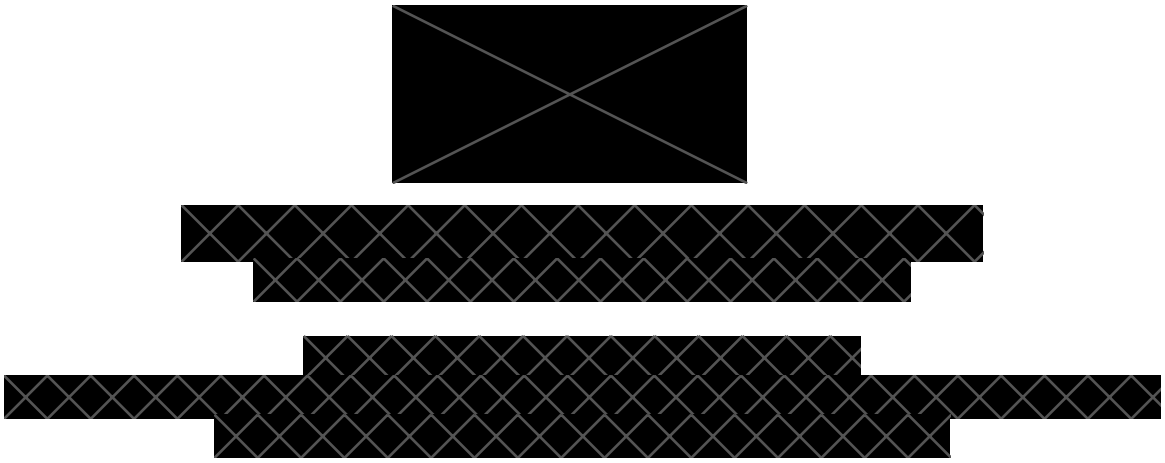


NOTE: Ensure you read and adhere to the referencing requirements of this unit as described on pages 7 and 8 in the learning guide. This is an individual assignment. It is each student's responsibility to be aware of and adhere to the policies and rules regarding misconduct.

Scenario

Rent-a-Car is a small family owned company that rents out cars, mainly to holiday makers. As Australia is a such a vast country it is often necessary to fly to holiday destinations, but families and holiday groups still often wish to have a vehicle available during their holiday. It is this market "Family Rent-a-Car is aiming their business at.

You have been hired by Rent-a-Car to produce an XML based trial solution for their customer record and rental record system. The system is to be used by Rent-a-Car's staff only. The first stage of this system involved the creation of data storage and the construction of Rental booking display for past, current and future bookings. The second part (this assignment) deals with server-side processing of the XML files to manage customer records and a partial solution for rentals, using PHP or classic ASP (do **NOT** use ASP.NET – the solution **MUST** run on your DWAX server account).

The system is to be used by Energetic Energy's staff only.

For this server-side assignment, you will need to implement the following functionality:

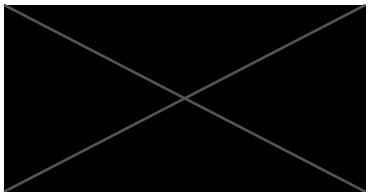
- Login to the system for staff
- Manage customers:
    - Add a new customer
    - Update/Change customer details
- Manage Rentals:
    - Adding a new rental record

Each part of the functionality will be discussed in detail below.

Coding standard:

You are expected to write code that is easy to read and for the viewer to quickly get an understanding of what has been done. This includes:

- using descriptive variable, element and attribute names
- indenting code appropriately so it is easy to see the nesting structure used
- documenting your code with comments where appropriate
- when creating links, only **relative paths** must be used.



