Homework 4

Fixing Payment Card Industry Java Web Application

Overview

In this homework, you will modify an existing Java Web application that violates several Payment Card Industry guidelines and recommendations. Your task is to locate the issues, based on the readings for this course, identify what is wrong and then fix the code. You will discuss each issue in terms of why the issue may cause a security vulnerability, and how you specifically fixed the issue.

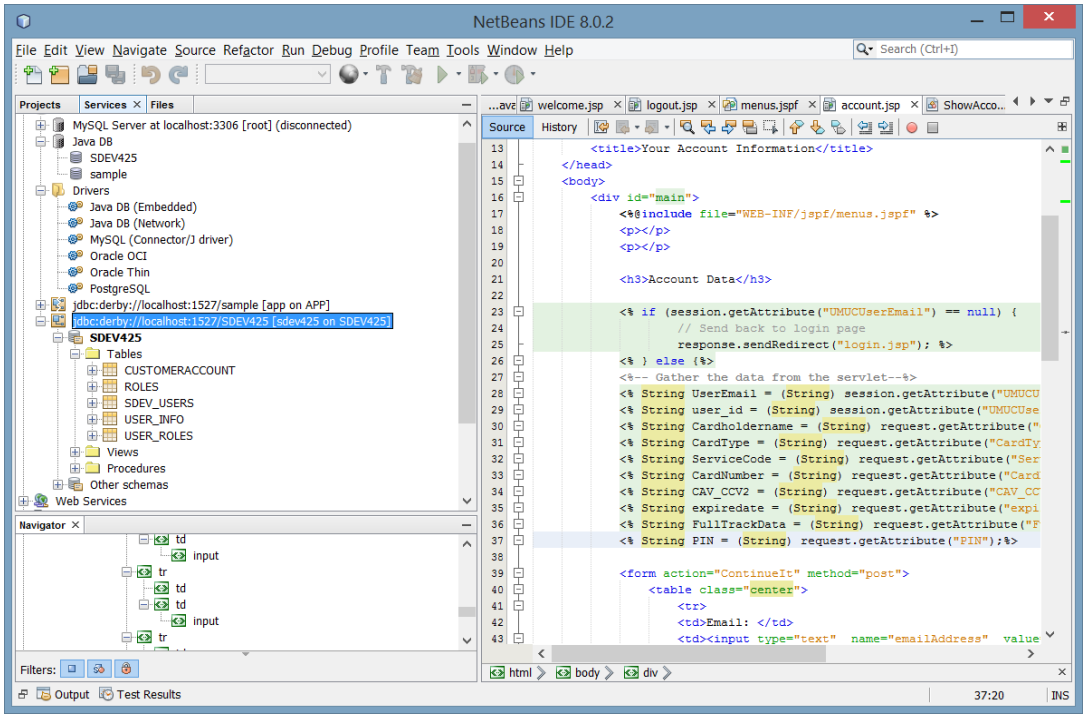
Assignment

Review, Run and Understand the Sample Java Web application.

