

**<<Yoga center>>**

**Software Design Document**

**Table of Contents**

[I. Overview 3](#_Toc69802609)

[1. Code Packages 3](#_Toc69802610)

[2. Database Schema 3](#_Toc69802611)

[II. Code Designs 4](#_Toc69802612)

[1. <Feature/Function Name1> 4](#_Toc69802613)

[a. Class Diagram 4](#_Toc69802614)

[b. Class Specifications 4](#_Toc69802615)

[c. Sequence Diagram(s) 4](#_Toc69802616)

[d. Database queries 5](#_Toc69802617)

[2. <Feature/Function Name2> 5](#_Toc69802618)

[III. Database Tables 5](#_Toc69802619)

[1. <Table name 1> 5](#_Toc69802620)

[2. <Table name 2…> 5](#_Toc69802621)

# I. Overview

## 1. Code Packages/Namespaces

*[Provide the package diagram for each sub-system. The content of this section including the overall package diagram, the explanation, package and class naming conventions in each package. Please see the sample and description table format below – following Java project naming convention]*

A diagram of a computer program

Description automatically generated

***Package descriptions & package class naming conventions***

|  |  |  |
| --- | --- | --- |
| **No** | **Package** | **Description** |
| *01* | DTO | * Contains data fields and simple getter and setter methods to access and manipulate the data for all object. * Name of an object, using camel case and append it with DTO ex: trainerDTO, lopHocDTO, … |
| *02* | *DAO* | * Classes that define the CRUD (Create, Read, Update, Delete) operations that can be performed on the data source. These interfaces or classes provide methods to insert, retrieve, update, and delete data objects. Used to abstract and encapsulate the access to a data source, such as a database or a web service. * Name of an object, using camel case and append it with DAO ex: trainerDAO, lopHocDAO, … |
| *03* | *Controller* | * Handles incoming requests and manages the flow of the application. It is responsible for receiving requests from clients, processing them, and generating appropriate responses. * Name of an object, using camel case and append it with Controller ex: trainerController, lopHocController, … |
|  | *Util* | * Contain configuration of the web, constant , public class contains public function like convert date and class to connect database. * Name of the purpose append Util ex: MailUtil, DataUtil,… |
|  | *Filter* | * Intercepts inappropriate requests and responses between the client and the servlets. * Name of filter purpose ex: authentication,… |
|  | *Paypal* | * Contain Paypal method for purchase * PaypalServices |
|  | *EvenListener* | * Contain initialize method when web application boost * Scope of the method appen Listener ex: AppContextListener |

## 2. Database Schema

*[Provide the tables relationship like example below – following MySQL database naming convention]*

nuA computer screen shot of a computer

Description automatically generated

***Table descriptions & package class naming conventions are as below***

|  |  |  |
| --- | --- | --- |
| **No** | **Table** | **Description** |
| *01* | lopHoc | *- Contain all lopHoc field*  *- Primary keys:* [maLopHoc] NVARCHAR(10) primary key  *- Foreign keys:*  + CONSTRAINT fk\_loaiLopHoc\_lopHoc FOREIGN KEY([maLoaiLopHoc]) REFERENCES loaiLopHoc([maLoaiLopHoc]),  + CONSTRAINT fk\_maRoom\_lopHoc FOREIGN KEY([maRoom]) REFERENCES room(maRoom) |
| *02* | hocVien | *- Contain all hocVien field*  *- Primary keys:* [maHV] NVARCHAR(10) primary key |
| *03* | Trainer | *- Contain all Trainer field*  *- Primary keys:* [maTrainer] NVARCHAR(10) primary key |
| *04* | ScheduleHV | *- Contain all* ScheduleHV *field*  *- Primary keys:* (maLopHoc,maHV,ngayHoc)  *- Foreign keys:*  + CONSTRAINT fk\_maSlot\_ScheduleHV FOREIGN KEY([maSlot]) REFERENCES slot(maSlot)  + Constraint fk\_maLopHoc\_ScheduleHV foreign key([maLopHoc]) references [lopHoc]([maLopHoc])  + constraint fk\_maHocVien\_ScheduleHV foreign key([maHV]) references [hocVien]([maHV]) |

# II. Code Designs

## 1. Create Class

### a. Class Diagram

A diagram of a computer program

Description automatically generated

### b. Sequence Diagram

A diagram of a project

Description automatically generated

## 2. <Feature/Function Name2>

…

# III. Database Tables

## 1. <Table name 1>

*[Give some lines about the table here>>*

*[Table fields, in the form of table format as below]*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| 1 | Field name1 |  |  |  |  |  |  |
| 2 | Field name2 |  |  |  |  |  |  |

## 2. <Table name 2…>