPTION CONDITION:** | Must be an existing member on the system | |

|  |  |  |
| --- | --- | --- |
| **USE CASE NAME:** | View booking | |
| **SCENARIO:** | The user will access the system in order to see bookings made and the admin is able to view all the bookings made by all members. | |
| **TRIGGERRING EVENT:** | The admin wants to view the available bookings | |
| **BRIEF DESCRIPTION:** | To see the total number of bookings available | |
| **ACTORS:** | Member and Admin | |
| **RELATED USE CASE:** | None | |
| **STAKEHOLDERS:** | Member, Admin | |
| **PRECONDTION:** | none | |
| **POSTCONDITION:** | Available bookings will be displayed on the screen | |
| **FLOW OF ACTIVITY:** | ACTOR | SYSTEM |
| 1. Click on view bookings tab | * 1. The system checks who the user is   1.2 Display bookings according to the user |
| **EXCEPTION CONDITION:** | None | |

## Design class diagram











## System sequence diagram







## Multilayer Sequence Diagram









## Package diagram









## CRC Cards

**Register member**

|  |  |
| --- | --- |
| <<controller>>  member\_controller | |
| Addmember | AddMember  Member |

|  |  |
| --- | --- |
| AddMember | |
| Addmember | Member |

|  |  |
| --- | --- |
| Member | |
| Addmember |  |

**Book session**

|  |  |
| --- | --- |
| <<controller>>  member\_controller | |
| makeBooking() | booking  makeBooking |

|  |  |
| --- | --- |
| booking | |
| makeBooking(?,?,?,?) |  |

|  |  |
| --- | --- |
| makeBooking | |
| Show message |  |

**Delete member**

|  |  |
| --- | --- |
| <<controller>>  member\_controller | |
| deleteMember() | Admin  deleteMember |

|  |  |
| --- | --- |
| Admin | |
| deleteMember(username) |  |

|  |  |
| --- | --- |
| deleteMember | |
| Show message |  |

# SYSTEM INSTALLATION

## HOW TO INSTALL THE SYSTEM

1. Open NetBeans IDE, then click on file, open project and choose the project.

2. Open the IIS and stop the manager sever

3. Open the httpd.conf file and replace the DocumentRoot and Directory with the directory where your project is

4. Save all the changes to the httpd.conf file

5. Open the php.ini file and replace the session.save\_path = "(project Directory)\tmp"

6. Open the XAMP and start Apache

7. Open SQLyog and restore the database

## ERD



## Data Script

/\*

SQLyog Community Edition- MySQL GUI v5.22a

Host - 5.1.30-community : Database - fitness

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Server version : 5.1.30-community

\*/

/\*!40101 SET NAMES utf8 \*/;

/\*!40101 SET SQL\_MODE=''\*/;

create database if not exists `fitness`;

USE `fitness`;

/\*!40014 SET @OLD\_FOREIGN\_KEY\_CHECKS=@@FOREIGN\_KEY\_CHECKS, FOREIGN\_KEY\_CHECKS=0 \*/;

/\*!40101 SET @OLD\_SQL\_MODE=@@SQL\_MODE, SQL\_MODE='NO\_AUTO\_VALUE\_ON\_ZERO' \*/;

/\*Table structure for table `tblbooking` \*/

DROP TABLE IF EXISTS `tblbooking`;

CREATE TABLE `tblbooking` (

`bk\_id` int(5) NOT NULL AUTO\_INCREMENT,

`sess\_id` int(5) NOT NULL,

`mem\_id` int(5) NOT NULL,

`book\_date` date DEFAULT NULL,

PRIMARY KEY (`bk\_id`,`sess\_id`,`mem\_id`),

UNIQUE KEY `bk\_id` (`bk\_id`),

KEY `FK\_tblbooking` (`mem\_id`),

KEY `FK\_tblbooking2` (`sess\_id`),

CONSTRAINT `FK\_tblbooking2` FOREIGN KEY (`sess\_id`) REFERENCES `tblsession` (`sess\_id`)