Deliverables:

You will be required to deliver the following by the due date:

1. **4 XML files**:
   - `staff.xml` (see description below in the Login section)
   - `customers.xml`

     The `customers.xml` file will contain customer records. Each customer record contains:
     - customer number
     - name: for a particular customer title and middle name is optional (so may or may not need to be stored), however first name and last name are required.
     - address – all components are required (street, suburb, postcode, state)
     - telephone – a customer must have at least one phone number, but may have up to three phone numbers
     - drivers licence
     - customer email address (not present in the sample data)
     - Nominated drivers (there can be several – if there are none nominated, the customer is assumed to be the driver). **For each** nominated driver, the following must be collected (note: logically this would make more sense to store in the rental record as it might change each time the customer rents a car, however this is consistent with assignment 1):
       - Licence number
       - name: title and middle name is optional (so may or may not need to be stored), however first name and last name are required.
       - address – all components are required (street, suburb, postcode, state)
       - Phone number (just one - a mobile number, is required)

     No other information is to be stored in this file.

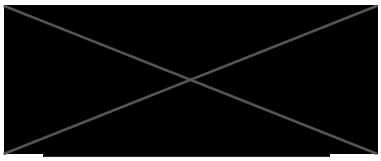
   - `vehicles.xml`

     The `vehicles.xml` file will contain details of the rental vehicles the company owns. Each vehicle record must contain:
     - Rego number
     - Make
     - Model
     - Year of manufacture
     - Current odometer reading
     - Description

     No other information is to be stored in this file.

   - `rentalRecords.xml`

     The `rentalRecords.xml` file will contain car rental bookings that have been made, as well as past car rental records. Each record contains:
     - rental record number
     - customer number (would match to one of the existing customers already in the customer file)
     - rego number (would match one of the rego numbers in the vehicles file)



- date the booking was made
- date the booking is for (when the car will be or was picked up)
- Number of days the rental is for.
- Charge per day (this would vary depending on length of time the car is hired for, any special campaigns that are running, etc so is entered at the time of booking).
- Fuel tank full on return (yes, or no, (or true/false) or nothing if the rental has not yet started or not yet returned). If no, which would only happen if the car was returned with less than a full tank, the amount that was charged for refilling will need to be stored.

No other information is to be stored in this file.

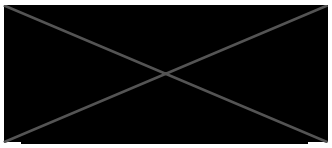
- You must demonstrate the **use of attributes in all XML files except staff.xml**

NOTE 1: the sample data provided in a separate document for the client side assignment **must** be present in the XML files for marking purposes.
NOTE 2: the xml files must **NOT** have any links to schema or xsl files. If reusing the same files as in assignment 1, ensure links to schema and xsl files are **removed**. Marks will be **deducted** if these links are present. No schemas or xsl are needed for this assignment.

2. Login:
This will be the entry point to the system.
An xml file must be created that stores the username and password of staff (`staff.xml`). This file must have login details for at least 3 staff members. Note that the username and password for a staff member must *not* be the same sequence of character, they should be different. All username/password pairs must be unique.
The login page should collect the information entered by the user, check against the data stored in `staff.xml`, and if valid, direct the user to the default page (see below).
All pages in the system except the login page must only be viewable by users who have successfully logged in to the system.
Ensure the staff login details are listed in your readme file.

3. Default page (`default.php/.asp`) – must have links for the following options:
- Manage customers
- Manage Rentals
- Logout

4. The customer management page must have three options available:
- Add a new customer:
  Selecting this option presents the user with a form to record customer details (customer number, driver licence, title (optional), name (first/last/middle (optional)), phone number(s), address, email address). The form should also allow for entry of zero to many nominated drivers (for each nominated driver: title (optional), name (first/last/middle (optional)), drivers licence, address, phone). When the form is submitted your solution must add a new customer record containing the correct data to customers.xml.



