UTCN
Computer Science Department
Software Design 2019

ASSIGNMENT A1

==========

1. Objective

The main objective of this assignment is to allow students to become familiar with *Layers* architectural pattern. Furthermore, we aim to become familiar with the DAO pattern for database access and Transaction Script / Table Module for domain modeling.

2. Application Description

Use JAVA/C# to design and implement an application for an online Parking Request System. The main goal of the application is to provide clear and transparent way of requesting and assigning parking spots in the parking lots of Cluj-Napoca. Each citizen can make a new request for a parking spot. The request can be done only for one car that the citizen owns, but he can select multiple parking lots that would suit him good. Each parking lot has a certain number of parking spots. If the citizen has multiple cars, he must file multiple parking requests.

The application should have two types of users: a regular user represented by the citizen and an administrator user, represented by the city clerk responsible for assigning the free parking spots. Both kinds of uses have to provide an email and a password in order to access the application.

The citizen (regular user) can perform the following operations:

- Register a new account.
- Add/Remove owned Car.
- View all request made by him.
- Make a new request.
- Update/Delete request.

The clerk (administrator) user can perform the following operations:

- See list of parking lots.
- See a list of parking requests for a parking lot (ordered by request date).
- Assign Parking Spots to citizens.
- Retract Parking Spots from citizens who did not pay the parking tax.
  (Mark parking spot as free)

3. Application Constraints

- Store the data in a MySQL database.
- Use the Layers architectural pattern to organize your application.
- Use a domain logic pattern: Table Module or Transaction Script.
- Use the Dao pattern to model data access.
- Implement the Data Access Layer both with Hibernate and JDBC. Use the Abstract Factory pattern to switch between implementations.
- The application has no UI.

4. Requirements

1. Create the analysis and design document (see the template).
2. Implement the application.
3. Write at least one Unit Test for each method in the business layer to demonstrate that it works.

5. Grading

Grade	Functionality
5	User Operations. Dao Implemented only with Hibernate.
7	Admin Operations. Dao Implemented only with Hibernate.
8	Data Access Implemented with JDBC and Hibernate. Use Abstract Factory to switch between JDBC and Hibernate.
10	Analysis and Design Document

Note:

1. **Applications that do not respect the imposed architectural and design patterns will not be accepted and will be graded with 2, irrespective to other implementations.**
2. **Business features without Unit Test will not be take into consideration.**