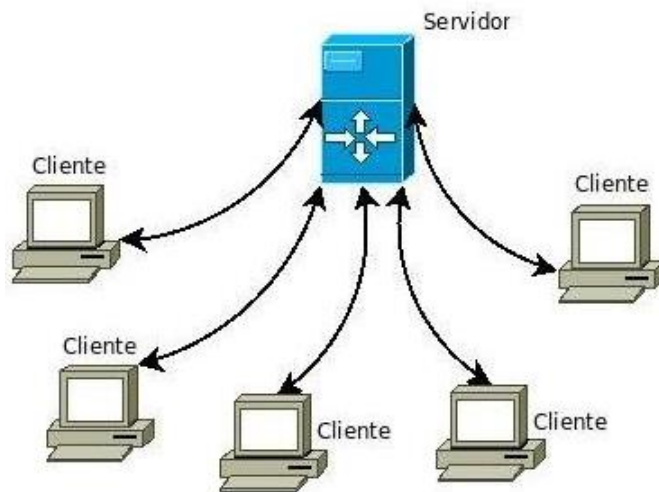


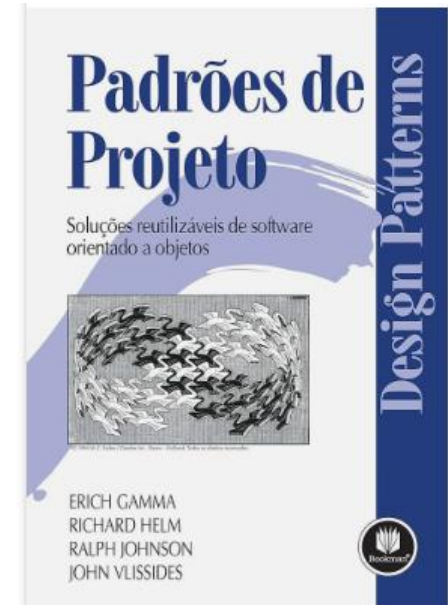
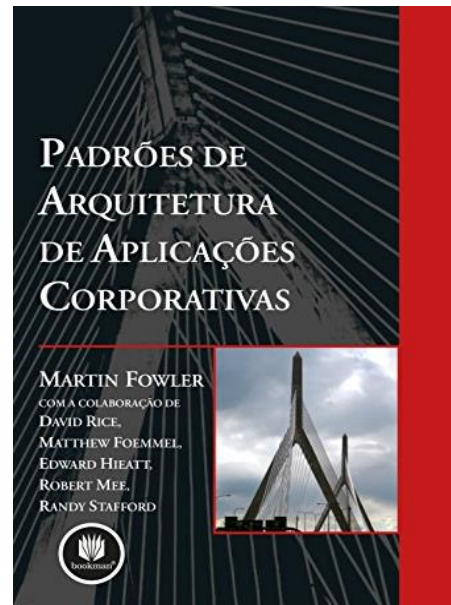
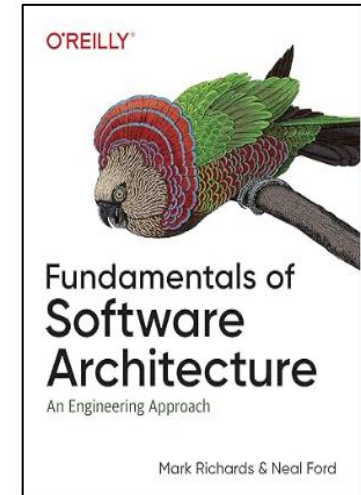
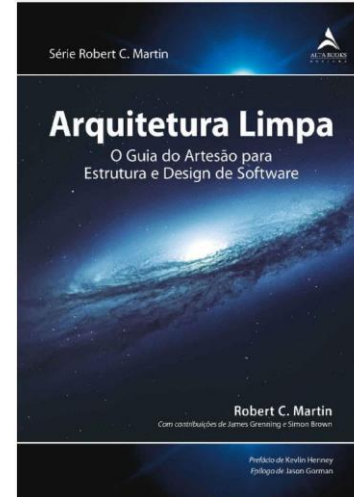
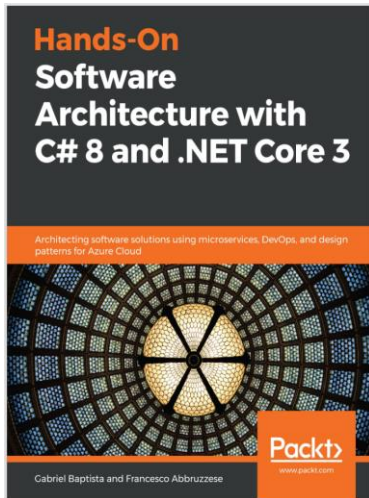
Arquitetura de Software

Unidade 5 – Arquitetura Cliente Servidor



Prof. Aparecido V. de Freitas
Doutor em Engenharia
da Computação pela EPUVSP
aparecido.freitas@online.uscs.edu.br
aparecidovfreitas@gmail.com

