**Cruise Reservation Application**

**Overall Project Summary**

Develop an ASP.NET application that provides on-line cruise reservations for a cruise line. The application will consist of web forms and will provide navigation between them through the use of buttons and links. Note: Sections highlighted like this only apply to groups with two members, and sections highlighted like this only apply to groups with three members.

When the application first opens, the traveller will be presented with a login screen. The traveller will provide his/her Oracle login information. The application will validate the credentials by logging into Oracle. After logging in, the traveller is presented with a screen that allows them to select the type of operation they want to perform: (1) View existing cruise reservations, (2) Reserve a cruise, (3) Manage the cruise destinations, or (4) Logout.

**Cruise Reservations Page**

The cruise reservations page contains a list of cruises that are currently reserved for the traveller. Any of these reservations can be cancelled by clicking the corresponding Cancel Reservation button.

**Reservation Page**

The reesrvation page allows the traveller to make a new reservation by specifying the cruise ship and the desired cabin number to be reserved. Attempts to reserve a cabin that has previously been reserved will display an appropriate error message.

**Cruise Destination Management Page**

The cruise destination management page contains the current list of destinations for each cruise ship. Any of these destinations can be removed from the cruise by clicking the corresponding Remove Destination button. The administrator can also add new destinations to a cruise from a list of available destinations.

**Project Stages**

The project is divided into three stages.

1. **Database Design:** Each team will create a database design (entities and attributes, functional dependencies, and ER diagram) which they will then present to the class. The class will vote on a final solution that will be implemented by all teams (a common design).
2. **Database Design Implementation:** Each team will implement the winning database design by mapping the design and producing the required CREATE TABLE statements. Each team will build and test their application in their own database workspace, but will ultimately demonstrate their application in a shared database workspace (hence the need for a common design).
3. **Application Design and Implementation:** For this stage, you and your partner(s) will design and implement the GUI application. While you are working with one or two partners (technically), each will have very separate responsibilities; the work submitted for this stage will be your own. The project is divided up into two or three sections — Part A, Part B, and Part C. Team members will integrate their separate parts together into a single application. Subversion will be used to facilitate project sharing. When you submit your stage one, all participants must declare the section for which they be responsible (A, B, or C).

**Using Subversion**

You and your partner(s) must use the subversion software repository for your shared development work. A username and password will be e-mailed to you.

**An Overview of Subversion**

To get a better understanding of subversion, a number of training videos that demonstrate various aspects of Subversion have been prepared. These videos are best viewed in the order presented below.

* [General Overview of Subversion](https://senema.senecac.on.ca/videos/671/general-overview-of-subversion)
* [AnkhSVN Installation](https://senema.senecac.on.ca/videos/695/ankhsvn-installation)
* [Initial Project Import](https://senema.senecac.on.ca/videos/696/anhksvn-initial-project-import)
* [Checking Out a Working Copy](https://senema.senecac.on.ca/videos/697/ankhsvn-checking-out-a-working-copy)
* [Updating the Base and Working Copy and Resolving Conflicts](https://senema.senecac.on.ca/videos/698/ankhsvn-updating-the-base-and-working-copy-and-resolving-conflicts)
* [Undoing Changes](https://senema.senecac.on.ca/videos/700/ankhsvn-undoing-changes)
* [Looking at History](https://senema.senecac.on.ca/videos/701/ankhsvn-looking-at-history)
* [Specifying Image MIME Type](https://senema.senecac.on.ca/videos/702/ankhsvn-specifying-image-mime-type)
* [Adding Files to the Repository](https://senema.senecac.on.ca/videos/699/ankhsvn-adding-files-to-the-repository)

For more instructions on using the repository, visit <http://people.senecac.on.ca/wayne.bryan/ols655/svc.html>.

**Importing a new Visual Studio Project into a Subversion Repository**

The repository will be used by both you and your partner(s). Only one of you can import the initial project into the subversion repository. Decide which of you it will be.

Start by creating your ASP.NET Web Application in Visual Studio. Rename the default page. Then, from the **File** menu, select **Subversion** and then **Add Solution to Subversion**.

From the Add to Subversion dialog, select your repository location. For your project, you will use the Repository URL that was e-mailed to you. If you are just practicing with Subversion, you can create a local repository using TortoiseSVN.

