

Leon-Hunter-Public-Portfolio

View the Project on [GitHub](https://github.com/) at <https://github.com/>

Click [here](#) to download the most recent `.pdf`

Senior Cloud Native Application Developer

Junior Cloud DevOps Engineer

About

Contact Information

- **Phone:** (302) 312-4489
- **Email:** xleonhunter@gmail.com
- **Website:** <https://git-leon.github.io/Leon-Hunter-Public-Portfolio>
- **LinkedIn:** <https://www.linkedin.com/in/leon-hunter-3230408b>
- **Github Organization Ownership**
 - <https://github.com/curriculeon>
 - <https://github.com/git-leon>

Education

- Delaware State University (2011 - 2015)
- Bachelor of Science (BS)
- Computer Science Major, Mathematics Minor

Summary

Experienced T-shaped Engineer and Technical Instructor with a demonstrated history of managing and up-skilling teams of Junior engineers. Highly skilled in Cloud Native Application Development using Java8, Spring, Mockito, and Angular cumulating more than [2000 total Git contributions in the past year](#). Knowledgeable in Cloud Dev Ops using unittest, JUnit5, Selenium, Docker, Kubernetes, Jenkins, and AWS. Strong information technology professional with a Bachelor of Science (BS) focused in Computer Science, Mathematics Minor from Delaware State University.

Personal Projects

Owner and Creator

Curriculeon: The Online Coding Curriculum

- [Curriculeon](#) is an online curriculum built in [RevealJs](#), [Jekyll](#), and [Github Pages](#), which offers free open-source learning content for the following topics:
 - Java Web Development

- Command Line Scripting
 - GitCli
 - HTML/CSS/JS
 - Python Scripting
 - Cloud Computing & Cloud Infrastructure
 - Development Operations - (DevOps)
- The following is a list in ascending order of delivery date of some of the relevant processes that were covered and applications that were built during a 14-week program in a single lecture (**90 minutes or less**). Demonstrations relied heavily on git, TDD and Cloud Native practices; enforcing version-control, portability, and testability across each of the developer's environments. It should be noted that each application, even the creation of the Github repository, was built in front of a live audience of 25 to 30 developers, while polling for and answering questions.
 - A [simple JUnit Testing application](#) built with Maven using JUnit5 testing paradigms and conventions for testing input and output of algorithms;
 - The intent of building this application for the developers is to demonstrate how to implement proper TDD practices. It is critical that this is presented early to the developer to ensure that each demonstration that follows can be built and then tested appropriately.
 - A [crude database-access application](#) built with Maven using JUnit5 and JDBC to test database connectivity, querying capabilities, and database insertion;
 - The intent of building this application for the developers is to demonstrate how to leverage Java's JDBC API to connect to delete, create, and connect to a new connection. The demonstration included how to switch Driver implementation by modifying the `pom.xml` to specify the JDBC driver.
 - A [scalable database-access application](#) built with Maven using JUnit5 and JDBC to test database connectivity querying capabilities and database insertion
 - The purpose of demonstrating how to build this application is to assist the developers with establishing how to build an application that leverages strictly JDBC and scales easily by adding more POJO-oriented logic.
 - This demonstration will prepare developers for proper application-design and scalability by abiding by SOLID principles and introducing GoF Design Patterns (particularly factories, builders, and singletons).
 - In the days that follow, the developers witness how proper design can scale flexibly. The decision to implement this POJO-oriented logic is to later enforce an entity-driven-design implementation using JPA and again later with a Spring Boot implementation.
 - A [scalable-database access application](#) built with Maven using JUnit5 and JDBC and JPA.
 - The developers witness how the aforementioned application is easily transformed into a JPA project by introducing `persistence.xml` in the `main/resources/META-INF` directory and including a JPA flavor in the `pom.xml`.
 - A [Spring application](#) which exposes basic CRUD operations of a controller managing an entity with implicit @Basic field-types
 - By leveraging the H2-console, the developers witness the entire development lifecycle of a Spring Boot application beginning with empty tables. As an Entity is defined, the developers immediately view the change in the console.
 - The developers benefit from viewing this demonstration by witnessing how Beans, EntityManager, IoC container, and Dependency Injection work in the context of Spring
 - The application is tested and interacted with via Postman

- **Leonium** is an **uber jar** deployed on [packagecloud.io](#) that is used to automate Selenium build configuration and script development.
- The library includes an **embedded .jar** used for dynamically downloading driver binaries
 - This allows this framework to run **GeckoDriver**, **ChromeDriver**, **PhantomDriver**, **HtmlUnitDriver** without downloading any external dependencies
- The library also includes mechanisms for
 - enabling logging for each web-interaction
 - enabling reporting using **ExtentReports** for aggregating results
 - helper classes with built-in intelligent-waits