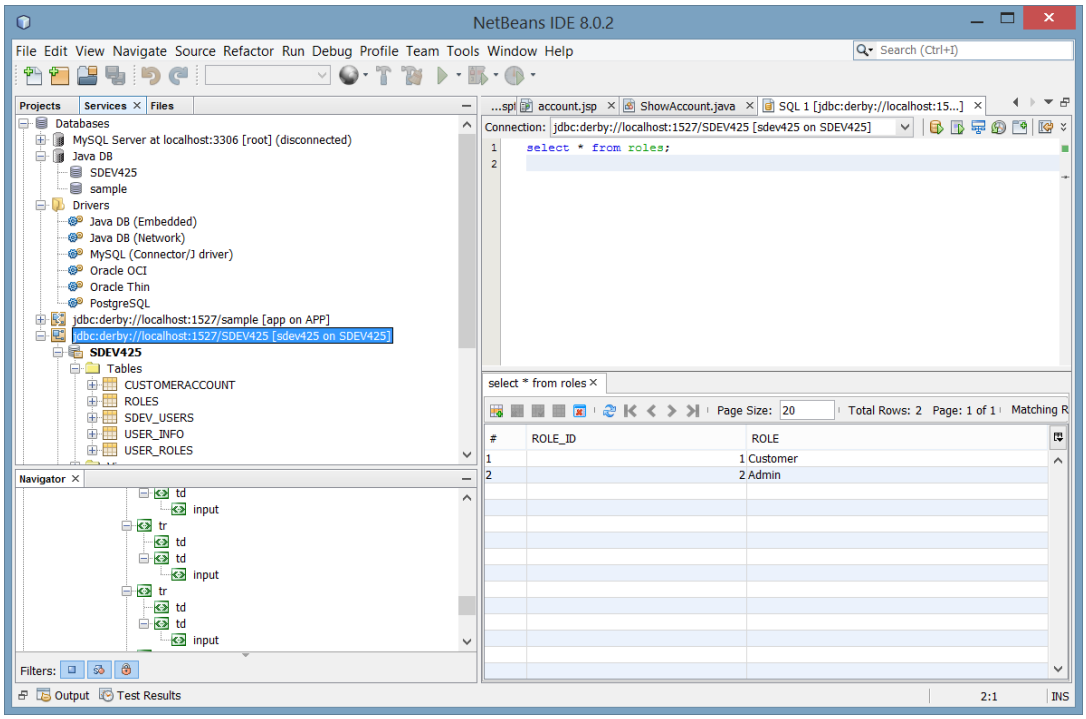
The current code, uses Java JSP and Servlets to allow a user to login to their account and view credit card data stored in the database. The functionality is relatively simple but several PCI compliance rules have been violated that will prevent the application from being approved by a PCI software auditor.

You should first load up the application, populate the database and make sure the application is working in your environment as expected. The application uses the Java Derby relational database. The script used to populate the application is attached in your project folder as well as the Java web project itself. You should be able to open the existing project using Netbeans. However; you may need to load the Derby drivers to the libraries for the project.

You can create a new database connection by clicking the services folder and then right mouse-click on Databases select new connection. You can then create a connection for the SDEV425 database. Below is a screen capture that represents my configuration.

