Explanation:

For this scenario, I have designed a three-tier architecture that utilizes the following AWS services:

1. **Amazon Virtual Private Cloud (Amazon VPC)**: A VPC is a virtual network that allows you to launch Amazon Web Services (AWS) resources into a virtual network. It provides you with complete control over your virtual networking environment, including the ability to create subnets, configure route tables, and network gateways. In this architecture, I have used a VPC to create a virtual network for hosting the different tiers of the application.
2. **Elastic Load Balancing (ELB)**: ELB is a highly available and scalable service that automatically distributes incoming traffic across multiple targets, such as Amazon EC2 instances. It enables you to improve the availability and fault tolerance of your applications by automatically detecting and routing traffic to healthy instances. In this architecture, I have used an ELB to distribute traffic across multiple EC2 instances running in the compute layer.
3. **Amazon Elastic Compute Cloud (Amazon EC2)**: EC2 is a web service that provides resizable compute capacity in the cloud. It allows you to quickly and easily launch instances of virtual servers, called EC2 instances, and enables you to scale up or down your computing capacity depending on your application needs. In this architecture, I have used EC2 instances to host the application logic and serve requests from the ELB.
4. **Amazon Relational Database Service (Amazon RDS)**: RDS is a web service that makes it easy to set up, operate, and scale a relational database in the cloud. It provides cost-efficient and resizable capacity while automating time-consuming administration tasks, such as hardware provisioning, database setup, patching, and backups. In this architecture, I have used RDS to host the backend database layer.

Traffic flows through the different AWS components in the following way:

1. A client sends a request to the ELB.
2. The ELB distributes the request to one of the available EC2 instances.
3. The EC2 instance runs the application logic and retrieves the necessary data from the backend RDS database.
4. The EC2 instance sends the response back to the ELB, which then forwards it to the client.

By utilizing the AWS services mentioned above, we can ensure that the application is highly available, scalable, and fault-tolerant. The ELB ensures that traffic is distributed evenly across multiple EC2 instances, providing high availability and fault tolerance. EC2 instances can be easily scaled up or down based on application needs, ensuring that the compute layer is always appropriately sized. The RDS database layer is highly available and automatically backed up, ensuring data availability and durability. Finally, by using a VPC, we can ensure that the entire architecture is securely isolated from other networks and accessible only to authorized users.