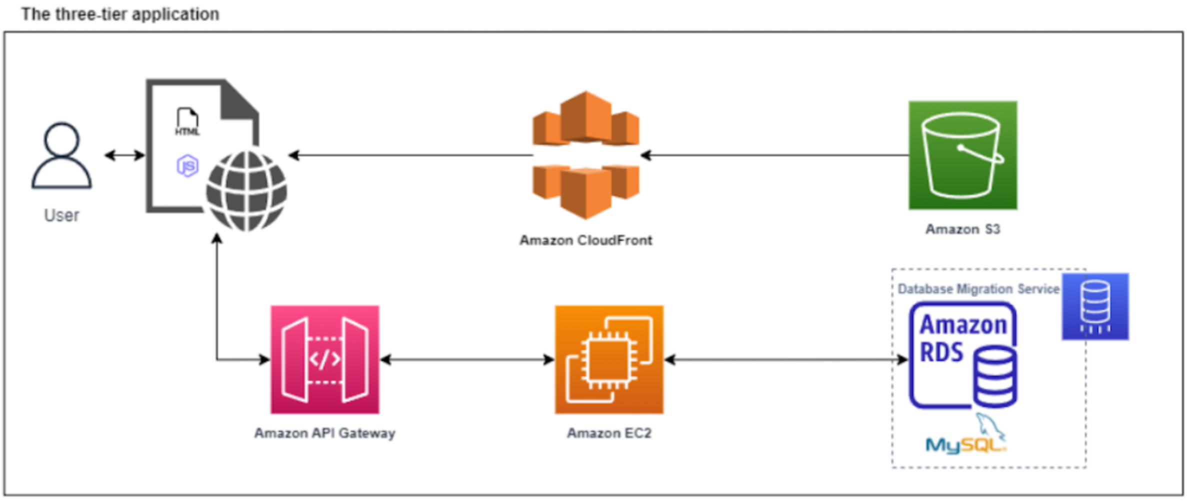


**Explanation:**

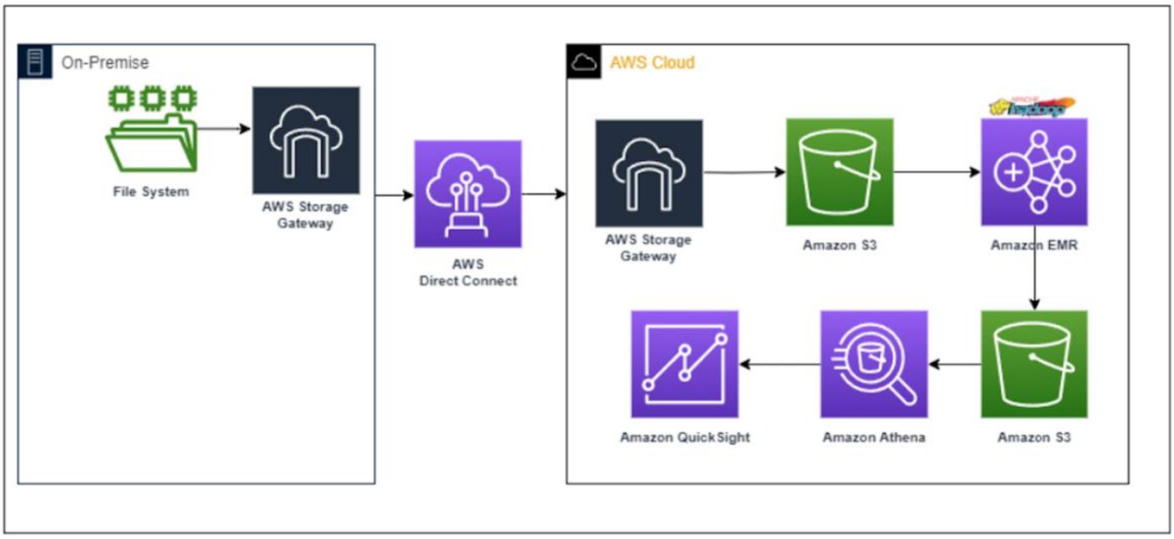
The solution operates by leveraging various AWS cloud-native technologies. To host static webpage content such as HTML, CSS, and JavaScript for the three-tier application, I opted for Amazon S3. Amazon S3 eliminates the need for a separate web server, provides storage management features to control costs, reduces latency, and meets regulatory requirements. End users access the front-end content through Amazon CloudFront, which enhances security with traffic encryption and offers cheaper data transfer rates than Amazon S3. Instead of accessing S3 directly, users interact with CloudFront, which delivers the web contents to their browsers. API calls from the browser reach the backend through the Amazon API Gateway.