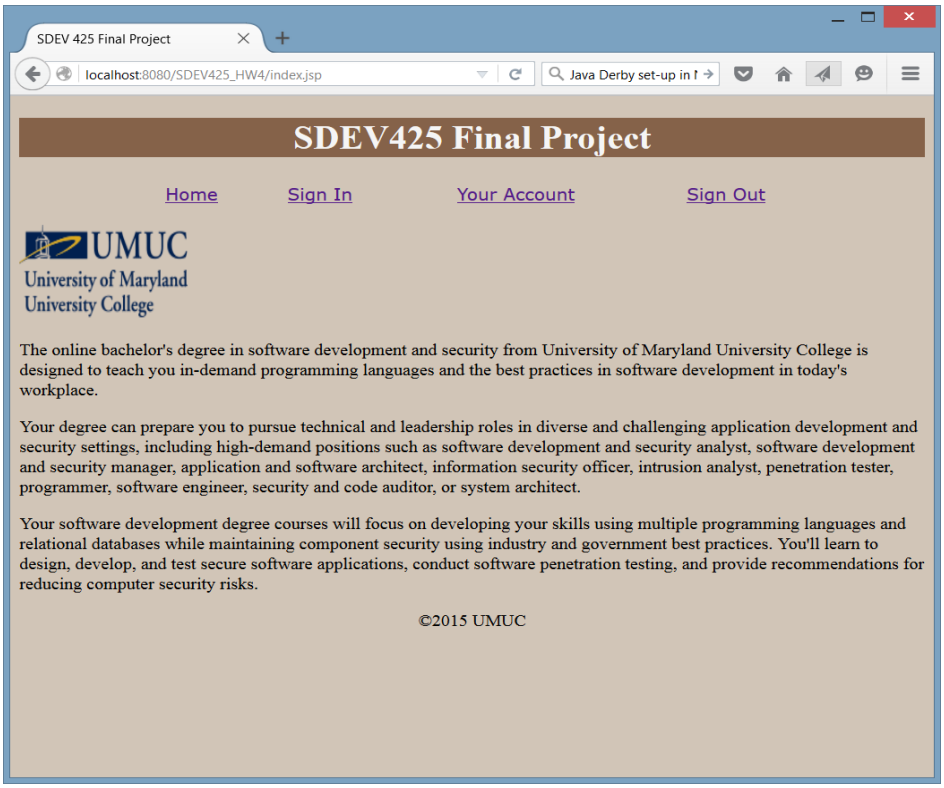


There are some tutorials on the Internet on using the Derby database within Netbeans. (e.g. https://netbeans.org/kb/docs/ide/java-db.html) The key is making sure you have the drivers in the library of your project and you run the scrips to populate the tables. To execute a command in the database from within Netbeans you right click the connection and select execute command. A window will then pop-up for that connection and you can execute any command, including all of those database scripts provided to you.



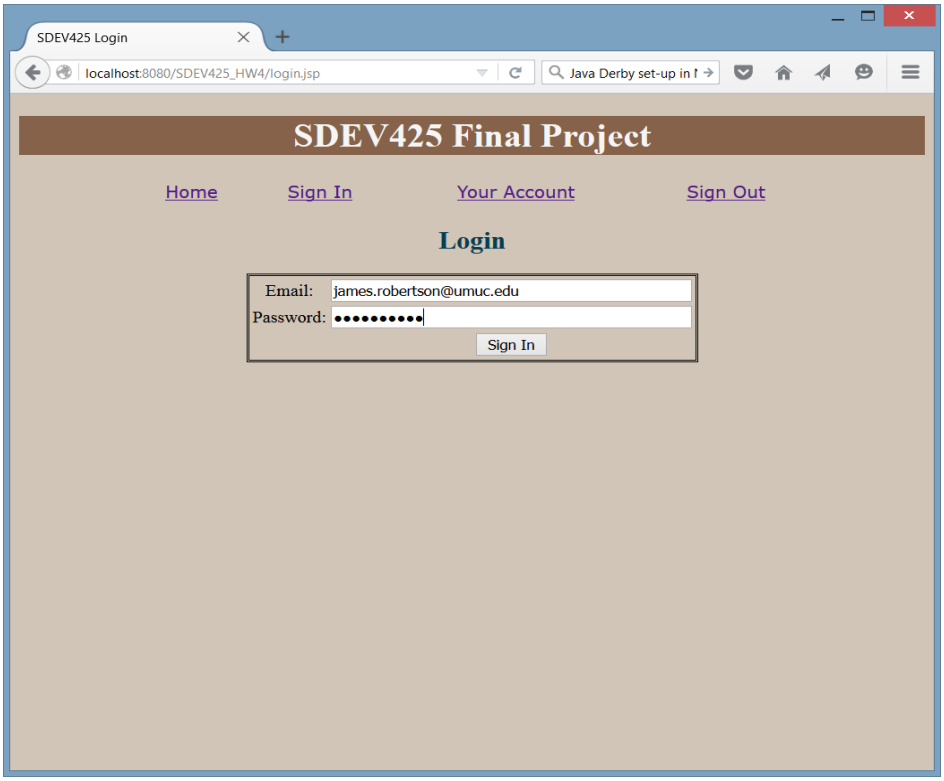
Once you have the database loaded, you can try the application. (This assumes you have properly installed the Java EE when you installed Netbeans).

To launch the application, just highlight the java project you loaded (SDEV425_HW4) and click the green arrow. Once launched, your glassfish server will start and your default browser will be invoked. The application will automatically launch and the home page will be displayed.

