CSE 445/598 Assignment / Project 5 (100 Points)

Summer 2016

Due: July 2, 2016, 11:59pm (Arizona Time), Plus one day grace period
Project must be submitted into the Blackboard's site AND deployed into the WebStrar.
We will read code from your Blackboard submission and test your code from WebStrar

---

Introduction

The aim of this assignment is to make sure that you understand and are familiar with the concepts covered in the lectures, including the Web application architecture, components and structure, controls, and state management.

This is an individual assignment. Each student must complete and submit independent work. No cooperation is allowed. You must use WebStrar to host your files, services, and application. The entire project must be submitted into the blackboard AND into the WebStrar server.

Practice Exercises (No submission required)

No submission is required for this part of exercises. However, doing these exercises can help you better understand the concepts and thus help you in quizzes or exams.

1. Reading: Textbook Chapter 5 and Chapter 6 section 6.2.
2. Do the multiple choice exercises in text for Chapter 5 and Chapter 6.
3. What kinds of Web-based computing models exist? Where is the computation (client side or server side) done in each of the models?
4. Explain how the files in ASP .Net Website application are organized in the application folder.
5. Explain what types of files exist in an ASP .Net Website application and what the functions of each type file are.
6. How do we create a user control, and how do we include the user control into a Web page?
7. What is the most frequently used function of the Global.asax file?
8. What kinds of state-saving mechanism exist, and what are the main features of each kind state-saving mechanism?
9. What is the most general state variable in ASP .Net? What type of data can be stored in this kind of state variable?
10. Compare and contrast the state management mechanisms: View State, Cookie, Session State, Application State, and Caching variables.

11. Discuss the relationship between the dependency and callback in the insert() method of Cache class.

12. How are dynamic graphics generated and maintained in ASP .Net environment? Read text section 5.6.

13. What is the execution model of ASP .Net application in the tightly managed Web server?

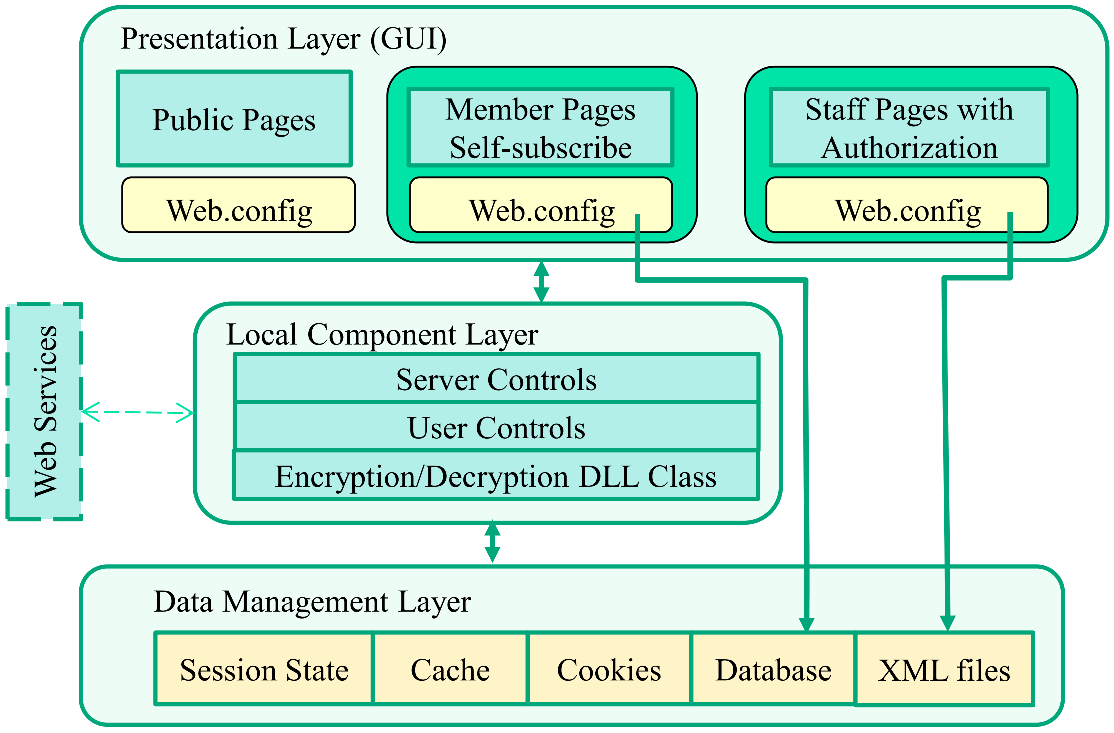
14. Study for the questions 2 through 12 in text section 5.8, and study the questions 2 through 15 in text section 6.5. Make sure that you understand these questions and can briefly answer these questions. Study the material covered in these questions can help you prepare for the quiz and exam, and can help you understand the homework assignment.