Bibliografia

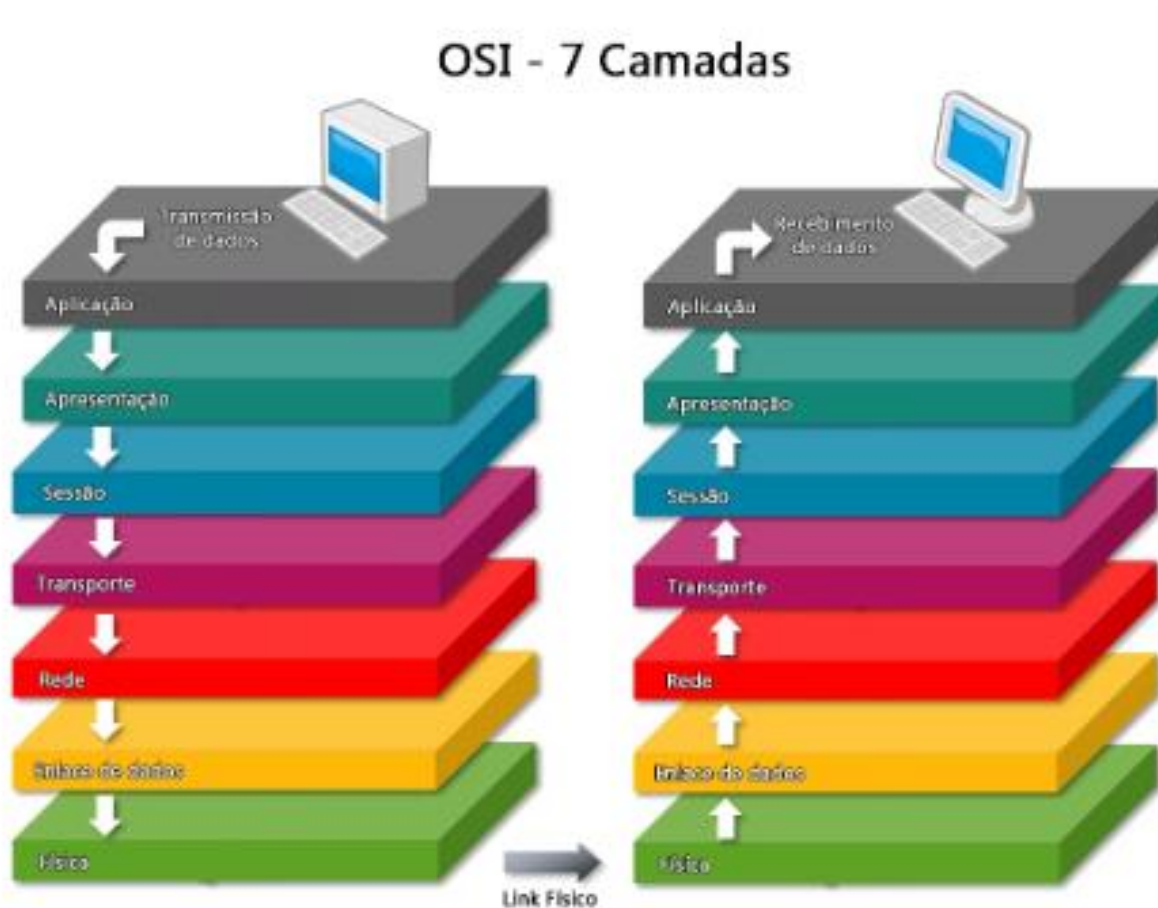


Camadas

- ⊕ Vimos na unidade anterior, a **Arquitetura em Camadas**;
- ⊕ Na arquitetura em camadas, uma **camada** é uma coleção **logicamente separada** de funcionalidades que tem um papel específico dentro do software;
- ⊕ Cada camada tem **responsabilidades únicas** e opera em um nível de abstração diferente.



Arquitetura em Camadas



Arquitetura em Camadas



Arquitetura em Camadas

- ⊕ Lembrando..
- ⊕ O objetivo é separar as preocupações do software de modo que as partes relacionadas a aspectos distintos da aplicação (como **interface de usuário, lógica de negócios, acesso a dados, etc.**) estejam isoladas umas das outras;
- ⊕ Isso facilita a **manutenção** e a **escalabilidade**, pois as mudanças em uma **camada** têm impacto **mínimo** ou nenhum sobre as outras.





Em uma arquitetura de camadas, como pode ser feita a comunicação entre as camadas ?

Comunicação entre Camadas

- ⊕ Na arquitetura de camadas, a comunicação entre as camadas é geralmente realizada de forma estruturada e organizada;
- ⊕ Existem várias técnicas e padrões para implementar essa comunicação, incluindo:
 - ✓ **Interface de Programação de Aplicações (API)**, por exemplo, chamadas JDBC a um **servidor de Banco de Dados** implementado um **nó da rede**;
 - ✓ **Chamadas de Função ou Método**: As camadas podem se comunicar diretamente, chamando funções ou métodos fornecidos por outras camadas;
 - ✓ **Mensageria**;
 - ✓ **Eventos**;
 - ✓ **Web Services**.

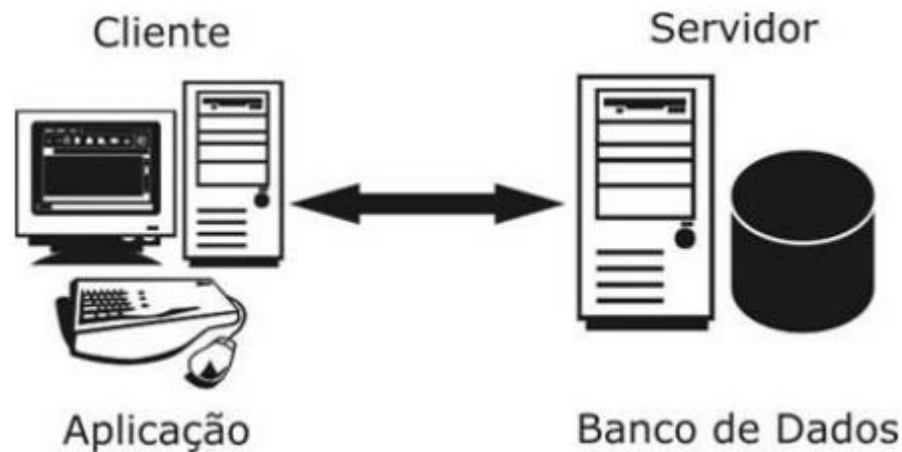
Arquitetura em Camadas (Cliente Servidor)

