Brandon John Etchison
402-403-8155
BrandonJava95662@gmail.com

Brandon John Etchison

Senior Application Developer/Tech Team Lead
7510 Sun Valley Dr. Omaha NE, 68157

SUMMARY:

11+ years of Experience in full-stack application software life-cycle development involving Analysis, Design, Development, Testing, Implementation, and maintenance of application software in a web-based environment, distributed n-tier, client/server, MVC, peer-to-peer, cloud-based, and broker architectures.

- Expertise in developing applications and Internet and Intranet projects in all phases of development, including analysis, design, coding, testing, implementation, troubleshooting, maintenance, and security.
- Advanced experience in the analysis of software requirement specifications/development and execution of test plans.
- Experience developing web applications using **Java/J2EE, Hibernate, IntelliJ, JSP, Servlets, Springboot, Spring, JavaScript, MySQL, Oracle, SQuirreL, SQLDeveloper, HTML 5, JDBC, multithreading, collections, and Java design patterns.**
- Dedicated focus on microservice architecture using **Java 8, Springboot, PCF, AWS, rabbitMQ, DB2(LUW), Cassandra, Spring Reactive, AppDynamics, and Splunk**
- Expertise in writing **SQL, PL/SQL**, stored procedures, triggers, functions, optimizing, and performing tuning.
- Knowledge in implementing Service-Oriented Architectures (SOA) using XML-based Web Services (**SOAP/ WSDL**) and publishing web services using code-based and schema-based approaches, as well as full **RESTFUL** services.
- Expertise in converting XML-based web services into Restful.
- Extensively worked with the **Spring/Springboot Frameworks and J2EE** design patterns, such as Session Facade, Data Access Object, Transfer Object, Business Delegate, Service Locator, Intercepting Filter, and MVC.
- Experience analyzing and reviewing client system resources and conducting business analysis. Creating, analyzing, and reviewing **Use Cases, Class Diagrams, and Sequence Diagrams using UML/ Rational Rose.**
- Expert working knowledge of build tools like **Maven**, unit testing with **JUnit**, and logging tools like **Log4j2.**
- Experience in version control management tools like **CVS, Harvest, SVN, and GIT**.
- Experienced in multi-tier web components based on J2EE architecture (JSPs/Servlets).
- A self-starting team player who multitasks, has **strong problem-solving skills**, and is **constantly driven to exceed expectations**.
- A **self-motivated** professional and **natural communicator**.Possessing **great technical, initiating, leadership, and problem-solving skills** and have proven to be a **good team player.**

Brandon John Etchison
402-403-8155
BrandonJava95662@gmail.com

- Expert in solving technical problems and providing i**n-depth analysis** and resolutions of technical issues.
- 3.5 years experience working remotely using communication tools like **Microsoft Teams, Zoom, and Google Meet**.
- 5+ years experience with security tools and training solutions such as **Coverity, Seeker, and Fortify** as well as **Secure Code Warrior**.
- 4+ years of **managing and leading Agile teams** working with IT professionals, Business professionals, and clients.
- Expertise using Scrum tools and concepts like **JIRA** or **Kanban** as well as standard Agile procedures such as **Sprints, Sprint retrospectives, Sprint planning, and Sprint reviews.**
- Expertise with documentation tools like **Confluence, Google Tool Suite, Microsoft Tool Suite, and Sharepoint.**
- Expertise with diagram tools such as **LucidChart, Draw.io, and Visio**.

Work History:

FirstData/Fiserv: As an Application Analyst and an IT Manager from July 2014 - Dec 23rd, 2021
7302-7330 Pacific Street Omaha, NE 68114

Dovel Technologies: As a Senior Application Developer/Tech Team Lead from Dec 27th, 2021 - Present
7901 Jones Branch Drive,
Suite 600
McLean VA 22102

Awards:

I have received the highest award Fiserv offers to its employees two times.