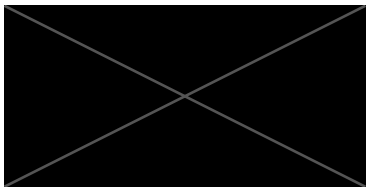
- Update/Change customer details:
  Selecting this option presents the user with a search facility to search for a particular customer by surname, customer number or licence. When a match is found, the customer details are shown in an editable form. Only customer name, address and phone numbers can be changed. When the form is submitted the updated data is saved for the right customer record in customers.xml
  If a match is not found when a search is conducted, an appropriate message must be displayed to the user.
  If more than one match is found, a list of matching customers must be displayed for the user to select from before displaying an editable form for the selected customer.
- Add nominated driver to existing customer:
  Selecting this option presents the user with the same search functionality as used for Update/Change customer details. Once the correct customer is located, the user is presented with a form to collect details to add a new nominated driver to the existing customer record.

5. Manage rentals:

- Selecting this option presents the user with the ability to create a new rental record. A new rental record can only be created for a customer that has already been added to the customer records (the customer must exist in customers.xml), and can only be for a vehicle that exists in vehicles.xml. Your solution must ensure this. A new rental record will need the following data: Rental record number, customer number (must match existing customer), rego number (must match existing vehicle), the date the booking was made, the date the vehicle is to be picked up, the number days the booking is for, and the dollar charge per day. Although not entered at this time, your solution must create the same xml structure for new rental records as you are using for existing records, so provisions must be made for future entries of fuel tank being full or not on return, and if there is a fuel charge (see sample data for assignment 1). Future versions of the trial solution will also programmatically check that the vehicle is free at the time of the proposed rental, however due to time limitations this is not implemented for this trial. Additionally, finalising rentals (when a vehicle is returned) is postponed to later trial versions of the system and will not be implemented in this assignment.

The designs of these pages are left open to your decision; however, the design must be appropriate for the users and the application's intended use and must have a professional look and feel, include necessary information for the user as well as appropriate navigation. You may use (but this is not necessary) a CSS framework, such a Bootstrap, to help with designing, but you may not use a template or theme of any kind. This means all code must be your own work, and that you have only applied a CSS framework to help with the look of 'your' design.

6. **1 read-me file** (`readMe.txt`)
This file should contain any notes for the marker, as well as the login details for the 3 staff. For example, if your assignment has sections that only partially works, then this should be noted here. This information may be important for the testing sequence used to test your assignment, so ensure relevant information is provided.



Business Rules this part of the solution must implement:

- A customer must supply first and last name. Title and middle name are optional. This means your solution must work if no title and/or middle name is entered, as well as work if these are entered. The correct structure must be created in the xml file.
- A customer may only supply one phone number, but can supply up to 3 phone numbers. Your solution must allow these options.
- A customer may not have any other nominated drivers, but could have many. Your solution must allow this.
- Before a rental record can be created, the customer must exist in the xml files. Your solution must ensure this.
- A rental record can only be created for Vehicles that exist in the xml files. Your solution must ensure this.
- Assumption – For this trial we are working with Australian addresses only

NOTE on contents of XML files: Ensure the xml files submitted contain only the sample data as supplied in the *Assignment 1 specification* folder on vUWS (+ customer email address), as part of the marking process will be to create new rentals using existing data.

Implementation Notes:

As the focus of this assignment is a server-side implementation using xml, there are a couple of things to note:

- The forms must look professional and be created with good usability design in mind, however it is not necessary to employ JavaScript error trapping for form inputs at this stage.
- Schema files for the xml files is not required for this assignment, and there should be **no** Schemas, DTDs, CSS or XSL attached to the **XML** files for this assignment.
- This is a stand-alone assignment. You will not require functionality from the client side assignment. You can re-use your xml files if appropriate, and you can re-use your design for the landing page if the design in assignment 1 was appropriate (however you will need to modify your links to what is listed above).

This assignment will require some research on how to combine XML and PHP (or classic ASP). Note that your assignment **must run on your DWAX server account** (do not use ASP.NET). A solution can be created in many different ways. It is up to you to decide on the design of your site, but the site must have a professional look (keep in mind the purpose of the application). Remember to keep the user informed about the activities they are attempting. There should be appropriate responses from the system, and appropriate **navigation** provided in all parts of the system.