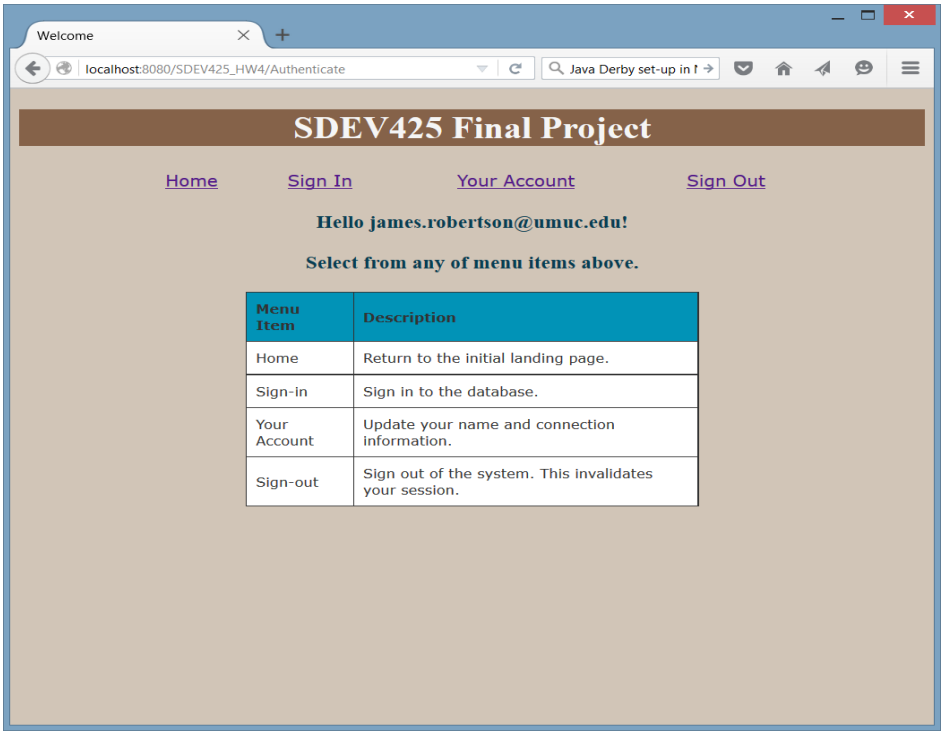


You can (and definitely should) review the database script to see the current users and associated passwords. As a test, you can click the Sign in link and enter this account information

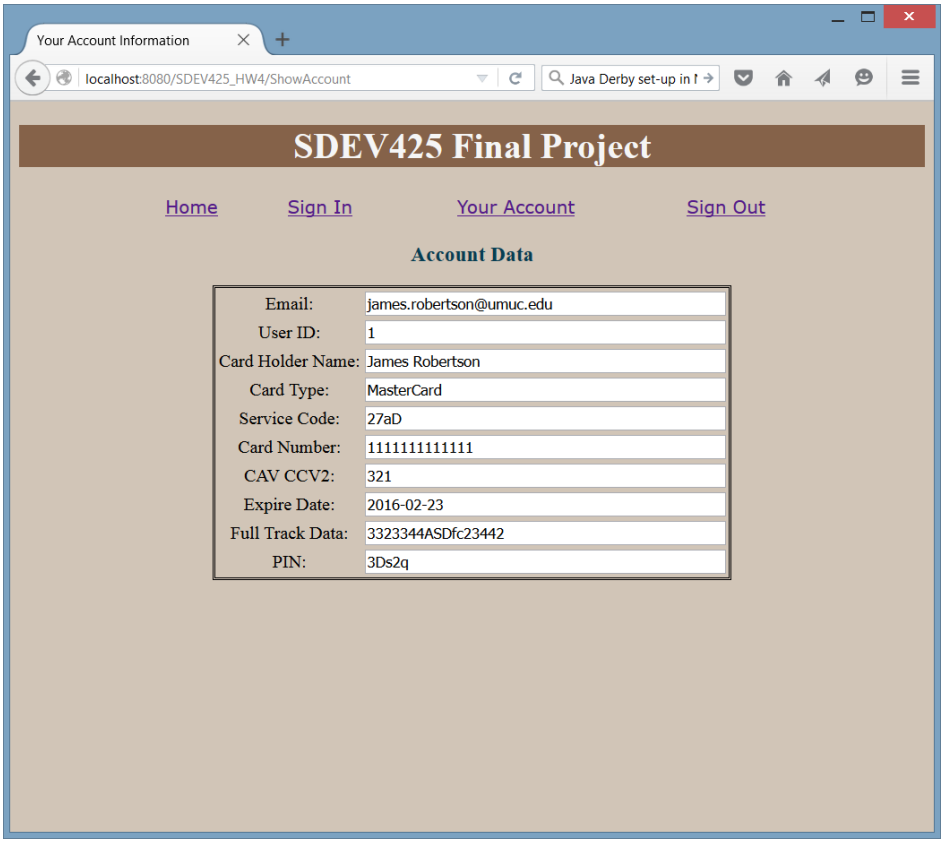
Email: james.robertson@umgc.edu
Password: mypassword



