06-06-2023

**DIFFERENT APPLICATION SERVERS AVAILABLE IN THE MARKET SPACE**

1. **Apache Tomcat:**

Apache Tomcat is an open-source Java-based application server that provides a lightweight environment for running Java web applications. It supports the Java Servlet and Java Server Pages (JSP) technologies and is widely used for hosting Java-based web applications.

**Overview About Apache Tomcat:**

1. Java Servlet and JSP Support: Apache Tomcat implements the Java Servlet and Java Server Pages (JSP) specifications, which are part of the Java Enterprise Edition (Java EE) platform. It serves as a container for executing servlets and JSP pages, enabling the development and deployment of dynamic web applications.
2. Web Server Capabilities: Tomcat includes a built-in HTTP server that can handle HTTP requests and responses. It can function both as a standalone web server or in conjunction with other web servers such as Apache HTTP Server or Nginx. This allows it to serve static content and act as a gateway for forwarding requests to Java web applications.
3. Servlet Container: Tomcat serves as a servlet container, providing an environment for executing Java servlets. Servlets are Java classes that handle incoming HTTP requests, process them, and generate dynamic responses. Tomcat manages the lifecycle of servlets, handling request dispatching, session management, and other servlet-related functionalities.
4. JSP Engine: Tomcat includes a JSP engine that translates JSP pages into servlets at runtime. JSP allows developers to embed Java code within HTML templates, providing a convenient way to create dynamic web pages. Tomcat compiles JSP pages into Java servlets, which are then executed to generate dynamic content.
5. Catalina: Catalina is the core component of Tomcat responsible for handling HTTP requests, managing servlets, and serving web applications. It provides the infrastructure for processing servlets, managing request/response cycles, and handling HTTP sessions.
6. Connectors: Tomcat supports various connectors that enable communication between the server and external clients. The most commonly used connector is the HTTP connector, which allows Tomcat to accept HTTP requests. Additionally, Tomcat supports connectors for other protocols, such as HTTPS, AJP (Apache JServ Protocol), and more.
7. Clustering and Load Balancing: Tomcat supports clustering and load balancing, allowing multiple instances of Tomcat to work together to distribute the load and provide high availability. Clustering enables scalability and fault tolerance by replicating sessions and data across multiple Tomcat instances.
8. Security: Tomcat provides several security mechanisms to protect web applications, including authentication, authorization, and secure communication. It supports various authentication methods, such as basic authentication, form-based authentication, and container-managed authentication. Additionally, Tomcat integrates with SSL/TLS protocols to enable secure communication over HTTPS.
9. Management and Monitoring: Tomcat provides management and monitoring capabilities through the Tomcat Manager web application. The Manager allows administrators to deploy, undeployed, start, and stop web applications remotely. It also provides monitoring statistics, including request counts, session information, and memory usage.
10. Embeddable and Extensible: Tomcat is designed to be embeddable, allowing developers to integrate it into their applications. It provides APIs and interfaces that enable programmatic configuration and control of Tomcat instances. Tomcat is also extensible, with support for custom components, valves, and realms, allowing developers to enhance its functionality.
11. Community and Ecosystem: Apache Tomcat has a large and active community of developers and users. The Apache Software Foundation provides extensive documentation, guides, and resources for using and configuring Tomcat. The community offers support through mailing lists, forums, and a bug tracking system.

Apache Tomcat is widely used for deploying Java-based web applications and is particularly popular in the Java development community. Its lightweight nature, simplicity, and robust feature set make it a preferred choice for developers seeking a Java.

1. **JBoss/ WildFly:**

JBoss, now known as WildFly, is an open-source Java-based application server developed by Red Hat. It supports the Java EE (Enterprise Edition) platform and provides a robust and scalable environment for running enterprise-level Java applications.

