Secure Application Endpoints

**Warning: Do not store your SSL certificates in the GitHub repository.**

* Secure your web application endpoints with valid SSL certificates.
  + For dev environment, you may use the [AWS Certificate Manager](https://aws.amazon.com/certificate-manager/) service to get SSL certificates.
  + For prod environment, you must request an SSL certificate from [Namecheap](https://www.namecheap.com/) or any other SSL vendor except for AWS Certificate Manager, import it into [AWS Certificate Manager](https://aws.amazon.com/certificate-manager/) from your CLI, and then configure your load balancer to use the imported certificate.
    - Command to import certificate must be documented in your README.md file.
* Plain text requests sent to HTTP do not have to be supported.
  + HTTP to HTTPS redirection is not required.
* Traffic from the load balancer to the EC2 instance can use plain text protocol such as HTTP.
* Users should not be able to connect to the EC2 instance directly.

Encrypted EBS Volumes

* All EC2 instances must now be launched with encrypted EBS volumes.
* EBS volumes must be encrypted with Customer managed key created as part of your CloudFormation template.

Encrypted RDS Instances

* RDS instances launched should be encrypted with (a separate) Customer managed key created as part of your CloudFormation template. The encryption key must not be shared with other resources.

Secure Database Connections

* Use Secure Socket Layer (SSL) or Transport Layer Security (TLS) for encrypted connections from your application to the RDS instance.

Documentation

* [Namecheap SSL Certificates](https://www.namecheap.com/security/ssl-certificates/)
* [Using SSL/TLS to Encrypt a Connection to a DB Instance](https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/UsingWithRDS.SSL.html)
* [Importing Certificates into AWS Certificate Manager](https://docs.aws.amazon.com/acm/latest/userguide/import-certificate.html)

Submission

The assignment will be considered late if commits are made to the **main** branch after the due date.

1. All work for this assignment must be done on the feature branch in your fork and merged to main when you are dev complete.
2. The feature and main branches must be in sync.
3. Submit your code from **all** repositories in this assignment. **Read the instructions carefully to create your zip file.**
   1. Create a folder with the naming convention **firstname\_lastname\_neuid**
   2. In the **firstname\_lastname\_neuid**clone all of your GitHub (organization) repositories with the **git clone**command. It is important that you clone the repos so that your commit history and branches are preserved.
   3. Once you have cloned all of your repositories, you will create a zip of the **firstname\_lastname\_neuid\_a\_##** directory. The zip file should be **firstname\_lastname\_neuid\_a\_##.zip** where **##** is the assignment number.
   4. Now unzip the zip file in some other directory and confirm the content of the zip files.
   5. Upload the Zip to this assignment.
4. You are allowed to resubmit. If you think there may be an issue with the ZIP file, feel free to submit it again. Only the latest will be used grading.

Grading Guidelines

The following guidelines are for information only. They are subject to change at the discretion of the instructor and TA. 10% penalty will be imposed if the application crashes due to unhandled exceptions/errors.

Previous Assignment Objectives

* TAs must verify that students have resolved all open issues from the previous assignment(s).

Git Repository Content Check (10% Penalty)

* Check the repository for any IDE-specific files. IDE configuration files must not be in the repository.
  + Verify their .gitignore configuration.
* Check the repository for build artifacts such as .class, .jar, .war files and build, node\_modules directory. None of these should be checked into the repository.
* Check for dependencies. Dependencies from the Maven repository or npm should not be committed to the git repository.

Pre-requisite for Demo (5% Penalty)

* Delete all VPCs you may have created in your demo AWS account.

Application Crash (10% Penalty)

* The application should not crash or return *500 Internal Server Errors* during the demo.

Security Check (15% Penalty)

* Check for hard-coded AWS credentials.
* Check IAM policies that grant access to all (\*) resources.
* Check for IAM roles and policies that have AdministratorAccess attached to them.
* Image deletion must be hard deleted in the S3 bucket.

Web Application Security (30%)

1. Validate that application is only accessible over HTTPS and not HTTP.
2. Check for the HTTPS lock icon in the browser.
3. Verify the issuer of the SSL certificate is [Comodo Certificate Authority (now Sectigo CA)](https://www.namecheap.com/support/knowledgebase/article.aspx/334/38/what-is-certificate-authority-ca).
4. Verify the certificate is stored in AWS Certificate Manager.
5. Validate that the load balancer is using a certificate from AWS Certificate Manager.

Secure Database Connection (10%)

1. Verify that DB connections from the application are secure.

Encrypted EBS Volumes for EC2 instances (30%)

* Verify that the EBS volumes attached to EC2 instances are encrypted using a custom key.

Encrypted RDS Instances (30%)

* Verify that the RDS instance is encrypted using a custom key.
* Verify that the encryption key is not reused for encrypting other resources such as EBS volumes.