- Em uma aplicação desenvolvida com arquitetura em camadas, caso a camada de acesso à dados seja efetivada por meio de **chamadas de biblioteca, através de uma rede**, para se conectar a um servidor de banco de dados, a arquitetura também é denominada **Cliente Servidor**.



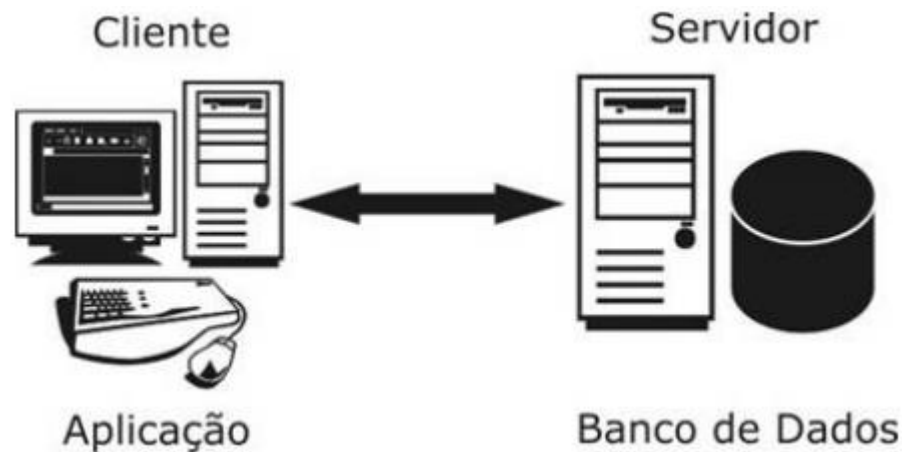
Arquitetura Cliente-Servidor

- ✦ A arquitetura **cliente-servidor** é um modelo de design de software onde tarefas e cargas de trabalho são distribuídas entre **provedores** de recursos ou serviços, conhecidos como **servidores**, e solicitantes de serviços, chamados **clientes**.



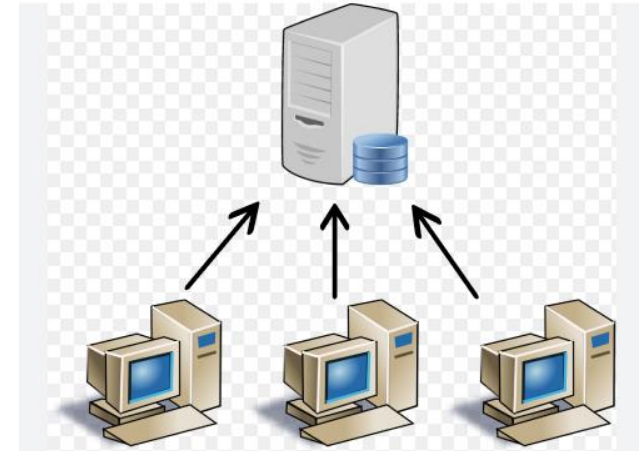
Arquitetura Cliente-Servidor

- ✦ A arquitetura **cliente-servidor** é um modelo de design de software onde tarefas e cargas de trabalho são distribuídas entre **provedores** de recursos ou serviços, conhecidos como **servidores**, e solicitantes de serviços, chamados **clientes**.



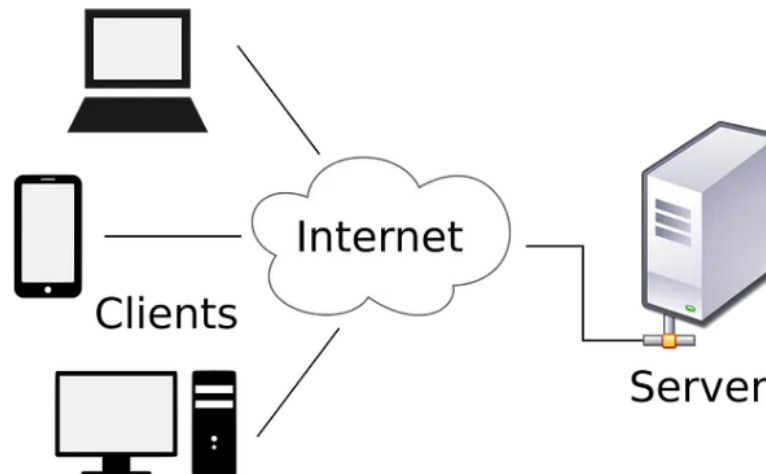
Servidor

- ❖ O **Servidor**: Imagine um restaurante. O servidor é como a **cozinha**, onde a comida (neste caso, os dados ou serviços) é preparada;
- ❖ No mundo da tecnologia, um **servidor** é um sistema de computador ou um programa que está configurado para fornecer recursos, dados, serviços ou programas para outros computadores, conhecidos como clientes, sob demanda;
- ❖ Exemplos comuns de servidores incluem servidores web, servidores de e-mail e servidores de banco de dados.