**Overview About JBoss:**

1. Java EE Support: JBoss/WildFly is fully compliant with the Java Enterprise Edition (Java EE) platform specifications. It supports a wide range of Java EE technologies and APIs, including Servlets, JavaServer Pages (JSP), JavaServer Faces (JSF), Enterprise JavaBeans (EJB), Java Message Service (JMS), Java Persistence API (JPA), and many more. This allows developers to build and deploy complex enterprise applications leveraging the Java EE ecosystem.
2. Modular Architecture: JBoss/WildFly follows a modular architecture, where the server is divided into smaller, independently deployable components known as modules. Each module encapsulates a specific functionality or Java EE component. This modular approach provides flexibility, improves performance, and allows for easy customization and maintenance of the server.
3. Application Deployment: JBoss/WildFly supports various deployment options for Java applications. It allows you to deploy applications in exploded form (unpacked directory structure) or as packaged archive files, such as WAR (Web Application Archive) or EAR (Enterprise Archive). It also supports hot deployment, allowing applications to be redeployed without restarting the server, which can enhance development productivity.
4. Clustering and High Availability: JBoss/WildFly provides clustering capabilities to support high availability and scalability. It allows you to create clusters of multiple server instances that work together to distribute the load and provide failover in case of server failures. Clustering enables horizontal scalability, fault tolerance, and load balancing for Java applications.
5. Management and Administration: JBoss/WildFly includes a management console, known as the WildFly Administration Console, which provides a web-based interface for managing and configuring the server. The administration console allows you to monitor server resources, deploy and undeployed applications, configure data sources, manage security settings, and perform various administrative tasks.
6. Security: JBoss/WildFly offers robust security features to protect Java applications. It supports various authentication mechanisms, including LDAP, Kerberos, and certificate-based authentication. It also provides authorization mechanisms to control access to resources, role-based access control, and fine-grained security policies. SSL/TLS encryption can be configured to secure communication between clients and the server.
7. Messaging and Integration: JBoss/WildFly includes a high-performance messaging system based on the Java Message Service (JMS) standard. It supports both point-to-point and publish-subscribe messaging models, allowing applications to exchange messages asynchronously. Additionally, JBoss/WildFly provides integration capabilities through technologies like Apache Camel and JCA (Java Connector Architecture), enabling seamless integration with external systems and services.
8. Database Connectivity: JBoss/WildFly supports various data source configurations for connecting to databases. It provides connection pooling, transaction management, and connection recovery features, improving the performance and reliability of database interactions in Java applications. It also supports advanced database features, such as connection multiplexing and XA (eXtended Architecture) transactions.
9. Extensibility: JBoss/WildFly is highly extensible, allowing you to add custom functionality and integrate with third-party libraries and frameworks. It provides extension points, APIs, and a robust module system that enables the development of custom modules, services, and components to extend the server's capabilities.
10. Community and Support: JBoss/WildFly benefits from an active and vibrant community of developers and users. The Red Hat community provides extensive documentation, tutorials, forums, and support resources. The community-driven development model ensures frequent updates, bug fixes, and enhancements to the server.

JBoss/WildFly is a feature-rich and robust application server that enables the deployment of Java-based enterprise applications. Its extensive Java EE/Jakarta EE support, modular architecture, scalability, and integration capabilities make it a popular choice for building and managing complex enterprise systems.

1. **Oracle WebLogic Server:**

Oracle WebLogic Server is a leading Java EE application server that offers a feature-rich platform for deploying and managing enterprise-level Java applications. It provides high availability, scalability, and performance optimizations for mission-critical applications.

**Overview About Oracle WebLogic Server:**

1. Java EE Support: Oracle WebLogic Server is compliant with Java EE specifications, allowing developers to leverage the Java EE ecosystem and build enterprise applications using technologies such as Servlets, JavaServer Pages (JSP), JavaServer Faces (JSF), Enterprise JavaBeans (EJB), Java Message Service (JMS), Java Persistence API (JPA), and more. It supports the latest versions of Java EE, ensuring compatibility with industry standards.
2. Scalability and High Availability: WebLogic Server offers robust scalability and high availability features to meet the demands of enterprise-level applications. It supports clustering, which allows multiple server instances to work together as a cluster to distribute the workload and provide fault tolerance. It also offers session replication, load balancing, and failover capabilities, ensuring uninterrupted operation even during hardware or software failures.
3. Management and Administration: WebLogic Server provides a comprehensive administration console for managing and configuring the server. The console offers a user-friendly web-based interface that allows administrators to monitor server health, configure resources, deploy applications, manage security, and perform various administrative tasks. It also supports scripting and automation through command-line tools and APIs.
4. Security: WebLogic Server offers robust security features to protect applications and resources. It supports secure communication over SSL/TLS protocols, ensuring encrypted data transfer between clients and the server. It provides authentication mechanisms, including username/password, certificate-based authentication, and integration with external identity providers. Role-based access control (RBAC) and fine-grained security policies enable administrators to control access to resources based on user roles and permissions.
5. High-Performance Architecture: WebLogic Server is designed for high performance and scalability. It utilizes advanced features such as connection pooling, thread pooling, and caching mechanisms to optimize resource utilization and improve response times. It also includes a highly efficient Java Virtual Machine (JVM) tuned for optimal performance on the Oracle platform.
6. Integration Capabilities: WebLogic Server offers extensive integration capabilities to connect with various enterprise systems and services. It supports Java Connector Architecture (JCA) for integrating with back-end systems such as databases, enterprise messaging systems, and legacy applications. It also provides support for popular integration technologies like Oracle Service Bus and Oracle SOA Suite for building and orchestrating service-oriented architectures (SOA).
7. Web Services Support: WebLogic Server provides comprehensive support for developing and deploying web services. It supports Java API for XML Web Services (JAX-WS) and Java API for RESTful Web Services (JAX-RS) standards. It also offers built-in support for WS-Security, WS-Reliability, and other web service standards, enabling secure and reliable communication between applications.
8. Enterprise-grade Messaging: WebLogic Server includes a robust messaging infrastructure based on the Java Message Service (JMS) standard. It provides support for both point-to-point and publish-subscribe messaging models, allowing applications to exchange messages asynchronously. It offers features such as durable subscriptions, message persistence, transactional messaging, and integration with enterprise messaging systems like Oracle Advanced Queuing.
9. Monitoring and Diagnostics: WebLogic Server provides extensive monitoring and diagnostics capabilities to ensure the health and performance of applications. It offers tools and APIs for monitoring server resources, tracking application performance metrics, and diagnosing issues. It supports integration with third-party monitoring tools and frameworks for centralized monitoring and management.