Owner and Creator

OpenCVL.jar: An OpenCV Wrapper With Haar Cascade Machine Learning

- **OpenCVL** is an **uber jar** deployed on [packagecloud.io](#) that is used to ease implementation of Object Detection Algorithm by leveraging **OpenCV** and **HaarCascadeClassifiers**.
- The library includes mechanisms for
 - a factory class for dynamically fetching haar cascade classifiers
 - face detection interface
 - default face detection implementation with **CascadeClassifier** dependency injection support
 - face detection builder class
 - face detection factory class

Owner and Creator

JFoot.jar: A Greenfoot Wrapper Library

- **Jfoot** is an **uber jar** deployed on [packagecloud.io](#) that is used to make Greenfoot applications with Maven.
- The library includes mechanisms for
 - converting greenfoot applications to maven applications
 - helper interfaces for building games
 - small gravity physics engine

Owner and Creator

ExceptionalFunctionalInterfaces.jar: A decoration of Java8 FunctionalInterfaces

- **Exceptional Functional Interfaces** is **.jar** deployed on [packagecloud.io](#) that is used to express code that explicitly throws an **Exception**, without ever explicitly handling it by deferring the **try/catch** to the respective **FunctionalInterface**'s static **tryInvoke** method.
- With the advent of **Java 8 Lambdas** and **Method References**, functional programming in java has become increasingly popular. Unfortunately, Java 8 does not account for creating lambda expressions which may throw an **Exception**. This shortcoming has caused a great deal of frustration when developing an exceptional-codebase which should be expressed functionally rather than with object orientation. This library aims to help remedy the deficiencies of exceptional expressions in Java 8 functional programming.

Experience:

Remote Lead Technical Instructor

Per Scholas: Cloud DevOps Engineering Course

November 2020 – February 2021; Boston, MA.

- Remotely leading a group of 30 aspiring engineers through a Cloud DevOps environment.
- teaching the following topics through live-demonstrations to an audience of 25 in daily Zoom meetings:
 - GitOps, Git, Shell scripting, SQL / DBMS, TDD, Python Scripting
 - Cloud Infrastructure, Cloud Security
 - CI / CD, Nginx, Build Management
 - Deployment and Jenkins Pipeline
 - Docker and Kubernetes
 - AWS Terraform, AWS ECS, AWS EKS, AWS Lightsail, AWS EC2, AWS Cloud9

Remote Lead Technical Instructor

Talent Path: Full Stack Java Engineering Course

November 2020 – December 2020

- Using Git, Zoom, Slack to manage a group of developers through the development of Java Web-applications.
- Leveraging Curriculeon [curriculum lectures](#) and [curriculum exercises](#) to launch learners into an immediately immersive development environment to enforce a deploy-on-day-1 culture.
- Learners receive morning lectures from the curriculum, then exercises from the curriculum to complement the lecture.
- The following is a list in ascending order of delivery date of some of the relevant content presented to learners.
 - A [Java assessment](#) built with Maven using JUnit5 testing paradigms and conventions for testing input and output of algorithms;
 - The intent of building this application for the developers is to demonstrate how to implement proper TDD practices. It is critical that this is presented early to the developer to ensure that each demonstration that follows can be built and then tested appropriately. This also gives the instructor the opportunity to gauge the classes aptitude. With this group, it was the case that they had strong understanding of the Spring framework, but poor understanding of Object Orientation, Design Principles, and Design Patterns. After making this discovery, the focus was to upskill the learners in the aforementioned areas.
 - A [Spring application](#) which exposes basic CRUD operations of a controller managing an entity with implicit @Basic field-types
 - By leveraging the H2-console, the developers witness the entire development lifecycle of a Spring Boot application beginning with empty tables. As an Entity is defined, the developers immediately view the change in the console.
 - The developers benefits from viewing this demonstration by witnessing how Beans, EntityManager, IoC container, and Dependency Injection work in the context of Spring
 - The application is tested and interacted with via Postman
 - A [system of classes](#) enforcing several singleton design-pattern implementations, repository design pattern implementation, and decorator design pattern implementation.

May 2013 – May 2014; Dover, DE.

- The objective of this research was to develop a system that would aid in the prevention of catastrophic medical events through persistent intelligent monitoring and early-warning alerting.
- Created interactive electronics using Arduino open-source prototyping platform.
- Coded in C++ to manipulate Arduino microcontroller and eHealth sensor shield.
- [Presented research](#) for 2014 Science and Math Investigative Learning Experiences Program (SMILE) Undergraduate Summer Research Symposium.
- Presented at 2014 ERN (Emerging Researchers National) conference.

Programming Tutor

Delaware State University

January 2013 – May 2013; Dover, DE.

- Aided learners with completing programs developed in java.
- Assisted learners with conceptualizing and understanding programming logic.

Project maintained by **Git-Leon**

Hosted on GitHub Pages — Theme by **orderedlist**