Education:

Masters of Divinity(Bible Focus) Planned graduation date 01/2027

Global Awakening Theological Seminary
1451 Clark St, Mechanicsburg, PA 17055

Bachelor of Science in Software Development Life Cycle of Video Games & Information Technology

Westwood College Online
3250 Wilshire Blvd, Los Angeles, CA 90010

Brandon John Etchison
402-403-8155
BrandonJava95662@gmail.com

Technical Skills:

Operating Systems:
Windows, Linux, AWS Workspace

Software Development Processes:
Agile (SCRUM, D3, Extreme Programming) and Ad Hoc Waterfall

Service Technology:
Jackson, Jersey, Resttemplate, and Reactive services

Scripting Language:
JavaScript, XML, HTML 5, ZK

Servers:
Tomcat, Websphere, Websphere Liberty, Netty, AWS, and PCF

Frameworks:
Struts, Spring, Springboot, ZK

Object Relational Mapping:
Hibernate

Databases & Related Tools:
Oracle 11g, MySQL, SQuirreL SQL, SQL Developer

Tools/IDE:
Eclipse, Visual Studio 2010, NetBeans, Rapid Application Developer, IntelliJ

Modeling Tools:
Visio, Confluence

Programming languages:
Java

Middleware / Distributed Technologies:
J2EE, JMS, Hibernate, Web Services.

Server-side programming:
Java Servlets, JSP.

Version Control:
Git, SVN, CVS, Harvest

Deployment Tools:
Jenkins, Concourse, Rundeck

Organization Tools:
JIRA, Confluence, Microsoft Teams, Zoom, Google Meet, Sharepoint

Recent Work:

Dovel/Guidehouse

OverArching Project:
Fixing security vulnerabilities against several applications with compromised frontend and backend code.

Description:
Using Coverity and Seeker to discover and fix application vulnerabilities with Java.

Duration: Dec 2021 - Present

Team Size: 11
7 Developers

Brandon John Etchison
402-403-8155
BrandonJava95662@gmail.com

2 QA Testers
1 Scrum leader
1 Project Owner

Responsibilities:

- Learned 40 legacy applications using many different UI frameworks; Angular 1.0, react, Primefaces, Vaadin, Dojo, just Jquery, and several supporting utility applications with no front end. As well as backend frameworks: Struts 1.0, Spring boot, Vaadin, JSF.
- I designed a game plan for fixing each JQuery page depending on the difficulty of the upgrade. Some were simple, and just the number had to be changed, while others required much more work as functions no longer existed in the newer code.
- Worked through legacy Javascript to piece together how the GMM application worked.
- I worked with the team to upgrade Primefaces from version 4 to 10 to resolve the JQuery finding for the Property and PTMS apps.
- Met with the team often to provide direction on CSS and other UI changes.
- Developed test plans for the client testing team to use to test our specific changes.
- Worked closely with the QA Testers to ensure our test plans were thorough and accurate, testing all functionality on a given UI page both positive and negative.
- Interviewed several new team members as the team changed over time.
- Was the primary reviewer on all Git code requests
- Worked closely with the client to review their requirements and pieced together a roadmap for the team.
- Met together for daily SCRUM meetings to follow progress on JIRA tasks and make sure our sprints stayed lined up with our timeline.
- Met weekly for Sprint planning and bi-weekly for Sprint review and Sprint Retrospectives.
- Deployed code in Dev/QA and worked with the team to find and destroy all bugs.
- Worked with the client in their internal testing to handle remaining bugs.
- Deployed code to production on the targeted date.
- Used Splunk to dig through error logs and fix problems with the legacy system.
- I created documents covering useful processes, procedures, and application setups for the team, including employee onboarding, release process, version upgrade steps, Git merge steps, etc.
- Researched Owasp findings for upcoming Java security fixes.
- Trained team on different security findings, such as injection attacks.
- Worked with Architecture to improve the CI/CD process.
- Worked with Architecture to improve code line coverage in projects.
- Worked with the Business and Architecture teams to design the process for full remediation of findings and process for handling false positives.

FirstData/Fiserv

OverArching Project: Business Services Layer

Role: Backend Developer/Technical Team Lead/Architect understudy

Brandon John Etchison
402-403-8155
BrandonJava95662@gmail.com

Description:
Create a layer of services that sits on top of the company's core services and provides orchestration and business logic to front-end applications on a client-by-client basis.

Project 1: Balance Transfer Services for BSL

Role: Backend Developer/Technical Team Lead/Architect understudy

Description:
Create a service that allows the client to perform balance transfer and admin functionality. This service was specifically created to replace legacy code that was unsupported in the company.