Oracle WebLogic Server is a mature and robust application server trusted by enterprises worldwide for running mission-critical Java applications. Its extensive feature set, scalability, integration capabilities, and compatibility make it a preferred choice for organizations seeking a reliable and scalable platform for their enterprise applications.

1. **IBM WebSphere Application Server:**

IBM WebSphere Application Server is a Java EE-based application server that provides a secure and scalable platform for running enterprise applications. It offers advanced features like clustering, load balancing, and dynamic caching for high-performance application deployment.

**Overview About IBM WebSphere:**

1. Java EE Support: IBM WebSphere supports the Java EE (Enterprise Edition) platform, allowing developers to leverage a wide range of Java EE technologies and APIs to build enterprise applications. It supports various Java EE specifications, including Servlets, JavaServer Pages (JSP), JavaServer Faces (JSF), Enterprise JavaBeans (EJB), Java Message Service (JMS), Java Persistence API (JPA), and more. It ensures compatibility with Java EE standards and offers support for the latest versions.
2. Scalability and High Availability: WebSphere Application Server is designed for scalability and high availability. It supports clustering, allowing multiple server instances to work together as a cluster to distribute the workload and provide fault tolerance. It offers advanced load balancing and failover capabilities to ensure uninterrupted operation even during hardware or software failures. It also supports horizontal scaling by adding or removing server instances dynamically based on application demand.
3. Enterprise Integration: WebSphere provides extensive integration capabilities to connect with enterprise systems and services. It offers support for various integration technologies, including Java Connector Architecture (JCA), Service Component Architecture (SCA), and Service Integration Bus (SIB). These features enable seamless integration with backend systems, databases, messaging systems, and legacy applications.
4. Management and Administration: WebSphere Application Server provides a comprehensive administration console and management tools for configuring and managing the server. The administration console offers a web-based interface for managing server resources, configuring applications, monitoring performance, and performing administrative tasks. It also provides scripting and automation capabilities through command-line tools and APIs.
5. Security: WebSphere includes robust security features to protect applications and resources. It supports secure communication over SSL/TLS protocols and offers features like authentication, authorization, and secure storage of sensitive information. It integrates with external user registries, LDAP directories, and identity management systems for centralized user management. It also supports fine-grained security policies and role-based access control (RBAC) to control access to resources.
6. Performance Optimization: WebSphere Application Server is optimized for performance. It employs various performance optimization techniques, including connection pooling, thread pooling, and caching mechanisms. It also provides features like dynamic caching, dynamic workload management, and intelligent routing to optimize resource utilization and improve application performance.
7. Application Deployment: WebSphere supports various deployment options to accommodate different application architectures and deployment scenarios. It allows applications to be deployed as packaged archives such as WAR (Web Application Archive), EAR (Enterprise Archive), and EJB (Enterprise JavaBeans) modules. It supports hot deployment, enabling applications to be updated and redeployed without restarting the server.
8. Cloud-Native Capabilities: WebSphere Application Server offers cloud-native capabilities, allowing applications to be deployed and managed in cloud environments. It supports containerization technologies like Docker and Kubernetes, enabling applications to be packaged as containers and deployed on cloud platforms. It integrates with IBM Cloud and other cloud providers, providing seamless integration and management of applications in a cloud-native environment.
9. Microservices Architecture: WebSphere provides features and tools to support the development and deployment of microservices-based architectures. It supports the MicroProfile specification, which provides additional APIs and frameworks for building microservices. It offers features like service discovery, circuit breaker, fault tolerance, and dynamic scaling, enabling the development and management of scalable and resilient microservices.
10. DevOps Integration: WebSphere seamlessly integrates with DevOps practices and toolchains. It supports continuous integration and continuous deployment (CI/CD) workflows, allowing developers to automate build, test, and deployment processes. It integrates with popular DevOps tools like Jenkins, Git, and Ansible, enabling streamlined application development and delivery.
11. Support and Ecosystem: IBM WebSphere benefits from IBM's extensive support and ecosystem. It provides comprehensive documentation, knowledge bases, and forums for developers and administrators. IBM offers commercial support options with various service-level agreements (SLAs) for organizations requiring enterprise-grade support.
12. Compatibility and Standards: WebSphere Application Server adheres to industry standards and ensures compatibility with other Java EE-compliant servers and frameworks. It supports open standards such as XML, SOAP, WSDL, JPA, JAX-WS, and more, enabling interoperability with diverse systems and technologies.