) ENGINE=InnoDB AUTO\_INCREMENT=5 DEFAULT CHARSET=latin1;

/\*Data for the table `tblbooking` \*/

insert into `tblbooking`(`bk\_id`,`sess\_id`,`mem\_id`,`book\_date`) values (1,1,11,'2018-02-05'),(2,2,12,'2017-05-09'),(3,3,13,'2015-02-05'),(4,4,14,'2015-09-06');

/\*Table structure for table `tblsession` \*/

DROP TABLE IF EXISTS `tblsession`;

CREATE TABLE `tblsession` (

`sess\_id` int(5) NOT NULL AUTO\_INCREMENT,

`sess\_name` varchar(30) DEFAULT NULL,

`sess\_cost` decimal(30,0) DEFAULT NULL,

PRIMARY KEY (`sess\_id`)

) ENGINE=InnoDB AUTO\_INCREMENT=6 DEFAULT CHARSET=latin1;

/\*Data for the table `tblsession` \*/

insert into `tblsession`(`sess\_id`,`sess\_name`,`sess\_cost`) values (1,'Boxing','1000'),(2,'Yga','200'),(3,'Swimming','500'),(4,'abs','600'),(5,'Bicept','600');

/\*Table structure for table `tbluser` \*/

DROP TABLE IF EXISTS `tbluser`;

CREATE TABLE `tbluser` (

`Fname` varchar(20) DEFAULT NULL,

`surname` varchar(20) DEFAULT NULL,

`gender` varchar(10) DEFAULT NULL,

`initial` varchar(5) DEFAULT NULL,

`memberID` varchar(13) DEFAULT NULL,

`address` varchar(30) DEFAULT NULL,

`phoneNo` varchar(10) DEFAULT NULL,

`email` varchar(30) DEFAULT NULL,

`user\_name` varchar(20) NOT NULL,

`user\_role` varchar(20) DEFAULT NULL,

`password` varchar(20) DEFAULT NULL,

PRIMARY KEY (`user\_name`),

KEY `FK\_tblmember` (`user\_name`)

) ENGINE=InnoDB DEFAULT CHARSET=latin1;

/\*Data for the table `tbluser` \*/

insert into `tbluser`(`Fname`,`surname`,`gender`,`initial`,`memberID`,`address`,`phoneNo`,`email`,`user\_name`,`user\_role`,`password`) values ('collin','malulka','Male','nkm','9815549747123','home street code','0710065321','nkc@gmail.com','member1','ADMIN','ADMIN'),('Chris','makhondo','Male','jk','9612215714078','bhella street ','0710058963','mk@gmail.com','member3',NULL,NULL),('tiyani','brown','Female','gt','961221571445','twala street ','0714458963','tiyi@gmail.com','member4',NULL,NULL),('collins','h6yty','Male','h6thy','42','tghyh','071000545','D@GMAIL.COM','member5',NULL,NULL),('MALELEKE','COLLINS','Male','WV','12','HOME','0710065985','SO@GMAIL.COM','member6',NULL,NULL);

/\* Procedure structure for procedure `DeleteMember` \*/

/\*!50003 DROP PROCEDURE IF EXISTS `DeleteMember` \*/;

DELIMITER $$

/\*!50003 CREATE DEFINER=`root`@`localhost` PROCEDURE `DeleteMember`(p\_id INT(5))

BEGIN

DELETE FROM tblbooking

WHERE mem\_id = p\_id;

DELETE FROM tblmember

WHERE id = p\_id;

END \*/$$

DELIMITER ;

/\*!40101 SET SQL\_MODE=@OLD\_SQL\_MODE \*/;

/\*!40014 SET FOREIGN\_KEY\_CHECKS=@OLD\_FOREIGN\_KEY\_CHECKS \*/;