You are required to create full-function .NET application to solve a case study.

Application functions include:

- Login
- Media Browse
- Media search (at least 3 criteria including Title search)
- Media borrow
- Media reserve
- Media return
- Master information maintenance (Add new, Update, Delete for Media, Genre, Language, Director)
- User information maintenance (Add new, Update, Delete)
- Reporting (at least 1 report)

Matt's BattlePlan

You should already have Login, Media Browse, Media Search and User information maintenance from class and the previous assignment.

Master information maintenance

Use the user add, update, delete code we did in class to help build the section to add Genre, Language and Director aswell as add, update, delete for Media. Remember to start in the Data Layer, then Business, then Presentation. NOTE: selecting genre,language,director for media should be a drop down

Media Borrow/Reserve/Return

You need to add to the user side to do the borrow, reserve, return. This part is intentionally vague. Look at the database and look at the TabReserve and TabBorrow tables.

Reserve:

This one is easier. All you need to do is add a new entry to the table adding the user and mediaID and get the current date of the system (google search C# get current date)

Borrow/Return:

Borrow is more complicated. So you need to add entries into it too, but only insert the user ID, media ID, borrow date and estimated return date. Let the rest of the fields be entered by default values. I'm guessing by the data here (I wouldn't have done it this way) that a media that has the ActualReturnDate greater then or equal to the current date, then its considered returned (default value seems to be 1/1/2000?)

Presentation:

I would add a Borrow and Reserve button to the current user page that Borrows or Reserves the media if its available. Only borrow a media if its not reserved by anyone else. If it is reserved by the user, delete their reservation when they borrow the media.

For return I would make a combo box populated with all of the media the user has borrowed and button next to it saying "Return Media" which basically sets the 'actualReturnDate' for that borrow entry to the current date. Only list borrowed media that have a actualReturnDate less then the BorrowDate (I know its confusing).

Reporting

This part is actually pretty easy. In admin, list a bunch of useful information e.g List all borrowed media entries or reserved media entries

re required to create full-function .NET application to solve a case study.

Application function include:

- Login
- Media Browse
- Media search (at least 3 criteria including Title search)
- Media borrow
- Media reserve
- Media return
- Master information maintenance (Add new, Update, Delete for Media, Genre, Language, Director)
- User information maintenance (Add new, Update, Delete)
- Reporting (at least 1 report)

Matt's BattlePlan

You should already have Login, Media Browse, Media Search and User information maintenance from class and the previous assignment.

Master information maintenance

Use the user add, update, delete code we did in class to help build the section to add Genre, Language and Director aswell as add, update, delete for Media. Remember to start in the Data Layer, then Business, then Presentation. NOTE: selecting genre,language,director for media should be a drop down

Media Borrow/Reserve/Return

You need to add to the user side to do the borrow, reserve, return. This part is intentionally vague. Look at the database and look at the TabReserve and TabBorrow tables.

Reserve:

This one is easier. All you need to do is add a new entry to the table adding the user and mediaID and get the current date of the system (google search C# get current date)

Borrow/Return:

Borrow is more complicated. So you need to add entries into it too, but only insert the user ID, media ID, borrow date and estimated return date. Let the rest of the fields be entered by default values. I'm guessing by the data here (I wouldn't have done it this way) that a media that has the ActualReturnDate greater then or equal to the current date, then its considered returned (default value seems to be 1/1/2000?)

Presentation:

I would add a Borrow and Reserve button to the current user page that Borrows or Reserves the media if its available. Only borrow a media if its not reserved by anyone else. If it is reserved by the user, delete their reservation when they borrow the media.

For return I would make a combo box populated with all of the media the user has borrowed and button next to it saying "Return Media" which basically sets the 'actualReturnDate' for that borrow entry to the current date. Only list borrowed media that have a actualReturnDate less then the BorrowDate (I know its confusing).

Reporting

This part is actually pretty easy. In admin, list a bunch of useful information e.g List all borrowed media entries or reserved media entries

CLOUD

You are required to create a web services to replace local connection between Win-based User Interface (UI) and Business Logic function that you implemented in Part 2.

Part 3.1 Convert direct accesses to web services.

You should have 2 projects solution to complete this Assignment as following:

- Web service project – Developed in Visual Web Developer
- Win-based UI project – Developed in Visual C#

Part 3.2 Win-Based applications

Ensure that all function are required from Part 2 are working, after change the direct reference to Business Logic to Web Services.

In this Assessment, **portability would be concerned**. All static value should be implemented as constant or enumeration.

Implementation Guidelines

A sample implementation would contain the following components

- User Interface (Win based application)
    - Login for staff/student
    - Master information modify function (Books, Authors, Categories etc.)
    - Books borrow/return function
    - Query page with results (Reporting for librarian/manager)
    - Logout Button
    - Do not add Business Logic and Data Access in UI Solution
    - Proper use of Constant and Enumeration
- Web Services
    - Consist of Web services, Business Logic and Data Access project
    - Query Management (Select, Insert, Update and Delete)

Matts Battle Plan

First we need to make the web services part before we break our original project.

Web Services

Open up visual studio web developer and create a new ASP.NET empty web application. Once its ready, go file->add->existing project and add your Data Access layer csproj file and Business Logic Layer csproj file. Your web application needs a reference to the business logic layer. From here reference the code I've done in class to make web services for all of your business logic layers methods that the presentation needs.

Note: Can only send basic datatypes via web services (ints, doubles, arrays, strings etc) for cross compatability. So make sure to send either basic types OR dataTables if the information is a list of results.

Presentation Layers

This part is not fun. Remove all references of the Business and Data Access projects from your Presentation Layer project. It should have broken your project in all sorts of places. Good!

Run your web services project, it should run a localhost url which you need to copy.

Now we need to add a web service reference in the presentation layer project, so right click services and choose 'Add Service Reference'. Ignore everything on this screen, click 'Advanced', ignore next screen and click 'Add Web Reference'. On the third screen, paste in the url and click the green arrow next to it. It should find your web services. Now on the right hand side give your web services reference a name you can easily remember and click 'Add Reference'. It should add to a folder called 'Web References'.

From this point, reference the class code and basically swap out allllllllllllllll of your business logic method calls here with web service calls. The hard part is just using datatables instead of using models. Good Luck!