DevOps Intern Assignment:

**Objective:** Construct and implement a scalable Node.js server utilizing Nginx and PM2 for process management, and MongoDB for data persistence. Leverage various AWS offerings for deployment, orchestration, and operational supervision.

**Tasks:**

1. **AWS Configuration:**

- Initialize an Amazon EC2 instance to serve the Node.js application, managed with Nginx and PM2 for enhanced performance and reliability.

- Employ AWS Elastic Beanstalk to facilitate the application's deployment and autoscaling capabilities.

- Employ AWS S3 for the custody of any static resources.

2. **CI/CD Pipeline Engineering:**

- Integrate AWS CodePipeline to streamline the continuous integration and deployment processes.

- Establish a source control mechanism using a platform such as GitHub.

- Configure automated build and deployment phases for a seamless development lifecycle.

3. **Monitoring and Observability:**

- Implement AWS CloudWatch to meticulously monitor application performance, track API calls, and oversee system-wide operations.

- Set alarms and create notifications for any irregularities or performance issues detected.

4. **Security Provisioning:**

- Enforce stringent AWS security protocols by configuring IAM roles and security groups to safeguard the application infrastructure.

5. **Docker Integration:**

- Containerize the Node.js application using Docker to ensure consistency across development, testing, and production environments.

6. **Advanced Monitoring:**

- Configure detailed logging for each API call to enable in-depth analysis and proactive issue resolution.

- Establish a comprehensive alarm and event notification system within AWS to maintain operational integrity.

7. **Bonus Objectives:**

- Implement an Elastic Load Balancer within AWS to distribute traffic and maximize availability.

- Set up Auto Scaling to adjust the compute resources automatically, aligning with the demand patterns.

8. **Deliverables:**

- A fully configured CI/CD pipeline in AWS CodePipeline.

- Detailed documentation encapsulating all architectural decisions, setup instructions, and operational guidelines.

9. **Assessment Benchmarks:**

- Proficiency in applying AWS services for deployment and runtime operations.

- The robustness of the CI/CD pipeline and Docker integration.

- Code quality, performance monitoring, and comprehensive documentation.