Duration: Jan 2020 - July 2020

Deadline: July 2020

Delivery: July 2020

Team Size: 4
2 interns
1 onshore
1 offshore

Responsibilities:

- Worked with the Architecture team to fully develop the microservice platform from user to database including all security features and monitoring that would be in place.
- Created flow diagrams and architecture diagrams for application.
- Created JIRA tasks to manage the teams work as we built the app from the ground up
- Mentored 2 interns and helped them program various pieces of the application through paired programming.
- Review legacy applications to piece together requirements.
- Was the primary reviewer on all Git code request.
- Designed the Database table structure based off of requirements using Confluence to create moc diagrams of each table.
- Used Java 8 and Spring Data to develop Repository classes to access the database.
- Using Java 8 I implemented the initial structure and designed the microservice platform for the code.
- Implemented service validation using both JWT and Mutual Auth cert exchange through Data Power. The Java code checks for a valid signature in the JWT and rejects the request if the signature is invalid sending a bad token response.
- Created a LoadBalancing solution that accepts requests from the internally built proxy on top of the validation to ensure data security.
- Used Java 8 to create a suite of Queue classes that connect to RabbitMQ. Viewed messages sent through the RabbitMQ UI.

Brandon John Etchison
402-403-8155
BrandonJava95662@gmail.com

- Used the @Async annotation to ensure messages were sent asynchronous to RabbitMQ.
- Configured PCF to set up RabbitMQ and AppDynamics then included these in the yaml configuration/deployment files.
- Using Spring Gateway I created a simple application that routes all requests to the proper application on PCF based on the application name in the URL.
- Added Springboot Actuator to the Gateway application in order to monitor data sent through the request.
- Implemented fail safe and Circuit Breaker Patterns in the Gateway application using Resillence4J and monitored those through the provided dashboard in Resillence4J.
- Created concourse pipeline requiring 90 code and branch coverage to build the application and rejecting merge requests if the application did not have a successful build.
- Switched to Jenkins pipeline with the same requirements.
- Created a suite of postman tests that get run through Newman every build/deployment that also fails the build if a test fails.
- Used Junit 5 to create all unit tests keeping the code 90% covered and validating coverage using Jacoco during Maven builds.
- Worked with Mainframe team to get all new tables built and users permissioned
- Using Java 8 and Spring Data I setup the connection to DB2.
- Opened all firewalls for all environments between DB and PCF.
- Created Rundeck job to deploy code to PCF.
- Used Java 8 and Spring AOP to create @Around Logic that contained a logger for all services based on an annotation added to the service. This service used Java Streams to parse the request, then using The Bouncy Castle API I encrypted all PCI and PII data before logging the request to Splunk. The logger also had a timer feature so we could keep track of how fast the requests/responses were happening.
- Created CRUD services for DB using SpringBoot.
- Using Spring Reactive I created a client that could call other web services and respond to the request with custom RestTemplate objects.
- Wrapped existing Balance Transfer service and successfully called it using Spring Reactive to return a RestTemplate object.
- Worked closely with the client to go through their requirements and pieced together a roadmap for the team.
- Worked with the team to add all business logic to the code using Java 8 and separating all tasks using JIRA.
- Met together for daily SCRUM meetings to follow progress on JIRA tasks and make sure our sprints stayed lined up with our timeline.
- Deployed code in Dev/QA and worked with the UI team to find and destroy all bugs.
- Deployed code to Cat and handed it off to the client for internal testing.
- Worked with the client in their internal testing to handle remaining bugs.
- Deployed code to production on the targeted date.

Brandon John Etchison
402-403-8155
BrandonJava95662@gmail.com

- Monitored code through AppDynamics and Splunk for the first Month after Prod seeing zero errors and then added alerting for the expected error types.
- Created 160+ documents covering the full application, including things such as; employee onboarding, service flow charts, Custom Exception codes, detailed service descriptions for all fields in the services, service request/response object, and more.