15. Having learned all the techniques that you need to build Web applications, now, you need develop your own Web application. Brainstorm among the team members and come up with your ideas of developing a sensible Web application. Ideally, this project solves a problem that you always wanted to solve, for example

    - Managing membership, finance, and activities of the club that you are an officer.
    - Managing your hobby collections: When was each item purchased, price, current value.
    - Loyalty center: allow businesses to check loyalty of a costumer based on buying history and habit;
    - Developing a Web testing tool: analyzes a Web page at a given URL and calls the links in the page to discover broken links;
    - An online store with recommendation list, like Amazon, eBay, etc.,
    - A mobile application to manage your shopping list and best places to buy, taking into account time, driving distances, and cost.
    - Combine this project with your other course projects. In this case, you need to clearly state what part is submitted for this course, and what part is submitted for another course.

Project Description (Submission Required)

In this project, you will develop a service-oriented Web application with access control. The application must simulate a realistic application for the end users. The architecture is shown in the following figure.



The system must be implemented as a **Website** Forms application and must be deployed to the given Web server. You may implement the application that you have outlined in project 3. You may choose to implement a different application. The code must be well commented. The application implemented must meet the following organizational, architectural, and functional requirements. The composed Website must have at least the following layers of components.

1 Presentation/GUI layer, consisting of ASPX pages and server controls, which allow users to interact with the application. The application must have at least these four ASPX pages: [20 points]

a. Public page. In this page, you must introduce clearly what application the system offers, how end users can sign up for the services, how the users (TA) can test this application and the required test cases/inputs. All the components and services used in the application must be listed in a "Service Directory", similar to the one that you created in Assignment 3. The directory must include provider name (member who is responsible for the component), type (Web service, DLL function, user control, etc.), operation name, parameters and types, return type, function description, and link to TryIt page. You can combine the TryIt pages into your application logic.

b. Member page: In this page, you must introduce clearly what application and functions the system offers. Users can register (self-subscribe) to obtain the access to this page. An image verifier must be used when a user registers. You can use the Forms application's built in Account management for this purpose (save user name and password). You may create your own access control component and store the credentials in an XML file.

c. Two staff pages: These two pages must have authentication and authorization access control. Some staff members can access one page, and other staff members can access the other page. The user

ID and password must be stored in an XML file. You are not allowed to use the built-in account management in order to exercise XML manipulation learned in Chapter 4.

2 Local component layer. This layer consists of the following two types of components: [20 points]

a. User control, and the control must have the fragment caching function implemented.

b. DLL class library modules, to implement at least the encryption/decryption functions.

You must implement at least one of the components. You may choose more components or other type of components in order to implement your business logic.

3 Remote service layer. This layer consists of at least one sensible Web service developed by you and one service discovered from a public repository (this service can be from the ASU repository). Self-developed services must be deployed into WebStrar.
It is the developers' (your) responsibility to make sure that the services are available and reliable when the TAs grade the assignment. [10 points]

4 Data management layer consists of **both** temporary states (session state and cache) **and** permanent states (XML file and database). You must implement at least two of the items in the list above: [30 points]

a. Permanent state (Built-in database or XML) for storing the user names and passwords of self-subscribed users.

b. Permanent states (XML file) for storing staff user ID, password, and role for authentication and authorization.

c. Cookie for storing user profile and Session state for storing temporary states for sharing among the sessions

5 Deploy all the components into the WebStrar server for testing. [20 points]

6 Submit complete code into Blackboard for grading (code reading).

Submission

You submit the following work (all questions) into the Blackboard submission site AND into the WebStrar. We will read code from your Blackboard submission and test your code from WebStrar.

Question 1: You must submit the Service Directory that lists all your components, so that we can test your components. The page must have a TryIt function for each component that you developed. If a component is implicitly used and is not visible/testable from your integrated GUI, you must add a test page to make it testable.

Question 2: Submit the local components that you have developed.

Question 3: Submit the Web service that you have developed, or that you have discovered.

Question 4: Submit the state management component that you have developed.

Question 5: Deployment. All code must be deployed into WebStrar. If your code is running on local host only, you will not receive the deployment points.

Question 6: Submit code into Blackboard for grading (code reading). If no submission in the Blackboard, your assignment will not be graded, and you will receive 0 grade.

Blackboard submission notice: The assignment consists of multiple distributed projects and components. They may be stored in different locations on your computer when you create them. You must copy these projects into a single folder for blackboard submission. To make sure that you have all the files included in the zip file and they work together, you must test them before submission. You must also download your own submission from the blackboard. Unzip the file on a different location or machine, and test your assignment and see if you can run the solution in a different location, because the TA will test your application on a different machine.

Grading of computer programs

The TA will grade your program following these steps:

(1) The TA will read your program and give points based on the points allocated to each component, the readability of your code (organization of the code and comments), logic, inclusion of the required functions, and correctness of the implementations of each function.

(2) Compile the code. If it does not compile, 40% of the points given in (1) will be deducted. For example, if you are given 20 points in step (1), your points will become 12 if the program fails to compile.

(3) If the code passes the compilation, the TA will execute and test the code. If, for any reason, the program gives an incorrect output or crashes for any input, 20% of the points given in (1) will be deducted.

(4) The TA will test the deployed application in the server assigned to your team.

Please notice that the TA will not debug your program to figure out how big or how small the error is. You may lose 40% or 20% of your points for a small error such missing a comma or a space!

Late submission deduction policy:

For this last assignment:

- No penalty for late submissions that are received within 24 hours of the given deadline;
- 2% grade deduction for every hour after the first 24 hours of the grace period!
- No submission after Monday, 11:59pm.