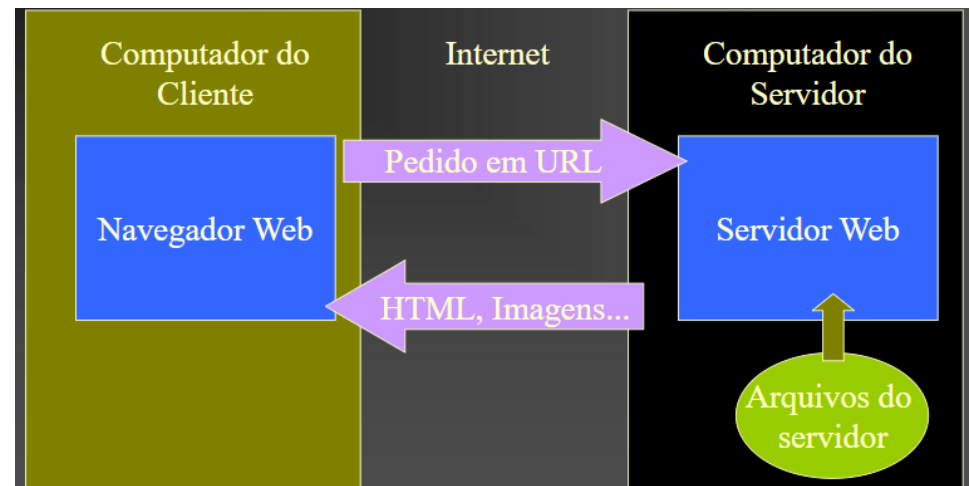
Cliente

- ⊕ O **Cliente**: Continuando com a analogia do restaurante, o **cliente** é a pessoa que pede a comida;
- ⊕ Em termos de computação, um **cliente** é um programa de computador ou dispositivo que solicita serviços ou recursos ao servidor;
- ⊕ Por exemplo, quando você usa um navegador para acessar uma página da web, o navegador é o **cliente** que solicita as páginas web ao **servidor**.



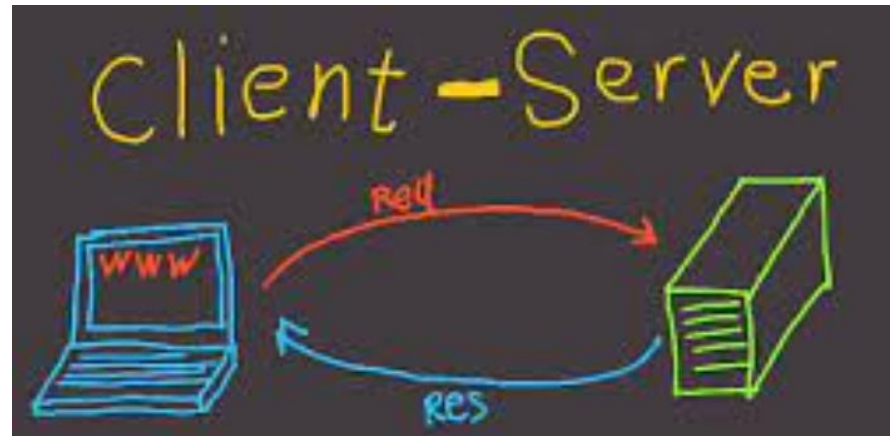
Como funciona ?

- ⊕ Quando um **cliente** precisa de um recurso ou serviço, ele envia uma solicitação ao servidor;
- ⊕ O **servidor**, após receber a solicitação, processa-a e retorna a resposta ou o recurso solicitado ao **cliente**;
- ⊕ Por exemplo, ao acessar um site, seu navegador (**cliente**) envia uma solicitação ao servidor que hospeda o site. O **servidor** processa esta solicitação e envia de volta os dados da página da web, que o navegador exibe.



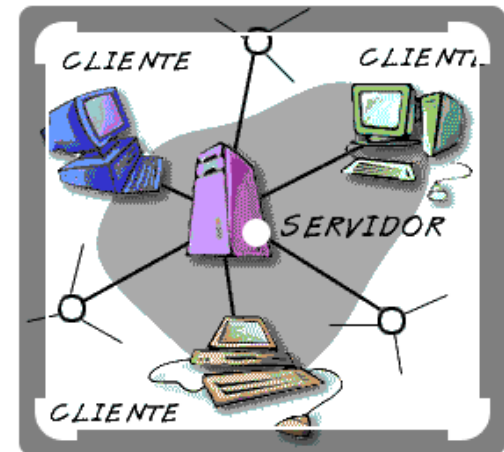
Vantagens

- ⊕ Esta arquitetura tem várias **vantagens**. Ela permite a **centralização** de **dados** e **recursos**, o que facilita a manutenção e a atualização;
- ⊕ Além disso, a distribuição de tarefas pode melhorar a eficiência e escalabilidade, pois os **servidores** podem atender a muitos clientes ao mesmo tempo.



Exemplos no Dia-a-Dia

- ⊕ Exemplos comuns de aplicações cliente-servidor incluem **navegação na web** (navegador como cliente, servidor web como servidor), **e-mail** (aplicativo de **e-mail** como **cliente**, servidor de e-mail como **servidor**), e muitas aplicações móveis que se conectam a servidores na internet para buscar informações.
- ⊕ Portanto, a arquitetura **Cliente-Servidor** é uma maneira fundamental de organizar e distribuir tarefas e recursos em sistemas de computação, tornando-a uma das **arquiteturas** mais comuns e importantes no mundo da tecnologia.



