Dance Registration System

Requirements Definitions Report

Group Number: 2

CPSC 488 Section 01

**TABLE OF CONTENTS**

**1 INTRODUCTION………………………..……………………………………………………4**

* 1. DESCRIPTION……………………………………………………………………………….4
  2. BROAD OBJECTIVES………………………………………………………………………4

*1.2.1 Functioning UI for Registering…………………………………………………………..*4

*1.2.2 Scheduling Functionalities………………………………………………………………..*4

*1.2.3 Parent-Child Visibility…………………………………………………………………….*4

**2 SCOPE……………………….………………………………………………………………....4**

2.1 PRODUCTS………………………………...………………………………………………...4

*2.1.1 User Interface…..*…………………..…………………………………………......*.*...4

*2.1.2 Database…...….*……………..……………………………………………………....4

2.2 ITEMS ADDRESSED..………..…...…………………………………………………………5

*2.2.1 Database Backup*…………………..…………………………………………......*.*...5

*2.2.2 Exporting Files…………………..*………..………………………………………......5

*2.2.3 Registering Users……………………………*…………………………………………5

*2.2.4 Assigning Students to Classes…………………………………………………………..…*5

*2.2.5 Removing Activities and Students…………………………………………………………5*

*2.2.6 Track Payments……………………………………………………………………………...5*

2.3 RECOMMENDED FEATURES…………………………………………………………..…5

*2.3.1 Saving Logs of Deactivated Users…………………………………………………….….5*

*2.3.2 Calculating Class Charges….*……..……………………………………………......*.*5

*2.3.3 Parent Payment*…………………………………………………………………...…*6*

*2.3.4 Child Attendance*……………………………………………………………………*.6*

2.4 TERMS

*2.4.1 Student/Child/Children…………………………………………………………………..…6*

*2.4.2 Parent………………………………………………………………………………………..6*

*2.4.3 Class Charges………………………………………………………………………………..6*

*2.4.4 Exporting Files……………………………………………………………………………….6*

*2.4.5Activities…………………………………………………………………………………….…6*

2.5 ASSUMPTIONS………………………………………………………………………………6

**3 SPECIFICATION REQUIRMENTS ………………………………………………………..7**

3.1 DIAGRAMS………………………...………………………………………………...............7

3.2 SYSTEM REQUIRMENTS…………………………………………………………………..7

3.4 SECURITY……………………………………………………………………………………7

*3.4.1 Privileges……………………………………………………………………………………..7*

*3.4.2 Excel Uploads………………………………………………………………………………..7*

3.5 PERFORMANCE SPECIFICATIONS……………………………………………………….8

3.6 REQUIRED CLASSES……………………………………………………………………….8

*3.6.1 Configuration Classes……………………………………………………………………….8*

*3.6.2 Controller Classes…………………………………………………………………………...8*

*3.6.3 Domain Classes………………………………………………………………………………9*

*3.6.4 Model Classes………………………………………………………………………………10*

*3.6.5 Repository Classes………………………………………………………………………11*

*3.6.6 Security Classes…………………………………………………………………………..12*

**1 INTRODUCTION**

* 1. **Description**

The Dance Registration System is a beginning software that is targeted towards dance studios and supplying a proficient system allowing registration of users, tracking users, payments, and activities. The system provides a button-based UI to ensure it is user friendly. The process allows for five different types of users: Admins, Managers, Instructors, Parents, and Children. All users have respective dashboards allowing for different functionalities based on the user role.

* 1. **Broad Objectives**
     1. Functioning UI for Registration

The UI allows the manager users to create/register new users. They can then assign them their user-roles.

* + 1. Scheduling Functionalities

Once a child is enrolled in the studio, the Manager can then assign them to specific classes/activities. They can determine the type of class, class level, and studio where the child will participate in the class.

* + 1. Parent-Child Visibilities

The parent has the capabilities to see all of their child/children’s activities. They can see the classes they are enrolled in and the specific time and place said class will occur.