After successful sign-in you will see the welcome message displayed:



Clicking on the Your Account menu will display the credit card information.



Clicking on Sign out will invalidate the current session.

Be sure to demonstrate the application runs properly in your development environment.

Carefully, review the code and perform analysis as needed. You should experiment with running the application as well as reviewing the code to identify possible areas of security concerns. You don't have to be an expert in JSP/Servlets, html or css to be able to find some of the issues that you have read about in the PCI documentation. However; it is recommended you experiment with the code so you have a baseline familiarity with the model and know how the JSP and servlets communicate with each other.

Focus on the PCI compliance issues found in sections 6 through 9 as you look for issues. There are multiple issues and you should work to fix and document as many as possible.

Hints:

a. Make sure your Java EE Netbeans is working properly.
b. Work to get the Derby database populated and working with the SDEV425_HW4 Web application.
c. Study and experiment with the code. For example, even if you haven't used CSS style sheets before you should be curious about how color schemes are used and how easily they can be changed in the css file. You should also note how the database connections are made and how JSP and servlets communicate.
d. Again, start on this early. This will take you longer than you think.

Deliverables

Provide all of your modified Java code, your modified database script and a PDF document describing how you addressed each issue. You should clearly describe the code and what PCI compliance issue were violated and how you fixed it. You should provide screen captures as needed to support your findings and improvements.

Be sure your PDF document is neat, well-organized and is well-written with minimal spelling and grammar errors. All references used should be included in your document.

Grading rubric:

Attribute	**Meets**	**Does not meet**
Sample Java Application	**10 points** Demonstrates the Java application is running properly in your development environment. (10 points)	**0 points** Does not demonstrate the Java application is running properly in your development environment.
PCI compliance	**70 points** Identifies PCI compliance issues found within the application. (35 points)	**0 points** Does not identify PCI compliance issues found within the application.

	Fixes and documents the PCI compliance found during the analysis. (35 points)	Does not fix or document the PCI compliance found during the analysis.
Documentation and Submission	**20 points** Provides all modified Java code, modified database script and a PDF or word document describing how each issue was addressed. (5 points) Provides screen shots and descriptions of the successful executing the code and the resultant output as applied to each security control. (5 points) Document is neat, well-organized and is well-written with minimal spelling and grammar errors. (5points) All references used should be included in your document. (5 points)	**0 points** Did not provide all modified Java code, modified database script or a PDF or word document describing how each issue was addressed. Did not provide screen shots and descriptions of the successful executing the code and the resultant output as applied to each security control. Document is not neat, well-organized and is not well-written with minimal spelling and grammar errors. All references used were not included in your document.