Chi Calvin Nguyen . Jarvis Consulting

I am a dedicated team player who is inquisitive, punctual, and goes by the book. I recently graduated with an Ontario College Advanced Diploma from the Computer Programming & Analysis program at George Brown College. During my time at George Brown, I've learned technologies such as React, Node.js, Angular, MongoDB, Spring, and ASP.NET by building projects like a front-end application that connects to an open-source weather API or full-stack applications where we built both the front and backend while following the Agile project management methodology. I've also gained other skills outside George Brown that are crucial to a workplace. My ability to communicate and act as a team player established itself throughout my time as a bartender/server. I worked in a rough fast-paced environment that required me to be on my toes, give the best possible service to customers, and work within a tight-knit team. I am quite a hobbyist with hobbies such as brewing beer, where I experiment with recipes to get the best possible batch, which is why I also like tech because I constantly learn, test, and create software. Overall, I'm hoping for an opportunity in the software development industry where I can prove myself as a team member.

Skills

Proficient: JavaScript, Angular/React, NodeJS, Java, Maven, Git, RDBMS/SQL

Competent: Spring, Docker, Python, Agile/Scrum, REST APIs, HTML/CSS

Familiar: Linux/Bash, C#, ASP.NET, PHP, Swift

Jarvis Projects

Project source code: https://github.com/Jarvis-Consulting-Group/jarvis_data_eng-C-CalvinNguyen

Linux Cluster Monitor [GitHub]: Collects the hardware specifications and usage for the Linux nodes within a cluster and stores this data inside the database. Docker manages a container running the PostgreSQL database, and this container is set up using a Bash script (psql_docker.sh). The other Bash scripts (hardware_info.sh and hardware_usage.sh) collect and store the data in the database using Linux commands like lscpu, vmstat, and psql. Finally, a cron job schedules the script's execution every minute and stores the data in the PostgreSQL database within the Docker container.

SQL: Manipulate an Existing Database [GitHub]: Familiarize ourselves with concepts of relational databases, relational database management systems, structured query language, optimizations, and data models. To grasp these concepts, I worked on manipulating existing data using an IDE called pgAdmin and manually compared my query results with the expected results given. The file clubdata.sql, which contained the existing data, was inserted into a PostgreSQL database deployed within a Docker container.

Core Java Apps [GitHub]:

- Twitter App: Allows us to perform operations that manage tweets; the Core Java Twitter App lets us post, find, and delete tweets all within a command line interface. The program follows the model-view-controller architecture design, the controller/service handles inputs and business logic, the model manages objects and data retrieved from the Twitter API, and the view prints out a tweet in a JSON format. To implement these designs, we used technologies such as Twitter RESTful API V2 that utilizes HTTP methods to perform the operations on the Twitter servers. Maven is another technology that handles the project's dependencies like the Apache HttpClient package that allows the app to send HTTP requests, the OAuth signpost package that enables the app to add OAuth authentication to the requests, and the fasterXML Jackson Core package that maps JSON properties to the tweet object. Integration and unit testing were utilized with the libraries Mockito and JUnit4 to test the application. Lastly, the Twitter app was deployed within a Docker container with Twitter API OAuth keys set as environment variables within the Docker container.
- JDBC App: Familiarize ourselves with design patterns like Data Access Object and Repository patterns while building an application that uses the Core Java JDBC API to perform CRUD (CREATE, READ, UPDATE, and DELETE) operations on an existing dataset within a database. Used the JDBC API within the java application; the JDBC API consists of two packages (java.sql and javax.sql) for operating with a database when used in conjunction with a JDBC driver. Maven managed the application's dependencies, like the specific JDBC driver that acts as a layer between the application using JDBC methods and the PostgreSQL database. The PostgreSQL database was deployed on a Docker container using the Postgres base image from Docker Hub, and the psql command inserted the data set into the database. Testing the application involved using the SLF4J logger to display records when retrieving data from the database and confirming database changes by accessing it from a terminal using psql.

