Zack Chambers

CST 235

Charbel Elkhoury

March 31, 2019

**Assignment 7: Securing Access to Web Pages in an Application**

**GitHub Link**: <https://github.com/zchambers3/CST235/tree/master/Week7>

Java EE provides APIs for dealing with requests from HTTP clients like Servlet API, JSPs, JSTL etc. For database access it provides APIs like JPA, JTA and there are APIs for creating web services, APIs for Security management and much more.

When implementing custom Java EE security and managing Java EE containers, software developers have access to the Java Authorization Contract for Containers and the Java Authentication Service Provider Interface for Containers. Java EE provides declarative and programmatic security. Java SE provides built-in cryptography, authorization and authentication.

The Java EE Security API specification defines portable, plug-in interfaces for HTTP authentication and identity stores, and an injectable SecurityContext interface that provides an API for programmatic security.

Implementations of the HttpAuthenticationMechanism interface can be used to authenticate callers of web applications. An application can supply its own HttpAuthenticationMechanism or use one of the default implementations provided by the container.

Implementations of the IdentityStore interface can be used to validate user credentials and retrieve group information. An application can provide its own IdentityStore or use the built in LDAP or Database store.

The HttpAuthenticationMechanism and IdentityStore APIs provide an advantage over container-provided implementations in that they allow an application to control the authentication process, and the identity stores used for authentication.

The SecurityContext API is intended for use by application code to query and interact with the current security context. The specification also provides for default group-to-role mapping, and defines a principal type called CallerPrincipal that can represent the identity of an application caller.

I would argue that any language or platform can be weak and vulnerable if you do not have a thorough understanding of the technology and programming model. In the end, no language or platform is secure from bad programmers. With great power comes great responsibility.

***Project Classes***

Please see attached.

***Screenshots***









