REVATURE

Nicolas Hernandez

Professional Summary

I am an engineer across multiple fields, with an affinity & motivation toward software engineering. I come from a mechanical engineering education, though I programmed Java in some of my spare time and was always hands-on with automation logic & software. My current goal is to further grow my skillset in software, a field where I can always find more motivation to get more done.

I'm newly-versed in a growing set of end-to-end Web development technologies, with which I've built and deployed full-stack Web projects. To illustrate my full-stack skill set, I've deployed Web applications over AWS EC2 instances, I've interfaced with databases using Portgres and JDBC, I've handled application logic & testing in Java & TypeScript React, and I've handled front-facing Web requests & responses with Java Servlets and Javascript-supplemented HTML pages. I've also presented my completed projects before panels of interested parties.

Certifications

Education

Bachelor's Degree in Mechanical Engineering

Georgia Institute of Technology, 2017

Associate's Degree in Physics

Dalton State College, 2014

Technical Skill Summary

Java	Maven, REST API, Servlets, Java (Core), JSPs, JUnit Test Automation
Database Technologies	Hibernate XML-Based & JPA-based ORM, Spring ORM, SQL (Core), JDBC, PL/pgSQL Procedural SQL Programming, PostgreSQL
Front-End Standard Web Technologies	HTML, CSS, JavaScript, AJAX, JSPs
Infrastructure, Deployment, and Web Services	Amazon AWS, Spring Boot, Amazon RDS Database Deployment & Management, Git, Apache

	Tomcat Java Server Technologies, Jenkins Web Deployment over Amazon EC2
Supporting APIs & Libraries	Spring MVC, Spring ORM, Hibernate ORM, Log4J, Spring AOP, Spring Data
Front-End Specialized Web Technologies	React, Redux, TypeScript, SCSS, ReactStrap, Jest Front-End Testing

Professional Experience

Revature **Sep-2020 to Present**

Client Engagement Portal

The Client Engagement Portal is a web application that was constructed in order to provide a means for clients to easily interact with Revature batches that are mapped to them. The hope is that this application would lead to a higher degree of customer involvement and lead to better customer service. Some of the main features of this application are centered around allowing the client to: access batches mapped to them, see the progress of the batch, plan interventions with batches, and indicate that they would like to hire more talent. These users should also be able to notify Revature admins of these various needs through the application. Admins should be able to view these client requests and contact information for the client.

The front end was constructed primarily using React, Redux, Jest/Enzyme for testing, and Material UI for styling. The back end was a Spring Boot application that used Maven, JUnit 4 for testing, Swagger for documentation, and an in-memory H2 database.

Responsibilities:

I took the front-end captain role, which I began by taking our day-one front-end design and creating several static React pages as a boilerplate for the front-end team. We used React Redux with TypeScript to build our front-end application, and we implemented several dependencies for styling, testing and authentication. As we progressed in the project, I did the following:

- Created further components as needed.
- Reviewed & revised my teammates' front-end changes.
- Oversaw code merges in Git.
- Connected components to backend APIs & the Redux store
- Communicated with other teams to inform the needs of our application.
- Managed our AWS Cognito user pool to accomplish frontend-based authentication.
- Documented code in my components.

- Supported design changes that occurred throughout the development process.

Environment:

Java, Spring, PostgreSQL, Maven, Amazon Web Services, JavaScript, HTML, CSS, React, Bootstrap, Jenkins, REST, Git, Hibernate, Docker

Revature Social Network (React) v2

In Revature's Social Network everyone is friends with everyone else. Users can register, login to the application, and start

sharing multimedia with everyone. Registered users are allowed to modify their personal information and upload

their profile pictures. The application provides a search feature that allows users to search out friends and look at their

profiles. Users are provided with a "feed", in which they can see what everyone is posting and like posts.

Users can access and use the application via an interactive client-side single paged application that stores and

retrieves multimedia using AWS S3 and consumes a RESTful web service that provides business logic and access to a

database.

Responsibilities:

I took responsibility for writing the database communication layer & object mappings in the back end, I established Redux state & front-to-backend communication for the front end, and I maintained consistent correspondence & teamwork with the four other developers on the team.