1. **SCOPE**
   1. **Products**
      1. User Interface

The UI runs on the localhost server. It allows all the functionalities to be performed.

* + 1. Database

The database is populated by loading Excel files into the program, through the UI. Once populated, the data stored can be displayed on the UI.

* 1. **Items Addressed**
     1. Database Backup

The database backup can be performed on the UI. This can only be preformed by the administrative users.

* + 1. Exporting Files

There are five excel files to be loaded into the program. Each of the files can also be exported. They can be found in: /fall2023registrationsystem/ProgramDocuments/Export Files

* + 1. Registering Users

Managers can register all of the users. At the time of registration, they will assign the user-role, in turn granting the new user specific privileges.

* + 1. Assigning Students to Classes

After the manager registers the new children users, they can then assign them to the requested classes. They will assign the class, class level, and studio where the class will be taught.

* + 1. Removing Activities and Students

Managers have the privileges to remove students from activites and to remove activities from the studio.

* + 1. Track Payments

The managers can view all of the balances for every parent. The parent can view their balance and all of their transaction history.

* 1. **Recommended Features**
     1. Saving Logs of Deactivated Users

The database should save a log of all the users once enrolled in classes at the studios.

* + 1. Calculating Class Charges

The balance should be calculated by accounting for all the classes he children attended per month, whilst factoring in discounts.

* + 1. Parent Payment

The parent users should be able to make payments on their respective accounts.

* + 1. Child Attendance

The instructors should be able to take attendance of every class they teach and that attendance should be logged in the database.

* 1. **Terms**
     1. Student/Child/Children

The users who will be enrolled in classes.

* + 1. Parent

Parent users of the Children.

* + 1. Class Charges

The cost per taking a class. These differ based on level and discounts are offered for multi-classes and siblings.

* + 1. Exporting Files

For each excel file uploaded, they can also be exported from the database.

* + 1. Activities/Classes

The courses offered at the studios for students to be enrolled in and instructors to teach.

* 1. **Assumptions**

The program is assumed to be working up to date.

1. **SPECIFICATION REQUIRMENTS** 
   1. **Diagram**

**Manager Interactions**

A diagram of a manager

Description automatically generated

Postconditions: parent gains insight into their child’s scheduled activities

* 1. **System Requirements**
* Intel® Core™ Processors 8th Generation or newer
* Microsoft Windows 11 or Windows Server 2022 or newer
* Eclipse IDE for Enterprise Java and Web Developers v2023-09 R
* MySQL Workbench v8.0.34
  1. **Security**
     1. Privileges

Users are assigned to user-roles when they are created. These roles determine what information they are granted access to view and or modify. Managers have the most capabilities.

* + 1. Excel Uploads

The program checks the excel files prior to reading them into the database. The first value in the first cell, A1, must match the hard-coded value defined in the program. If matched, then the program will proceed to read in the data.

* 1. **Performance Specifications**

At this time, the program does not require much optimization. One major speed/time optimization is found when loading the larger excel files. These will take approximately 2-3 seconds longer than smaller files. The charges excel file is the smallest, for reference.

* 1. **Required Classes**
     1. Configuration Classes

AdminConfig.java

Configures the initial setup for admin and users by checking and creating an admin if none exists.  It utilizes the admin and user repositories to interact with the database and ensures presence of both Admin and User entities with predefined email and password.

ContextConfig.java

Configures and provides beans for Apache HTTP client and request configuration, allowing for customized settings for HTTP communication within the system.

WebConfig.java

Enables Cross-Origin Resource Sharing (CORS) by specifying allowed origins, methods, headers, and allowing credentials, ensuring secure communication with client origin. 

WebSecurityConfig.java

Configures security settings that define authentication providers, password encoding, and role-based access controls.  Specifies access permissions for URLs based on user roles, handles login/logout functionality, and manages session creation policies.

* + 1. Controller Classes

AccountController.java

Handles user account-related operations, including viewing account details, updating user information, and managing password changes with validation.