Exemplo de uma aplicação **console** desenvolvida em **Java** com a Arquitetura Cliente-Servidor



Arquitetura Cliente Servidor – Exemplo

```
1 var now = new Date();
2 var hours = now.getHours();
3 var minutes = now.getMinutes();
4 var seconds = now.getSeconds();
5
6 var ampm = "am";
7 var colon = '<IMG SRC="images/colon.gif">';
8
9 if (hours >= 12) {
10     ampm = "pm";
11     hours = hours - 12;
12 }
13
14 if (hours == 0) hours = 12;
15
16 if (hours < 10) hours = "0" + hours;
17 else hours = hours + '';
18
19 if (minutes < 10) minutes = "0" + minutes;
20 else minutes = minutes + '';
21
22 if (seconds < 10) seconds = "0" + seconds;
23 else seconds = seconds + '';
```



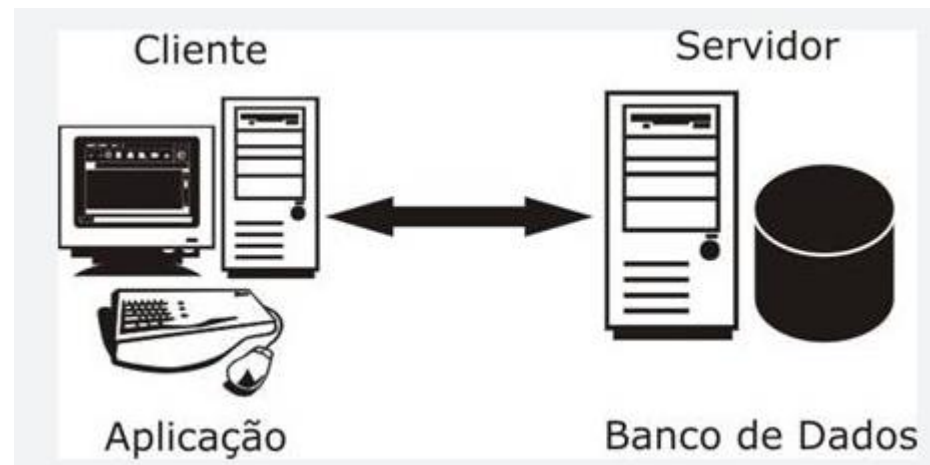
Código **Java**, com arquitetura **Cliente-Servidor**, que acessa um **servidor** de Banco de Dados **MySQL** que mantém uma lista com **10 perguntas** sobre as capitais de diferentes países;

A aplicação, **via console**, envia a pergunta e o usuário digita a resposta e envia para o programa. No final do **quiz**, são mostradas as respostas corretas, erradas e a nota do usuário (de 0 a 10). A aplicação é **Cliente-Servidor** uma vez que a aplicação faz requisições ao Servidor de Banco de Dados para checar a veracidade da resposta do usuário.

Arquitetura Cliente Servidor – Exemplo

Para se escrever a aplicação **Cliente-Servidor** para se empregar um banco de dados **MySQL**, será necessário realizar-se algumas etapas:

- ❖ Criar uma tabela **MySQL** para armazenar as informações;
- ❖ Modificar a aplicação para conectar-se ao banco de dados e buscar as informações;
- ❖ Incluir manualmente o driver **MySQL** na aplicação.



Arquitetura Cliente Servidor – Exemplo

- ❖ Criar uma tabela **MySQL** para armazenar as informações;

```
CREATE DATABASE IF NOT EXISTS quiz_capitais;
```

```
USE quiz_capitais;
```

```
CREATE TABLE capitais (  
    pais VARCHAR(100) PRIMARY KEY,  
    capital VARCHAR(100) NOT NULL  
);
```