We built the project from the ground up using Spring & Hibernate in the back and React/Redux with TypeScript in the front. For some of our final metrics:

- Wrote approx. 1500 lines of Java code, incl. JPA annotations / Spring ORM to abstract away much of the JDBC code used in previous projects.
- Included over 1200 lines of TypeScript / JavaScript using React with Redux.
- Tested components using Jest.
- Styled the front end using SCSS.

- Measured approx. 800-ms reload-free update time for responsive frontend actions such as making posts and updating your profile picture.
- Minimized the need for page changes; hot-swapped components to allow feed updates, user profiles, and profile editing all from the same page.
- Hosted all necessary content on 3 main page components.
- Integrated Amazon S3 to serve custom profile pictures.

The Github repos for this project are here:

Back End: https://github.com/Hernandezn/Clipper

Front End: https://github.com/Hernandezn/ClipperFront

Environment:

Spring, Java, SQL, Hibernate, HTML, CSS, JavaScript, Log4J, JUnit, React, Redux, TypeScript, Agile-Scrum

Expense Reimbursement System (core tech)

The Expense Reimbursement System (ERS) will manage the process of reimbursing employees for expenses incurred while on company time. All employees in the company can login and submit requests for reimbursement and view their past tickets and pending requests. Finance managers can log in and view all reimbursement requests and past history for all employees in the company. Finance managers are authorized to approve and deny requests for expense reimbursement.

Responsibilities:

I created, tested, and deployed the application from the ground up within a two-week timeframe. I ultimately made my design customer-focused for ease of use, minimizing button presses, wait times, and redirects for all of the required Web app functions.

This was a full-stack project, so I set up & interfaced with an Amazon RDS database in the back, I handled information & requests via Java servlets & JDBC, and I served HTML/CSS/JavaScript pages at the front end. As for some of my metrics, I did the following:

- Coded over 3400 lines of Java code.

REVATURE

- Created more than 500 additional lines of code for frontend technologies.
- Tested with 70% test automation coverage with JUnit.
- Measured 3900-ms initial build time for the server.
- Included frontend-based (functionally immediate) UI updates where possible.
- Confirmed less than 2 seconds of request-response time for backend data retrieval.
- Minimized number of pages for ease of use; 2 pages served all content via DOM manipulation and AJAX requests.

The Github repo for this project is here: https://github.com/Hernandezn/ReimbursementProject

Environment:

JavaScript, HTML, CSS, AJAX, SQL, Java, Servlets, JDBC, PostgreSQL, Agile-Scrum

Gestamp Chattanooga 1 **Dec-2018 to Mar-2020**

Kobayashi America **Oct-2017 to Dec-2018**

Internet of Things Hardware & Software Deployment

Created a condition monitoring system that tracked power usage, input & output air pressure, vibration, and motor output on a production machine. This machine was initially entirely isolated in terms of connectivity. Data was output to a local intranet address

Responsibilities:

Studied & designed a working Internet of Things pipeline from the ground up. This was done in a small single factory as an overarching background project over the course of a year, so I had primary responsibility for the design and implementation of the entire project.

I was responsible for:

- fleshing out the then-vague "Internet of Things" term into a concrete production-capable system

- seeking input across departments to help specify our design
- shopping around & meeting vendors to specify which hardware to buy
- designing our system around this hardware
- training & reading technical documentation for the new hardware & software
- finding our critical monitoring areas with electrical schematics & machine analysis (since all verbal documentation was in Japanese)
- working with a team to connect & hang all wires for a local intranet network
- connecting all sensors & wires to the critical monitoring points on the actual machine
- hanging, assembling, and programming the PAC system that was taking inputs from our sensors to record data
- programming a frontend that took data from the PAC and displayed it in a business-usable format, and
- managing this project in the background while I handled more urgent team-oriented engineering & maintenance projects in the foreground.

Environment:

PLC Industrial Controller Programming & Wiring, Ladder Logic, Opto22 OptoScript, Electrical Schematic Reading, Electronic Wiring & Design, Circuit & Signal Analysis, 4-20mA & 0-10V/0-5V Sensors (for pressure, electrical current, vibration, and rotation using encoders), HTML/CSS/JavaScript frontend, MS Azure DB, DraftSight CAD

Georgia Tech RoboJackets **Sep-2014 to Dec-2014**