AdminController.java

Manages administrative functionalities, including displaying user role breakdowns, creating users with various roles, handling user deletions, resetting passwords, and performing database backups.

AssignChildController.java

Manages assignment of children to specific activities, handles the display of assignment forms, processing forms, and submissions, as well as updates the child assignment details.

DisplayController.java

Manages display of various information, including activities, children, and parent balances, as well as displaying information in regard to the admin, managers, and instructors.

ErrorController.java

Handles custom 403 errors, providing mapping to specific HTML page when access to resource is forbidden.

ExportController.java

Handles export requests, providing methods to export the Excel files for admin, instructor, charges, parent/child, and activity data.

LogonController.java

Handles user registration, login, and redirects user to their respective dashboard depending on their roles.  Manages password change for first-time logins and provides additional functionalities such as determining user roles and checking email existence in specific tables

ManagerController.java

Handles manager dashboard, creating and managing instructors, parents, and children, exporting data, selecting parents and children, and handling class-related actions such as deleting and updating activities.

ParentController.java

Handles parent dashboard, selecting child to display activities, and displaying activities for selected child.

TransactionController.java

Manages payment-related operations, including displaying payment form, processing cash and check payments, and updating parent and transaction details accordingly.

UploadController.java

Manages uploading and processing of Excel files containing data related to activities, admins, managers, charges, instructors, and parent/children, as well as populating the corresponding repositories and database tables.

* + 1. Domain Classes

Activities.java

Entity class storing information about activities, including but not limited to activityID, start/end times, location, and instructor.

Admin.java

Entity class storing information about admin, including but not limited to adminID, first/last name, email, and password.

Charges.java

Entity class storing information about charges, including but not limited to classLevel, flat rates, and multi-child discounts.

ChildId.java

Defines a composite key for specific child information, consisting of the childID, activityID, and activityLevel.

Children.java

Entity class storing information about children, including but not limited to childID, age, parentID, and contains a mapping to the composite key defined in the “ChildId” class.

Instructor.java

Entity class storing information about instructors, including but not limited to instructorID, first/last name, classes taught, and password.

Location.java

Entity class storing information about studio locations, including but not limited to studioID, address, and phone number.

Manager.java

Entity class storing information about managers, including but not limited to managerID, first/last name, email, and password.

Parent.java

Entity class storing information about parents, including but not limited to parentID, secondary contact, and account balance.

Transactions.java

Entity class storing information about transactions, including but not limited to transactionID, amount, and payment type.

* + 1. Model Classes

CustomUserDetails.java

Implements *UserDetails* interface to provide custom user details, including authorities and information associated with the *User* class.

User.java

Entity class storing information about users, including but not limited to id, email, password, and first/last name.

* + 1. Repository Classes

ActivitiesRepository.java

Repository for *Activities* entity, providing methods to interact with the database.

AdminRepository.java

Repository for *Admin* entity, providing methods to interact with the database.

ChargesRepository.java

Repository for *Charges* entity, providing methods to interact with the database.

ChildRepository.java

Repository for *Children* entity, providing methods to interact with the database

InstructorRepository.java

Repository for *Instructor* entity, providing methods to interact with the database.

LocationRepository.java

Repository for *Location* entity, providing methods to interact with the database.

ManagerRepository.java

Repository for *Manager* entity, providing methods to interact with the database.

MapCustomUserRepository.java

Custom implementation of the *UserRepository* to store and retrieve user information.

ParentRepository.java

Repository for *Parent* entity, providing methods to interact with the database.

TransactionRepository.java

Repository for *Transaction* entity, providing methods to interact with the database.

UserRepository.java

Repository for *User* entity, providing methods to interact with the database

* + 1. Security Classes

PasswordConstraintValidator.java

Custom implementation of the *ConstraintValidator* to validate password constraints using the Passay library.

ValidPassword.java

Marks fields or types that need to be validated against specific password constraints.  Associated with the *PasswordConstraintValidator* class for specifying the rules.