Specifications

Introduction

Some restaurants in Valais would like to manage their food delivery in the region. They ask you to create this platform with the help of N-tier and a database. This project aims to cover the subjects presented in following course:

- 623-1 : Implémentation du système d'information / Implementierung des Informationssystems

Application layers requirements

This application consists of three layers:

- 1 database (SQL Server Server)
- 1 or more class library projects (depending on your needs) to access the data (DAL, BLL, …)
- 1 MVC application as a user interface

© HES-SO Valais

Course 623-1 - Implémentation du système d'information / Implementierung des Informationssystems

Overview

During the semester your will learn how to develop an application using multiple layers. Thanks to exercises and demonstrations, you will implement a Data Layer (DAL), a Business Layer (BLL) and finally you will learn ASP.NET MVC and use it as user interface.

The idea of the project is to change the DAL and BLL created during the lessons and create a new user interface developed as a MVC application.

Database

The database must be a **SQL Server** database and you must create your own tables and their relationships.

The database must store data to manage:

- dishes sold by restaurants
- orders from customers
- staff from VS Eat responsible for the delivery in cities
- staff login
- customers login

You can add test data for restaurants and dishes directly in the database.
You can change/add table properties as you whish

Database schema will be your first deliverable.

Features

Constraints:

- Each order is identified by a number. This number in addition to firstname lastname will be used to cancel the order at least 3 hours before delivery.

User stories:

- **(login)** – A customer must create an account with his/her address before using the website
- **(order)** – A logged customer can choose dishes from a list given by each restaurant available on the website to form an order. He/she will add delivery time (every 15 min) for his/her order. At the end of the order the price that the customer has to pay to the courier will be displayed
- **(delivery management)** – The system will assign the delivery of an order to one courier who is available in the same city as the restaurant where the order is made. One courier cannot have more than 5 orders to deliver every 30 minutes.
- **(delivery interface)** – each courier can log in the system to see his/her upcoming deliveries. Once one delivery is made the delivery person will archive it by pressing a button on the delivery interface.

Please notice that you will work on this project during the class, but you will also need to work on it outside the school.

Ask if one feature is not clear for you!!

If one feature in the minimal specifications is not finished, no point will be given to the extra-features finished

© HES-SO Valais

Deliverables

Deliverables for this project are:

- **A functional and « in production » MVC application**
  - Normalized code based on given standards
  - Application **must run** and **be bug free**
- Technical documentation
  - **Comment** your code in order to make your code easily understandable
  - **1 or more readme files** which explain all required steps to run the application (database path, database users/passwords, application users/passwords…), **ESSENTIAL** for the correction!
- **1 logbook** explaining how you organized yourself in your project, what problems you encountered and how you solved it. Use day granularity.

We ask for 3 deliverables:

1. database diagram **(03.10.2021)**
   First tables modelisation
2. database + business layer **(7.11.2021)**
   Database created
   DAL + BLL layers in visual studio
3. entire project **(09.01.2022)**
   Project deployed on the server at school (153.109.124.35)
   Ready to test

Details

- MVC + N-tier must be used for this project
- **Individual project**

Technical information

Implementation / Development :

- Programming language : C# with Visual Studio (**version provided in the virtual machine**)
- ASP.NET Core
- Database: Microsoft SQL Server Express (**version provided in the virtual machine**) during development and full SQL Server for deployment

Application architecture

- N-tier architecture
  - Presentation layer
  - Data access layer (DAL), Business logic layer (BLL)
  - Data layer (database)

Tips

Start **directly to write code with quality** (using comments, errors handling…). It is better to do it from the beginning, rather than to finalize things at the end, increasing the risk to discover more bugs and never do it.

Follow the development standards and best practices learned during the course:

- Data access layer
- Create a login to access the database (no direct access to data as dbo user)
- …

© HES-SO Valais

Providing features that are fully functional and « clean », even if few features are developed, is a better approach than providing plenty of features with a bad implementation (bugs…).

Organisation

The project **sources** must be delivered:
- for Sunday 3.10.2021 at 23:59 for the first deadline
- for Sunday 7.11.2021 at 23:59 for the second deadline
- for Sunday 9.01.2022 at 23:59 for the last deadline

Deliverables will be uploaded on Cyberlearn.

Evaluation

Evaluation will depend on your work and your implication in this project:

- Evaluation will be individual and based on tests of your project functionalities. It will take place on the **last week before the holiday** (exact date will be defined during the course)
- Project will be evaluated following this list of criteria:
  - Working or not?
  - Clean code, no useless code and best practices are used
  - Correct use of MVC and N-Tier concept
  - SQL commands
  - CSS and look and feel of the user interface
- Your project should be fully functional and in an "in production" status
  - Available functionalities should be fully implemented
  - Bug free
- Evaluation should include questions about your project (architecture, development choices, …)

6	Excellent	Extra work Personal implication Soft skills
5	Good	Match project description
4	Sufficient	Minimal implementation
3	Not sufficient	Big bugs
2	Very not sufficient	Does not compile
1	Inacceptable	Nothing

© HES-SO Valais