To create a local repository using TortoiseSVN (assuming you have installed TortoiseSVN), create a new empty folder on your hard drive. Then, right-click in the blank space inside that folder (with Windows Explorer). From the pop-up menu, choose **TortoiseSVN** and then **Create repository here**. Next you will need to get the URL representing this local repository. Right-click on a blank space within the folder (again using Windows Explorer) and choose **TortoiseSVN** and then **Repo-browser**. When the repository browser window opens, note the URL at the top (you will need to use this to import the Eclipse project into the repository).

If the repository URL does not appear in the list of URLs, simply type the URL.

For the folder name, we want to use the project name but with a subfolder of **trunk**. So, select the repository folder (click it) and then check the **Add trunk Folder for Project** checkbox.

To complete the import of the new Visual Studio project into the repository, give an import comment (the default comment **Initial import.** is usually fine). Click **OK**. For the project URLs, you will be prompted for a username and password. Supply the username and password e-mailed to you (you can save them so you won't have to keep re-entering them).

Now that we have attached the project to the repository, we need to select which files are to be versioned within it.

In your project explorer, right-click the top-level solution and select **Subversion** and then **Add**. Then click **OK** to add all of the project files to the repository. Next, select the project (click it) and from the toolbar buttons above, select **Show All Files**. Open the **My Project** folder and expand all of the nodes within it. Right-click each of the files and select **Subversion** and then **Subversion Properties**. In the SVN Properties dialog, click **Add** and select **svn:needs-lock** and click OK. Repeat the process for the remaining files in the My Project folder and also for the web form **designer** file and the **Web.config** file. Finally, right-click the solution and select **Solution File** and then **SVN Properties** and set the svn-needs-lock property. And then right-click the project and select **Project File** and then **SVN Properties** and set the svn-needs-lock property.

You are now ready to update the repository with your project files. Right-click the top-level solution folder and select **Commit Solution Changes**. Give a commit comment (like **Initial project import.**) and click OK.

Your project has now been versioned within the Subversion repository.

For more instructions on using the repository, visit <http://people.senecac.on.ca/wayne.bryan/ols655/svc.html>.

**Importing the project from the Subversion repository into Visual Studio**

If your partner has imported the project into the repository, you now need to get a local copy into your own workspace. This procedure can also be used to create a second local copy (maybe on your home computer) of the project.

Start by clicking the Visual Studio **File** menu and choose **Subversion** and then **Open from Subversion**. If your repository URL does not appear in the drop down, click the **Add repository url** button. Next, specify the repository URL (for your projects, this will be the Repository URL e-mailed to you). If you are prompted for your username and password, use the ones e-mailed to you (you can save them so you won't have to keep re-entering them). Open the project folder in the repository and choose the **trunk** folder. Select the solution (**.sln**) file and click **Open**. Specify the local directory where the local project files should be saved and click OK.

You now have a local copy of the repository project.

For more instructions on using the repository, visit <http://people.senecac.on.ca/wayne.bryan/ols655/svc.html>.

**Grading**

Marks for this project are broken down as follows:

|  |  |
| --- | --- |
| **Description** | **Weight** |
| Database Design and ER Diagram | 2.5% |
| Mapping and CREATE TABLE | 2.5% |
| GUI design & Application Code | 8% |
| 1Presentations | 1% |
| 2Peer Evaluation | ±2.5% |
| Subversion Repository Use | 1% |

1Each person must present in order to get a grade for the presentation portion.  
2The Peer Evaluation component comes from an evaluation from your classmates. The peer evaluation component can raise or lower your grade and is determined by taking the difference between your peer evaluation score (out of 2.5%) and the average peer evaluation score (out of 2.5%). This ensures that your evaluation by your peers is relative to the evaluation of your peers.

[Stage 1 Details](http://people.senecacollege.ca/wayne.bryan/ols655-20162/project/stage1.html)  
[Stage 2 Details](http://people.senecacollege.ca/wayne.bryan/ols655-20162/project/stage2.html)  
[Stage 3 Details](http://people.senecacollege.ca/wayne.bryan/ols655-20162/project/stage3.html)