IBM WebSphere Application Server is a robust and comprehensive application server trusted by enterprises worldwide for deploying and managing Java applications. Its extensive features, scalability, integration capabilities, and compatibility make it a preferred choice for organizations seeking a reliable platform for their enterprise applications.

1. **GlassFish:**

GlassFish is an open-source Java EE application server developed by Eclipse Foundation. It offers a scalable and extensible platform for deploying Java applications and provides support for the latest Java EE specifications.

**Overview About GlassFish:**

1. **Jetty:**

Jetty is a lightweight and embeddable Java-based application server that supports the Servlet and WebSocket APIs. It is often used for embedded deployments and development environments where low overhead and fast startup times are essential.

**Overview About Jetty:**

1. Java EE Support: GlassFish fully supports the Java EE platform, allowing developers to leverage a wide range of Java EE technologies and APIs for building enterprise applications. It supports various Java EE specifications, including Servlets, JavaServer Pages (JSP), JavaServer Faces (JSF), Enterprise JavaBeans (EJB), Java Message Service (JMS), Java Persistence API (JPA), and more. It ensures compatibility with Java EE standards and offers support for the latest versions.
2. Open-Source: GlassFish is an open-source project, which means the source code is freely available and can be modified and distributed under open-source licenses. It fosters community collaboration and enables developers to contribute to the project's development and improvement.
3. Lightweight and Fast: GlassFish is known for its lightweight nature and fast startup times. It provides a lightweight runtime environment that minimizes resource usage and allows for quick application deployment and startup. This makes it suitable for development, testing, and deployment in various environments, including local development machines and production servers.
4. Developer-Friendly: GlassFish offers a developer-friendly environment with features that enhance productivity. It includes a robust administration console for managing and configuring the server, as well as tools and plugins for popular integrated development environments (IDEs) like Eclipse and NetBeans. It supports hot deployment, allowing developers to make code changes and see the results immediately without restarting the server.
5. Modularity and Extensibility: GlassFish follows a modular architecture, allowing developers to choose and configure only the required components for their applications. It offers a wide range of extensions and plugins, enabling developers to enhance the server's functionality and integrate with third-party libraries and frameworks. It also supports the OSGi (Open Services Gateway Initiative) framework for modular application development.
6. Clustering and High Availability: GlassFish supports clustering for high availability and scalability. It allows multiple server instances to work together as a cluster, distributing the workload and providing fault tolerance. It offers features such as session replication, load balancing, and failover capabilities, ensuring uninterrupted operation even during hardware or software failures.
7. Security: GlassFish includes robust security features to protect applications and resources. It supports secure communication over SSL/TLS protocols and offers features like authentication, authorization, and secure storage of sensitive information. It integrates with external user registries, LDAP directories, and identity management systems for centralized user management. It also supports fine-grained security policies and role-based access control (RBAC) to control access to resources.
8. Monitoring and Diagnostics: GlassFish provides monitoring and diagnostics tools to monitor application performance, track resource usage, and diagnose issues. It offers a built-in monitoring infrastructure that collects and displays real-time performance metrics, allowing administrators to identify and resolve bottlenecks. It also supports integration with third-party monitoring tools for centralized monitoring and management.
9. Integration Capabilities: GlassFish offers extensive integration capabilities to connect with enterprise systems and services. It supports Java Connector Architecture (JCA) for integrating with backend systems such as databases, enterprise messaging systems, and legacy applications. It also provides support for popular integration technologies like JAX-WS and JAX-RS for building and consuming web services.
10. Community and Support: GlassFish benefits from a vibrant and active community of developers and users. The open-source nature of GlassFish encourages community participation, collaboration, and the sharing of knowledge and resources. The community provides forums, mailing lists, and online resources for support, troubleshooting, and knowledge sharing.
11. Java EE Compatibility: GlassFish ensures compatibility with the Java EE standards and specifications. It undergoes rigorous compatibility testing to comply with the Java EE platform's specifications, ensuring that applications developed on GlassFish can be deployed on other Java EE-compliant application servers without modification.
12. Integration with NetBeans: GlassFish has a strong integration with the NetBeans IDE, an open-source integrated development environment for Java development. NetBeans provides seamless integration with GlassFish, allowing developers to easily deploy, debug, and manage applications on the server directly from the IDE. It offers features like code generation, auto-completion, and integrated testing for efficient application development.
13. Cloud-Native Capabilities: GlassFish supports cloud-native application development and deployment. It offers features and tools for containerization and cloud integration. It supports containerization technologies like Docker and Kubernetes, allowing applications to be packaged and deployed as containers. It also integrates with cloud platforms like Oracle Cloud, AWS, and Azure, providing seamless deployment and management of applications in cloud environments.
14. Upgrade and Patching: GlassFish releases regular updates, patches, and new versions to address security vulnerabilities, bug fixes, and feature enhancements. It provides tools and mechanisms to upgrade the server and apply patches easily. The community and support resources provide guidance and assistance in upgrading and keeping the server up-to-date.