Environment: Java 8, J2EE, Spring, Springboot, Spring Data, IntelliJ, PCF, HTTP, Log4J2, Splunk, Appdynamics, RabbitMQ, DB2(LUW), Cassandra, Datapower, AOP, Resillence4J, Spring Reactive, Voltage, Postman, Newman, Junit, find-bugs, Jacoco, and checkstyles

Project 2: Payments on BSL

Role: Backend Developer/Technical Team Lead/Architect understudy

Description:
Convert legacy SOAP payment services to Restful services using Datapower.

Duration: May 2020 - July 2020

Deadline: July 2020

Delivery: Aug 2020

Team Size: 2
1 onshore
1 offshore

Responsibilities:

- This project was on a smaller timeline because the previous project laid all the groundwork and we didn't need to duplicate a lot of the efforts. All security. Gateways, filters, firewalls, etc had already been taken care of and we just started building new services off of the platform we created, which was the whole point of the platform to begin with.
- Dug through and fully comprehended Legacy code to get requirements for the project.
- Created Key Value JSON objects to store client specific payment configurations in Cassandra.
- Using Spring Cache and HazelCast I created logic within the code to cache the client config objects based on the key provided within the JWT token. Then I added a way to clear the cache in the Gateway application. This was needed because the cassandra connection was a bit slow and could add 1-1.5s to a service call.
- Created flow diagrams and architecture diagrams for application.
- Created JIRA tasks to manage the teams work .
- Reviewed all Git code request.

Brandon John Etchison
402-403-8155
BrandonJava95662@gmail.com

- Working closely with Payments team, redesigning all 27 of the SOAP services to be simple REST calls where the front end team only needed to know about a few simple fields.
- Using GSON libraries created JSON requests to send through Datapower with all of the information in the configs and all of the required fields that the Payments team needed. This allowed the front end to send objects with only 2-3 fields when the request actually needed more like 30-40.
- Using Java 8 and the previously created Reactive client I called Registration services to validate if a user was registered, based on an optional registration Id provided in the request, and get all valid information for a payment request. If the user was not registered I created separate logic to allow for one time payments and populate the request with the needed data.
- Created a suite of postman tests that get run through Newman every build/deployment.
- Created a solution that determines what client logic to use based on a value provided in the security token using the Java Reflect libraries.
- Created Rundeck job to deploy code to PCF.
- Set up monitoring and alerting based on previous work.
- Went through client requirements and pieced together a roadmap for the custom logic.
- Implemented all business logic to the code.
- Deployed code in Dev/QA and worked with the UI team to find and destroy all bugs.
- Deployed code to Cat and handed it off to the client for internal testing.
- Worked with the client in their internal testing to handle remaining bugs.
- Deployed code to production on the targeted date.
- Monitored code through AppDynamics and Splunk for the first Month discovered 2 issues missed in development 1 based on configs that was an issue with how the payments team handled historical data, the second issue was a timeout issue with the services also related to the historical data of a client being too large.
- Fixed both issues and deployed to Prod again 1 week after the due date. Haven't had an issue since.
- Created 50+ documents covering the full application, including things such as service flow charts, Custom Exception codes, detailed service descriptions for all fields in the services, service request/response object, and more.

FirstData/Fiserv

Project: Enterprise Security Systems

Role: Backend Developer/Analysist

Description:

This is a security portal that handles user management, user authentication, and user authorization for the applications it protects. I was on this application for several years and had multiple projects.

Brandon John Etchison
402-403-8155
BrandonJava95662@gmail.com

Duration: Sept 2013 - Aug 2016

Team Size: 3
All onshore

Starting Environment: Java 4, J2EE, Spring 2, Spring Security 1, Hibernate 1, Spring Quartz, ZK 3.4, Rapid Application Developer(RAD), Websphere 4.1.1, Log4J, Oracle 7, Junit 2

Final Environment: Java 7 due to Websphere limitation, but it was compatible with Java 8, Spring 4.1.1, Spring Security 3.1.2, Hibernate 2.8, Spring Quartz, ZK 7.2, RAD, Websphere 8.5.5, Log4J2, Oracle 12, Junit 4, Splunk, Appdynamics, Datapower, Fortify

Responsibilities:

- First project I ever had on this platform was to upgrade to Websphere 8.5.5. Originally the platform was going to be replaced and we only had 3 applications that were using it.
- Replaced config files and got Webshpere 8.5.5 working in 2 weeks. Had 4 months to complete the project. Was told to use the remaining time to upgrade the open source.
- Upgraded Java version to Java 8. Later reduced it to Java 7 because Websphere 8.5.5 didn't support Java 8 at that time, but the code was compliant with Java 8 even though it didn't leverage any of the features.
- Upgraded Spring and Spring Security to the current versions. This involved a lot of code and library changes as well as some config changes and application context changes.
- Upgraded Hibernate to the current version, thankfully this change was pretty easy and the only thing that I had to mess with was Websphere configuration settings. The ORM files all remained the same.
- Upgraded the UI to the current version of ZK, this was probably the hardest part of all of these upgrades as most of the UI patterns used in 3.4 were non existent in 7.2 and with the lack of unit testing we were really scared about breaking things in production for the working applications. But we didn't let fear stop us and we dealt with UI bugs as they came up, including working with the team in Thailand that developed ZK and really learning the application from the ground up. Even though it is tightly coupled I am a pretty big fan of this UI now because of its unique integration with Java.
- Upgraded to Log4J2.
- Added Splunk logging to the application.
- Added the application to Appdynamics.
- Upgraded to Junit 4.
- Created an in memory HSQLDB and then proceeded to create 100s of tables and 1500 Junit tests to get the code fully covered.
- Created a DR environment for the application and all applications on ESS.
- For a short time I monitored another team that was writing an application that was supposed to replace ESS. I reported back to the manager they were missing a lot of

Brandon John Etchison
402-403-8155
BrandonJava95662@gmail.com

critical functionality and the application was bloated and would overall be unsuccessful. Almost 1 year later they listened to me after the portal being built had finally gotten its first application put on it. The application was riddled with bugs and connection issues in production, I offered to onboard the same application to ESS. It took 2 days to get them up and running in the QA environment mainly due to firewalls, and 3 more days they were live and working in Production. This led to the cancellation of the other portal possibly saving the company millions in development work and also it led to my next set of projects.

- Onboarded 16 new applications to ESS making the total 20.
- Setup a standard for onboarding applications.
- Using JDBC and named hibernate queries I created user and usergroup reporting that would list out a users activity within ESS and the roles/privileges the users had. This could be done on individual users or an entire security usergroup. Used specific libraries to allow these reports to be downloaded in both Excel and PDF formats.
- Created IPV4 whitelisting and functionality to add it at a user level or a usergroup level. Then created a UI screen using ZK that displayed all failing logins that were rejected by the whitelist. On that screen I created a recursive function that would take a single IP that was failing for a user multiple times, add it once to the DB, and then remove all records of the failing login attempts from the proper table so it would no longer display in the UI. We ultimately deemed IP Whitelisting as an insecure means of security and went with stronger proxy settings using Siteminder and eventually PING. But the whitelisting still exists today for client comfort even though in my mind it doesn't provide much security.
- Created a Batch process using Quartz that took a file from the server sent by another application and created all usergroups and roles that were added within the other application.
- Created a universal message system that allowed us to broadcast messages for individual applications to all users upon login, this was helpful with maintenance activities.
- Configured Websphere apache setting to create a reverse proxy through siteminder for each of the applications being onboarded.
- Set up Mutual Auth through Datapower to connect to the Siteminder SOAP web services that were used to keep the Siteminder LDAP and ESS DB in sync.
- Created a signed cookie for extra application validation for user privileges.
- Switched from a cookie to a JWT token.
- Created a MultiFactor Authentication process using browser cookies to remember the user. This was set up at an application level so it did not impact all applications to begin with.
- Pitched switching the MFA solution to PING and onboarding all of the applications that made sense to ESS. This would potentially save the company millions of dollars in code rewriting effort as they would need to implement MFA across all 156 applications. This pitch was finally accepted in 2017 and the progress is still ongoing although I am not involved currently.

Brandon John Etchison
402-403-8155
BrandonJava95662@gmail.com

- Created a Round Robin Load Balancing solution for an application that needed Hot/Hot support when our Load Balancing team said it was currently possible for them. This happened within the apache configurations using a simple cookie with a single value to determine what server to send the request too.
- Implemented Fortify to the Maven build in order to determine security holes and findings within the application.