**PROFESSIONAL SUMMARY**

* 8 years of experience in developing backend with the help of Node.js, MongoDB, pg SQL, Express.js, Docker, Kubernetes, Terraform, and Elastic Search.
* Proficient in serverless architecture and integrating with various AWS services including AWS Lambda, SQS, SNS, S3, Step Functions, EC2, AWS EKS, and other AWS services.
* Created RESTful APIs using Express JS and Loopback frameworks, added authentication and authorization using jwt also used oauth2 and saml for SSO.
* Integrated with various external services like salesforce using jsforce and mailed using node mailer and MailChimp and integrated with different payment gateways like Instamojo.
* Proficient in multiple databases such as MongoDB, PostgreSQL, MSSQL, Redis, and Kafka Streams.
* Successfully loaded customer data from various sources into Salesforce
* Created CI/CD pipelines with Azure Pipelines and Jenkins.
* Experienced in deploying lambda functions and containerized microservices into multiple environments using Kubernetes solutions like AWS EKS, OpenShift Container Platform, and IBM Cloud Foundry.
* Proficient in running automated test cases including unit, integration, and end-to-end tests using Mocha, Chai, Sinon, and Cypress.
* Developed connectors for IBM App Connect integrations and created a website for photographers to publish event pictures. Worked on chatbots for customer questions and created a mock test website to help students in preparation of various exams.
* Experienced cloud engineer with a strong background in serverless technologies and AWS services. Gained expertise in AWS Lambda, Step Functions, API Gateway, S3, SQS, SNS, Node.js, and MuleSoft.
* Having a proven track record of delivering successful projects, including transforming MuleSoft DWL files to AWS Lambda using Node.js and implementing backend APIs using Express.js. experience with testing frameworks such as Mocha, Sinon, and Chai.
* Having experience in Node JS, Express JS, Serverless, MongoDB, PsSQL, MSSQL, MySQL, Elastic Search, Redis, Terraform, Kafka, Docker, Jenkins, Azur Pipelines,
* Having an experience in AWS services like S3, EC2, EKS, lambda, step functions, SQS, SNS. IBM cloud services like IBM CosS3 and Cloud Foundry
* Worked on Kubernetes, and OpenShift container platform.
* Worked on Testing Frameworks like Mocha, Chai, Sinon and Cypress.

**WORK HISTORY**

|  |  |
| --- | --- |
| **Client** | **Duration** |
| Mary Kay | Jun 2022 – Present |
| IBM | Oct 2021 – Jun 2022 |
| Snapwelt. | Sept 2015 – Sept 2021 |

**PROFESSIONAL EXPERIENCE**

**Client: Mary Kay Jun 2022 – Present**

**Role: Lead Software Consultant**

**Responsibilities:**

* Transformed MuleSoft DWL files to AWS Lambda using Node.js, resulting in cost-effective solutions for the company.
* Configured AWS S3 and API Gateway to initiate AWS Lambda functions on specified events.
* Utilized the AWS Step Functions service to seamlessly coordinate the execution of multiple functions, allowing us to easily implement branching logic and error handling in our workflow. This significantly improved the reliability and robustness of our application.
* Our team utilized AWS Simple Queue Service (SQS) and Simple Notification Service (SNS) to build a highly available and scalable messaging system for our application. We used SQS to decouple various components of our system, allowing them to operate independently and asynchronously.
* Worked on SNS to publish notifications and trigger lambdas in response to events occurring within our system.
* Utilized stored procedures in MS SQL to calculate consultant scores, transformed the data into the desired format, and stored it in S3 buckets for future reference.
* Our web application was built using Node.js and Express.js, and we used Microsoft SQL Server (MSSQL) as our database back-end. We found that the combination of Node.js, Express.js, and MSSQL provided a powerful and scalable platform for our application, enabling us to efficiently handle high levels of traffic and data volume.
* Created backend APIs using Express.js to support the needs of the front office teams.
* Utilized testing frameworks such as Mocha, Sinon, and Chai to ensure code quality.
* We leveraged Azure Pipelines to automate our build, test, and deployment processes, streamlining the delivery of code changes to our production environment. By using Azure Pipelines, we were able to deliver new features and updates consistently and reliably to our customers, improving their experience and satisfaction with our product.
* Experienced using JSforce, a JavaScript library that provides a simple and powerful way to interact with the Salesforce API. With JSforce, it is easy to perform common Salesforce tasks such as creating, reading, updating, and deleting records, as well as executing complex queries and operations. The ability to use JavaScript to access Salesforce data and functionality made it easier for me to build integrations and customizations on the Salesforce platform.
* To deploy and manage our microservices-based application, we used a combination of Kubernetes, Docker, AWS Elastic Kubernetes Service (EKS), and terraform. By utilizing these technologies, we were able to easily scale our application and maintain a high level of availability across multiple regions. The integration of Kubernetes and Docker allowed us to package and deploy our services in a consistent and reliable manner, while AWS EKS and Terraform provided us with the tools to efficiently manage and provision our infrastructure on the cloud.