GlassFish is a flexible, feature-rich, and community-driven Java EE application server. Its open-source nature, Java EE compatibility, and strong integration with development tools make it an attractive choice for developers and organizations looking for an enterprise-grade application server for Java applications.

1. **Apache Geronimo:**

Apache Geronimo is an open-source Java EE application server developed by the Apache Software Foundation. It provides a full-featured Java EE environment for hosting enterprise applications and supports multiple Java EE versions.

**Overview About Apache Geronimo:**

1. Java EE Support: Apache Geronimo supports the Java EE platform, allowing developers to build enterprise applications using a wide range of Java EE technologies and APIs. It supports specifications such as Servlets, JavaServer Pages (JSP), JavaServer Faces (JSF), Enterprise JavaBeans (EJB), Java Message Service (JMS), Java Persistence API (JPA), and more. It ensures compatibility with Java EE standards and provides support for different versions.
2. Modular Architecture: Geronimo follows a modular architecture that allows developers to select and configure only the required components for their applications. It provides a flexible and lightweight runtime environment, enabling efficient resource utilization and better performance. The modular approach allows for easier application deployment and the ability to customize the server according to specific requirements.
3. Web Services Support: Geronimo offers comprehensive support for web services and provides a runtime environment for deploying and consuming them. It supports standards such as Java API for XML Web Services (JAX-WS) and Java API for RESTful Web Services (JAX-RS). It allows developers to expose web services and invoke them using various protocols and data formats.
4. Developer-Friendly: Geronimo provides tools and features that enhance developer productivity. It offers integration with popular integrated development environments (IDEs) like Eclipse and NetBeans, enabling developers to streamline their development workflow. It supports hot deployment, allowing applications to be updated and redeployed without restarting the server, thereby reducing development time.
5. Administration and Management: Geronimo provides a web-based administration console for managing and configuring the server. The administration console offers a user-friendly interface for managing server resources, configuring applications, monitoring performance, and performing administrative tasks. It also provides command-line tools and APIs for scripting and automation of administrative tasks.
6. Clustering and High Availability: Geronimo supports clustering, allowing multiple server instances to work together as a cluster. Clustering provides scalability, load balancing, and high availability by distributing the workload and providing fault tolerance. Geronimo includes features like session replication, load balancing, and failover capabilities to ensure uninterrupted operation even during hardware or software failures.
7. Security: Geronimo includes robust security features to protect applications and resources. It supports secure communication over SSL/TLS protocols and provides authentication and authorization mechanisms. It integrates with external user registries and supports role-based access control (RBAC) to control access to resources. Geronimo also offers features for securing web services and implementing fine-grained security policies.
8. Integration Capabilities: Geronimo supports integration with various enterprise systems and technologies. It provides connectors for integrating with backend systems such as databases, enterprise messaging systems, and legacy applications. It supports Java Connector Architecture (JCA) and provides features like transaction management and connection pooling for efficient integration.
9. Community and Support: Apache Geronimo benefits from a vibrant community of developers and users. The community actively contributes to the development, improvement, and support of Geronimo. It provides forums, mailing lists, and online resources for support, knowledge sharing, and troubleshooting.
10. Open-Source: Apache Geronimo is an open-source project, which means the source code is freely available and can be modified and distributed under open-source licenses. The open-source nature encourages community participation, collaboration, and innovation.