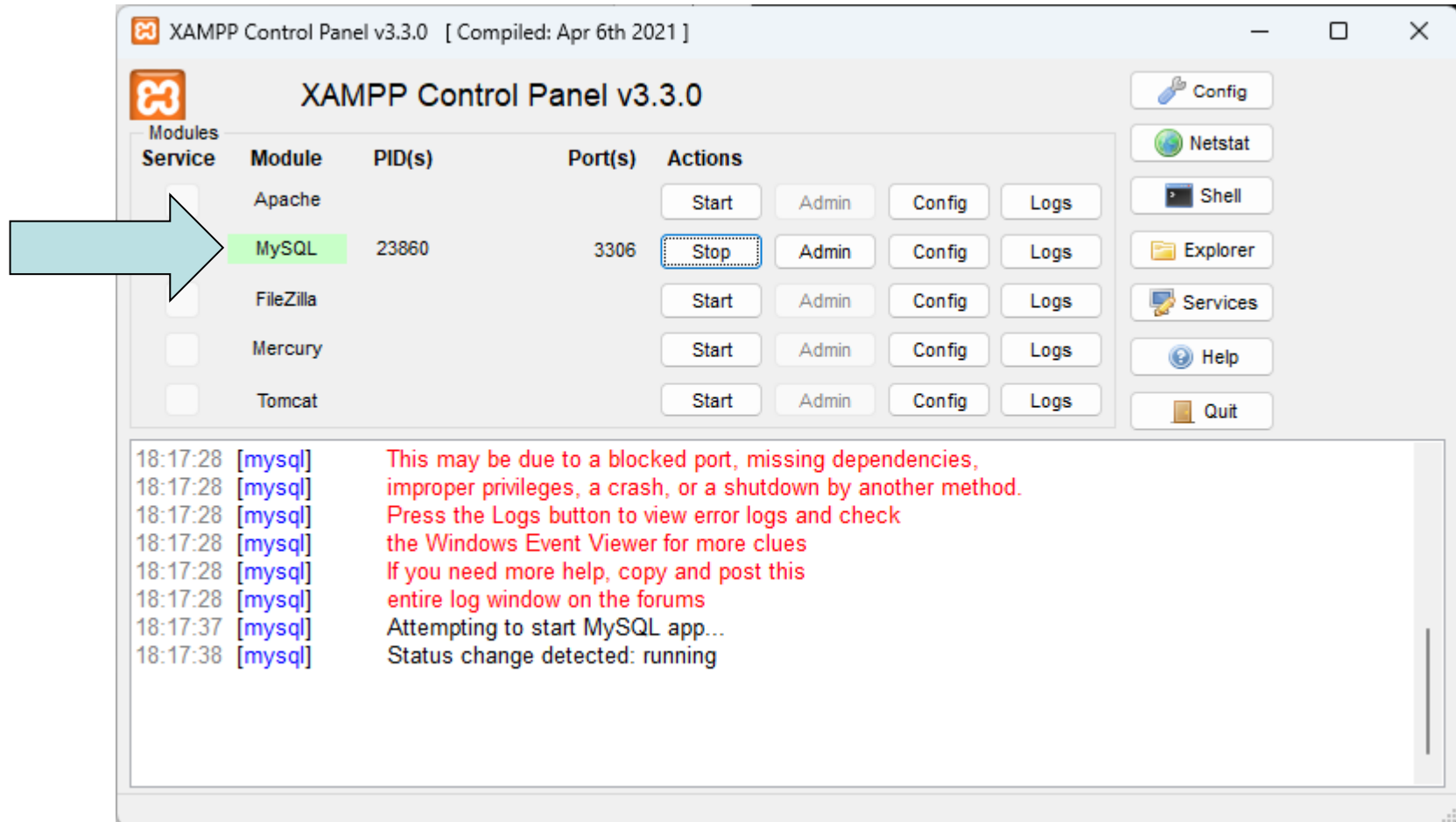
```
INSERT INTO capitais (pais, capital) VALUES ('Brasil', 'Brasília');
```

```
INSERT INTO capitais (pais, capital) VALUES ('França', 'Paris');
```

```
-- Adicione os outros países e capitais aqui
```