**Client: IBM Oct 2021 – Jun 2022**

**Role: Lead Software Engineer**

**Responsibilities:**

* Worked on IBM app connect (IBM cloud >> Integration >> App connect)
* Developed connectors for IBM App Connect to rapidly connect applications to a variety of external systems and services.
* Using the intuitive visual flow designer and pre-built connectors, customers were able to quickly create integrations that would have otherwise required custom coding, saving them time and effort as worked to deliver new features and functionality to our customers.
* Working on Loopback connectors, we found them to be a powerful and flexible tool for integrating with a variety of external data sources. Used Loopback connectors to connect to databases, APIs, and other services, allowing me to easily retrieve and manipulate data within my application. The declarative configuration style of Loopback connectors made it easy for me to set up new integrations and modify existing ones as needed.
* Used loopback framework to develop connectors to integrate salesforce, oracle EBS and IBM COSS3.
* To ensure the quality and reliability of our application, we used a combination of Mocha, Chai, and Sinon for our testing needs. Mocha provided the framework for organizing and running our tests, Chai enabled us to make assertions about the behavior of our code, and Sinon allowed us to stub and mock external dependencies. Together, these tools gave us the confidence that our application was working correctly and ready for deployment.
* Deployed containerized applications in cloud foundry, Kubernetes clusters (k8s).
* We implemented Apache Kafka as the messaging backbone for our distribution system. Kafka's robust, scalable, and high-throughput design allowed us to handle a large volume of events and enabled us to easily add new consumers and producers to our system as needed. The flexibility and reliability of Kafka were instrumental in the success of our application.
* Used Kafka message queue to publish and subscribe events (on booting up of connectors).
* Redis as a key-value store for caching frequently accessed data, as a message broker for communication between microservices, and as a data structure store for storing and manipulating complex data types. The versatility and speed of Redis made it an essential component of our tech stack. sed Redis for storing customer credentials (API key, access key Id and other relevant info).
* Mongoose to define the schema of my documents and perform CRUD operations on my data. Elasticsearch, on the other hand, is a powerful search and analytics engine that used to index and search large volumes of data quickly and efficiently. By combining the two technologies we were able to build powerful and efficient applications that were able to store and retrieve data at scale.

**Client: Snapwelt. Sept 2015 – Sept 2016**

**Role: Senior Software Engineer**

**Responsibilities:**

* Worked on Node, pgSQL, and Express to create backend APIs. Also worked on testing part using Mocha & Chai. And wrote a few automation test cases using Cypress.
* Node.js applications implementation on various authentication and authorization mechanisms to secure access to my application's resources. JSON Web Tokens (JWTs) and OAuth2 to enable user authentication and authorization and implemented role-based access control to restrict access to certain resources based on a user's permissions. By properly implementing authentication and authorization we can ensure that only authorized users were able to access my application's resources and perform certain actions.
* Used node mailer for sending emails via mail gun for authentication-related emails like sending verification tokens or reset-password emails.
* Experienced in using the AWS Recognition service for facial recognition and face matching. We used AWS Recognition to build a system that could automatically detect and match individuals in images.
* Busboy, a streaming parser for HTML form data, and AWS S3's Multipart Upload capability to manage the upload of large images to our server. The combination of Busboy and Multipart Upload allowed us to efficiently manage the streaming and storage of images, even when they were large. This enabled us to provide a smooth and reliable experience for photographers, who were able to upload their images quickly and without any issues.
* Experienced building web applications using Node.js and the Express.js framework, and PostgreSQL as a database back-end. The combination of Node.js, Express.js, and PostgreSQL provides a powerful and scalable platform for building web applications. Used this stack to build a variety of applications, including REST APIs, real-time web apps, and server-rendered web pages. The versatility and performance of this stack have made it a go-to choose for many of my projects.
* Implemented and deployed multiple microservices during this period.
* Worked on MongoDB for storing event info, participants, and photographers' info. and used Redis for caching purposes. Used MongoDB sharding to manage high traffic.
* Responsible for managing the team, writing automation test cases, also deploying the code for production into EC2 machines.

**EDUCATIONAL BACKGROUND**

* Student of Bachelor of Engineering (Electrical Engineering), IIT Patna.
* Bachelor of Commerce, Rabindranath Tagore University