Apache Geronimo is a reliable and feature-rich Java EE application server that offers a flexible and scalable platform for deploying enterprise applications. Its modular architecture, Java EE compatibility, and developer-friendly features make it a suitable choice for organizations and developers.

1. **Payara Server:**

Payara Server is an open-source Java EE application server derived from GlassFish. It offers additional features, performance optimizations, and improved management capabilities for running Java applications in production environments.

**Overview About Payara Server:**

1. Java EE Support: Payara Server fully supports the Java EE platform, allowing developers to leverage a wide range of Java EE technologies and APIs for building enterprise applications. It supports specifications such as Servlets, JavaServer Pages (JSP), JavaServer Faces (JSF), Enterprise JavaBeans (EJB), Java Message Service (JMS), Java Persistence API (JPA), and more. It ensures compatibility with Java EE standards and provides support for the latest versions.
2. Derived from GlassFish: Payara Server is derived from the GlassFish project, which means it inherits many of the features and qualities of GlassFish. GlassFish is known for its robustness, stability, and compatibility with Java EE standards. Payara Server builds upon these foundations, adding additional features and enhancements.
3. High Performance: Payara Server is designed to deliver high performance and scalability for enterprise applications. It includes features like connection pooling, thread pooling, caching, and clustering, which optimize resource usage and improve application response times. It leverages efficient algorithms and techniques to handle high traffic loads and provide excellent performance.
4. Developer-Friendly: Payara Server offers a developer-friendly environment with features that enhance productivity. It provides seamless integration with popular integrated development environments (IDEs) like Eclipse and IntelliJ IDEA, enabling developers to streamline their development workflow. It supports hot deployment, allowing applications to be updated and redeployed without restarting the server, which speeds up the development process.
5. Administration and Monitoring: Payara Server provides a comprehensive administration console and monitoring tools for managing and monitoring the server. The administration console offers a web-based interface for configuring server resources, deploying applications, and managing security. It includes features for monitoring server performance, tracking resource usage, and diagnosing issues. It also supports scripting and automation through command-line tools and APIs.
6. Clustering and High Availability: Payara Server supports clustering to achieve high availability and scalability. It allows multiple server instances to work together as a cluster, distributing the workload and providing fault tolerance. It includes features like session replication, load balancing, and failover capabilities to ensure continuous operation even in the event of hardware or software failures.
7. Security: Payara Server includes robust security features to protect applications and resources. It supports secure communication over SSL/TLS protocols and provides authentication, authorization, and secure storage of sensitive information. It integrates with external user registries, LDAP directories, and identity management systems for centralized user management. It also supports fine-grained security policies and role-based access control (RBAC) to control access to resources.
8. MicroProfile and Jakarta EE Support: Payara Server supports MicroProfile, a set of APIs and specifications for building microservices-based applications. It also supports Jakarta EE, the successor to Java EE, ensuring compatibility with the latest standards and enabling developers to take advantage of the latest features and technologies.
9. Community and Support: Payara Server benefits from an active and growing community of developers and users. The community provides support, documentation, forums, and resources for troubleshooting, knowledge sharing, and community-driven development. Payara Ltd., the organization behind Payara Server, also offers commercial support and enterprise editions with additional features and support services.
10. Cloud-Native Capabilities: Payara Server supports cloud-native application development and deployment. It provides features and tools for containerization and integration with container orchestration platforms like Kubernetes. It enables seamless deployment and management of applications in cloud environments, facilitating scalability and flexibility.

Payara Server is a reliable, feature-rich, and well-supported Java EE application server that offers excellent performance, scalability, and management capabilities. It is a popular choice for organizations looking for a robust and flexible platform to deploy their Java applications in enterprise environments.