With Amazon S3 serving as the web server, Apache Web Server is no longer required. The API Gateway handles API requests, facilitates data transmission to and from the internet, and manages authentication. It can efficiently scale and provides valuable performance metrics for API calls. Requests coming from the internet are routed to the Java application running on Amazon EC2 instances.

The existing backend Java application code operates within Amazon EC2 instances, granting us complete control over computing resources. This allows us to run Java applications using any Java runtime environment on the EC2 server instances. EC2's quick provisioning and scaling capabilities enable us to respond promptly to changing computing requirements based on traffic. By utilizing Amazon EC2 Auto Scaling, we can easily adjust the number of instances based on usage. Database calls from the Java application are directed to MySQL, which is hosted on Amazon RDS.

To migrate the existing MySQL database to AWS, we employ the AWS Database Migration Service. Amazon RDS is then used to manage and operate the MySQL database. Choosing RDS as a managed service simplifies database management, freeing us to focus on the application rather than handling complex and time-consuming database maintenance.

The data analytics workload



**A justification for the specific approach they took for the migration?**

The data analytics solution I propose involves a hybrid approach, with the majority of the workload running on AWS. To achieve this, I suggest transferring on-premises data to AWS using AWS Direct Connect. This service offers the shortest path to AWS resources, ensuring a secure and consistent network experience without touching the public internet. It also reduces networking costs with low data transfer rates out of AWS, which is ideal for handling large datasets.

To facilitate the data transfer, I recommend utilizing AWS Storage Gateway. This option allows low-latency data access to on-premises applications while benefiting from AWS's agility and security capabilities in the cloud. It also provides virtually unlimited cloud storage and supports compliance efforts with features like encryption and audit logging.

Once the data is in AWS, it will be stored in Amazon S3, which I selected for its continuous availability, storage management features for cost reduction (pay-as-you-go model), low latency, and ability to meet regulatory requirements. Amazon S3 enables running big data analytics, artificial intelligence (AI), machine learning (ML), archiving data at a minimal cost, and performing critical data backups and restores.

For processing the data, an Amazon EMR cluster running Apache Hadoop is the next step. Amazon EMR streamlines the installation and configuration of Hadoop components, including Hadoop MapReduce, YARN, and HDFS. It simplifies the creation and operation of big data environments and easily scales resources to meet varying business needs.
Once processed by Hadoop, the data can be written back to an S3 bucket for further usage. This separation ensures that processed and ingested data remains available closer to the next analytic step.

For the subsequent step, I recommend using Amazon Athena. This service offers instant data querying with results delivered in seconds, and you only pay for the queries you run. With its

support for standard SQL, Amazon Athena provides an interactive query experience for analyzing data stored in Amazon S3. Additionally, Athena being serverless eliminates the need for infrastructure management, making it an efficient choice.
I recommend utilizing Amazon QuickSight as the data visualization tool for this project. The primary reason for choosing QuickSight is its seamless integration with Amazon Athena. Furthermore, QuickSight operates without the need for server management and follows a pay-per-use pricing model. With native AWS service integrations and built-in security, it aligns well with other AWS-managed services and scales efficiently. These attributes contribute to smoother data insights and visualization while reducing costs, as there is no need to purchase additional licenses for other tools.

Both of these solutions can be further enhanced by implementing a similar setup in a different availability zone, such as Active-Active or Active-Passive. This approach improves system resilience, ensuring a more robust and reliable infrastructure.