- Grep App: Designed the Java Grep app to replicate the egrep Linux command with the recursive option enabled so that it reads all files within the directory or subdirectory and includes a redirection operator to write matched lines to a file (egrep -r [regex/pattern] [sourceFile] > [targetFile]). Implementation involved using core Java packages like stream API and lambda functions to iterate through directories, lines from a file or a string stream, and java.io, which reads and writes text data from or to the files. Maven builds the project, packages it into a JAR file, and manages dependencies like SLF4J. Debug logging used SLF4J, and unit testing used JUnit4, which aided in manually testing the app by displaying files and matched lines. For deploying, we created a docker image using a Dockerfile, distributed it onto the Docker hub, and ran it on a docker container.

Springboot App [GitHub]: Replacing a legacy monolithic trading system with issues in scalability and maintainability; the Spring Boot app is a proof-of-concept backend REST API designed to resolve these problems by following the microservices architectural design. The microservices architectural design is scalable as more microservices are deployed and is maintainable because downtime or errors in one microservice should not affect another. The trading system allows clients to create accounts, deposit or withdraw funds and purchase or sell stocks. The technologies used to develop the application are Docker for the virtualized isolated runtime environment, a PostgresSQL database to persist the data, Maven manages the project's dependencies, maintains a consistent project structure, and builds/packages the project. The Spring framework is the main part of the project, and it does things like providing a default configuration for Maven and managing the lifecycles of the class components. Spring also adds dependencies like the Apache Tomcat Web Servlet that extends the Java Servlet API, processes HTTP requests, and uses Spring annotations to select the appropriate controller/method based on the path. Another dependency is the Spring JDBC which abstracts the JDBC API and manages the creation/release of connections without the developer having to hardcode them. The project testing involved integration and unit testing with testing libraries like Junit4, Mockito for creating Mock objects, and Spring starter test that utilizes a test config class to create bean objects for the tests. Finally, the application is packaged and deployed within two docker images, one for the PostgresSQL database with the relational database tables and the trading application itself. The part that makes this a microservice is since the backend REST API does not manage any states, a balance loader may manage multiple instances of the backend that acts independently from one another.

Highlighted Projects

Ionic/Angular Mobile Music Player Application & Streaming System [GitHub]: Plays local audio files on Android mobile devices, and the app can also create an account to upload and stream files from the backend server/system. MongoDB stores account information once an account is registered. The front-end/client-side mobile application uses Angular/(Ionic); it can read local audio files, stores their metadata into an SQLite local storage, play these files, and connect to the backend server once authenticated. The NodeJS backend server allows users to create accounts, upload audio files onto the server, and stream it back to the user.

Spring Cookbook App [GitHub]: Allows users to create, manage, search, share, and favorite recipes on the application. Includes other features like planning a meal schedule or creating a shopping list from the ingredient's checklist in a recipe. The project uses Spring for the backend controllers and services, Thymeleaf for the front-end templating engine in HTML/CSS files, and H2 Database that is integrated into Spring automatically through a SQL file.

Professional Experiences

Software Developer, Jarvis (2023-present): I joined as a Junior Software Developer to consult and aid in developing Jarvis' client's software solutions. To do this, I learned new technologies while working on projects at Jarvis in an Agile/Scrum environment where there were daily scrum meetings in the morning to discuss project progress and sprint meetings every two weeks. The technologies learned at Jarvis are specially selected as they are common within the professional tech workplace environment, and an example of these technologies are Linux, SQL, Java, Spring, etc.

Bartender/Server, Hazel's Diner (2018-2020): Worked as a team in a fast-paced environment where communication with coworkers and customers was the key to fulfilling orders and providing excellent service within a given time frame. Responsibilities of the position included bussing food, mixing drinks, cleaning tables, managing dishwashers, and other tasks. I gained certifications such as Smart Serve and First Aid for the position. I learned more about the restaurant's workflow when working alongside the team, and this diversified my responsibilities within the restaurant.

Education

George Brown College (2019-2022), Ontario College Advanced Diploma, T127 Computer Programmer Analyst - School of Computer Technology - Dean's List (2019-2022): Was on the Dean's list by achieving a term GPA above 3.5 - GPA: 3.83/4.0

Miscellaneous

- Co-operative games like board games or online multiplayer video games.
- Brewing beer (Pale ales, IPAs, Stouts)
- Leisure ice skating, watching hockey (on TV or in-person) and playing street hockey or pickup hockey (shinny).