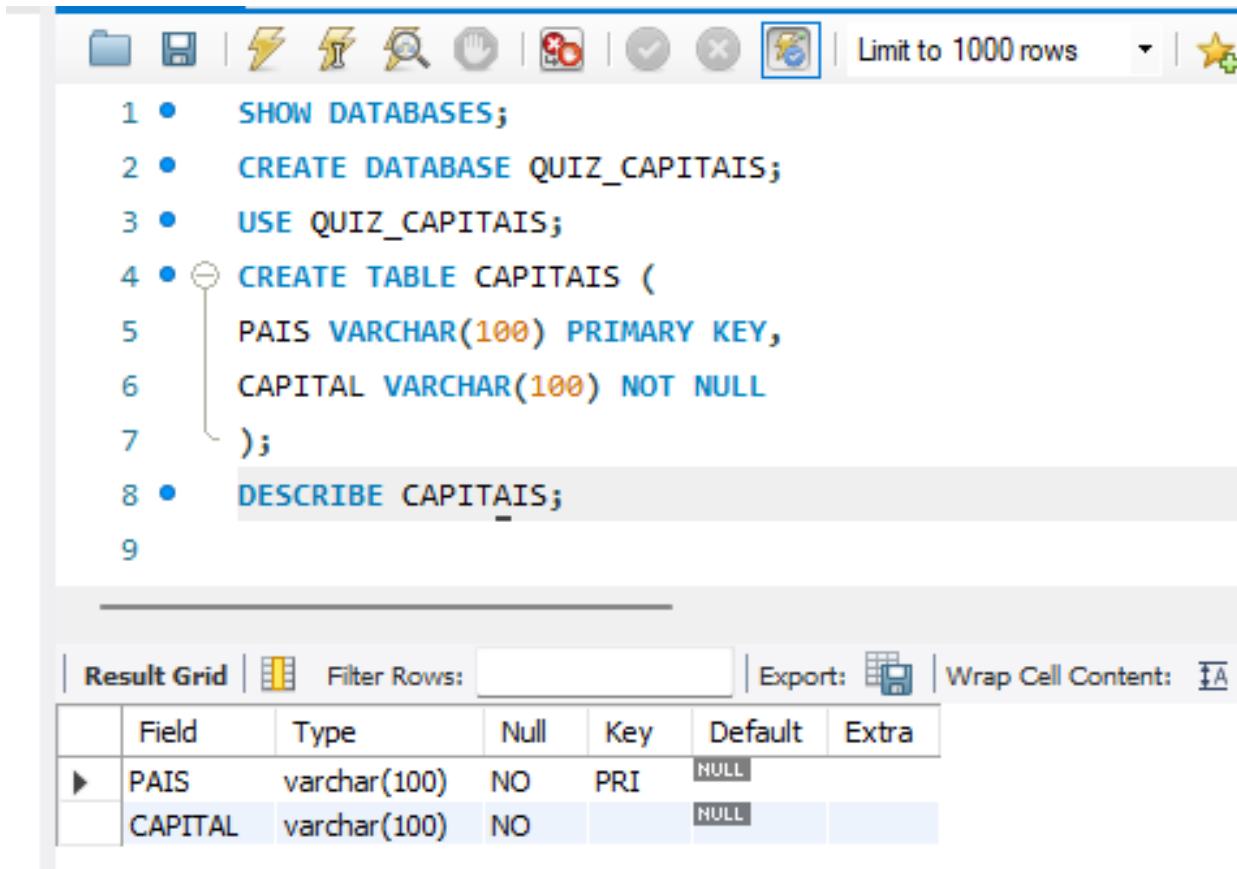
Arquitetura Cliente Servidor – Exemplo

❖ Iniciando o Servidor de Banco de Dados



Arquitetura Cliente Servidor – Exemplo

❖ Acessando o Servidor com Workbench



The screenshot shows the MySQL Workbench interface. The SQL editor contains the following commands:

```

1 • SHOW DATABASES;
2 • CREATE DATABASE QUIZ_CAPITAIS;
3 • USE QUIZ_CAPITAIS;
4 • CREATE TABLE CAPITAIS (
5     PAIS VARCHAR(100) PRIMARY KEY,
6     CAPITAL VARCHAR(100) NOT NULL
7 );
8 • DESCRIBE CAPITAIS;
9


```

The result grid for the `DESCRIBE CAPITAIS` query is displayed below the editor:

	Field	Type	Null	Key	Default	Extra
▶	PAIS	varchar(100)	NO	PRI	NULL	
	CAPITAL	varchar(100)	NO		NULL	

Arquitetura Cliente Servidor – Exemplo

❖ Acessando o Servidor com Workbench

Result Grid | Filter Rows: | Export: | Wrap Cell Content: 

	Field	Type	Null	Key	Default	Extra
▶	PAIS	varchar(100)	NO	PRI	NULL	
	CAPITAL	varchar(100)	NO		NULL	

Result 1 x

Output

📄 Action Output

	#	Time	Action	Message
✓	1	18:22:26	CREATE DATABASE QUIZ_CAPITAIS	1 row(s) affected
✓	2	18:22:48	USE QUIZ_CAPITAIS	0 row(s) affected
✓	3	18:23:52	CREATE TABLE CAPITAIS (PAIS VARCHAR(100) PRIMARY KEY, CAPI...	0 row(s) affected
✓	4	18:24:08	DESCRIBE CAPITAIS	2 row(s) returned

Arquitetura Cliente Servidor – Exemplo

❖ Acessando o Servidor com Cliente MySQL

```
XAMPP for Windows - mysql
Setting environment for using XAMPP for Windows.
apare@PC1-CIDO c:\xampp
# mysql -uroot -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 11
Server version: 10.4.28-MariaDB mariadb.org binary distribution

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

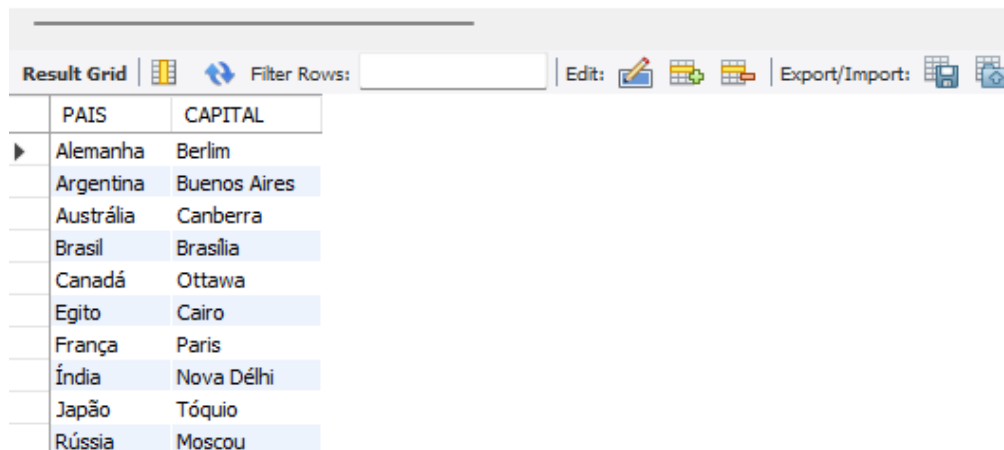
MariaDB [(none)]> use quiz_capitais;
Database changed
MariaDB [quiz_capitais]> describe capitais;
+-----+-----+-----+-----+-----+-----+
| Field | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| PAIS  | varchar(100)  | NO   | PRI | NULL    |       |
| CAPITAL | varchar(100) | NO   |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
2 rows in set (0.009 sec)

MariaDB [quiz_capitais]> |
```

Arquitetura Cliente Servidor – Exemplo

❖ Inserindo dados no Banco de Dados

```
3 • USE QUIZ_CAPITAIS;  
4 • INSERT INTO CAPITAIS (PAIS, CAPITAL) VALUES  
5     ("Brasil", "Brasília"),  
6     ("França", "Paris"),  
7     ("Japão", "Tóquio"),  
8     ("Alemanha", "Berlim"),  
9     ("Canadá", "Ottawa"),  
10    ("Índia", "Nova Délhi"),  
11    ("Rússia", "Moscou"),  
12    ("Austrália", "Canberra"),  
13    ("Argentina", "Buenos Aires"),  
14    ("Egito", "Cairo");  
15  
16 • select * from capitais;
```



The screenshot shows a database application interface. At the top, there is a toolbar with icons for 'Result Grid', 'Filter Rows', 'Edit', and 'Export/Import'. Below the toolbar is a table with two columns: 'PAIS' and 'CAPITAL'. The table contains the following data:

PAIS	CAPITAL
Alemanha	Berlim
Argentina	Buenos Aires
Austrália	Canberra
Brasil	Brasília
Canadá	Ottawa
Egito	Cairo
França	Paris
Índia	Nova Délhi
Japão	Tóquio
Rússia	Moscou

Arquitetura Cliente Servidor – Exemplo

❖ Inserindo dados no Banco de Dados

```
XAMPP for Windows - mysql
MariaDB [quiz_capitais]> describe capitais;
+-----+-----+-----+-----+-----+-----+
| Field | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| PAIS   | varchar(100)  | NO   | PRI | NULL    |       |
| CAPITAL | varchar(100)  | NO   |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
2 rows in set (0.009 sec)

MariaDB [quiz_capitais]> select * from capitais;
+-----+-----+
| PAIS      | CAPITAL      |
+-----+-----+
| Alemanha  | Berlim       |
| Argentina | Buenos Aires |
| Austrália | Canberra     |
| Brasil    | Brasília     |
| Canadá    | Ottawa       |
| Egito     | Cairo        |
| França    | Paris        |
| Índia     | Nova Délhi   |
| Japão     | Tóquio       |
| Rússia    | Moscou       |
+-----+-----+
10 rows in set (0.000 sec)

MariaDB [quiz_capitais]> |
```

Arquitetura Cliente Servidor – Exemplo

❖ Baixando Driver JDBC – Maria DB



PRODUCTS SERVICES PRICING RESOURCES

Knowledge Base » Server & Client Software » Client Libraries » Application Programming Interfaces » Java Connector » About MariaDB Connector/J

Home

Open Questions

MariaDB Server

MariaDB MaxScale

MariaDB ColumnStore

Connectors

History

Source

Flag as Spam /
Inappropriate

Translate

Created

About MariaDB Connector/J

The most recent **Stable (GA)** release of MariaDB Connector/J is:

MariaDB Connector/J 3.3.2

[Download MariaDB Connector/J](#)

MariaDB Connector/J is used to connect applications developed in Java to MariaDB and MySQL databases using the standard JDBC API. The library is LGPL licensed.

Date	Release	Status	Min. Java Compat.	Release Notes	Changelog
19 Dec 2023	3.3.2	Stable (GA)	Java 8	Release Notes	Changelog
29 Nov 2023	3.3.1	Stable (GA)	Java 8	Release Notes	Changelog
25 Aug 2023	3.2.0	Stable (GA)	Java 8	Release Notes	Changelog
08 Nov 2023	2.7.11	Stable (GA)	Java 8	Release Notes	Changelog

Arquitetura Cliente Servidor – Exemplo

❖ Baixando Driver JDBC – Maria DB



PRODUCTS SERVICES PRICING RESOURCES DEVELOPERS COMPANY

DOWNLOAD MARIADB PRODUCTS AND TOOLS FOR DEVELOPMENT AND PRODUCTION

FREE

MariaDB Community

MariaDB Enterprise

Repo Setup

Connectors

Data Access

Lightweight, advanced connectors for high-performance data access and data streaming.

MariaDB Connector/J is a lightweight JDBC driver (Type 4) for building Java applications on top of MariaDB, complete with built-in connection pooling and encrypted connections via TLS/SSL. MariaDB Connector/J is LGPL.

This version of MariaDB Connector/J is for Java 8, Java 11, and Java 17.

Product

Java 8+ connector

Version

3.3.2-GA

OS

Platform Independent

Arquitetura Cliente Servidor – Exemplo

❖ Configuração do Driver num Projeto Java com Eclipse

1. Baixar o Driver JDBC do MariaDB:

Primeiro, você precisa baixar o driver JDBC do MariaDB. Isso pode ser feito a partir do site oficial do MariaDB ou de repositórios como o Maven.

2. Adicionar o Driver ao seu Projeto:

- Abra o Eclipse e o seu projeto Java.
- Clique com o botão direito no projeto e selecione "Properties".
- No painel esquerdo, clique em "Java Build Path".
- Vá para a aba "Libraries" e clique no botão "Add External JARs...".
- Selecione o arquivo JAR do driver JDBC que você baixou e clique em "Open".

Arquitetura Cliente Servidor – Exemplo

❖ Configuração do Driver num Projeto Java com Eclipse

3. Configurar a Conexão:

- No seu código Java, você precisará configurar a conexão com o banco de dados MariaDB. Isso inclui especificar a URL de conexão, o nome de usuário e a senha.
- Aqui está um exemplo de como você pode fazer isso:

java

Copy code

```
String url = "jdbc:mariadb://localhost:3306/seuBancoDeDados";  
String user = "seuUsuario";  
String password = "suaSenha";  
  
Connection con = DriverManager.getConnection(url, user, password);
```

- Não se esqueça de tratar exceções, como `SQLException`.

Arquitetura Cliente Servidor – Exemplo

❖ Configuração do Driver num Projeto Java com Eclipse

4. Desenvolver seu Código:

- Agora você pode começar a escrever seu código para interagir com o banco de dados MariaDB usando JDBC.
- Isso inclui criar `Statement` ou `PreparedStatement`, executar consultas SQL e processar os resultados.



Arquitetura Cliente Servidor – Exemplo

❖ Configuração do Driver num Projeto Java com Eclipse

5. Testar a Conexão:

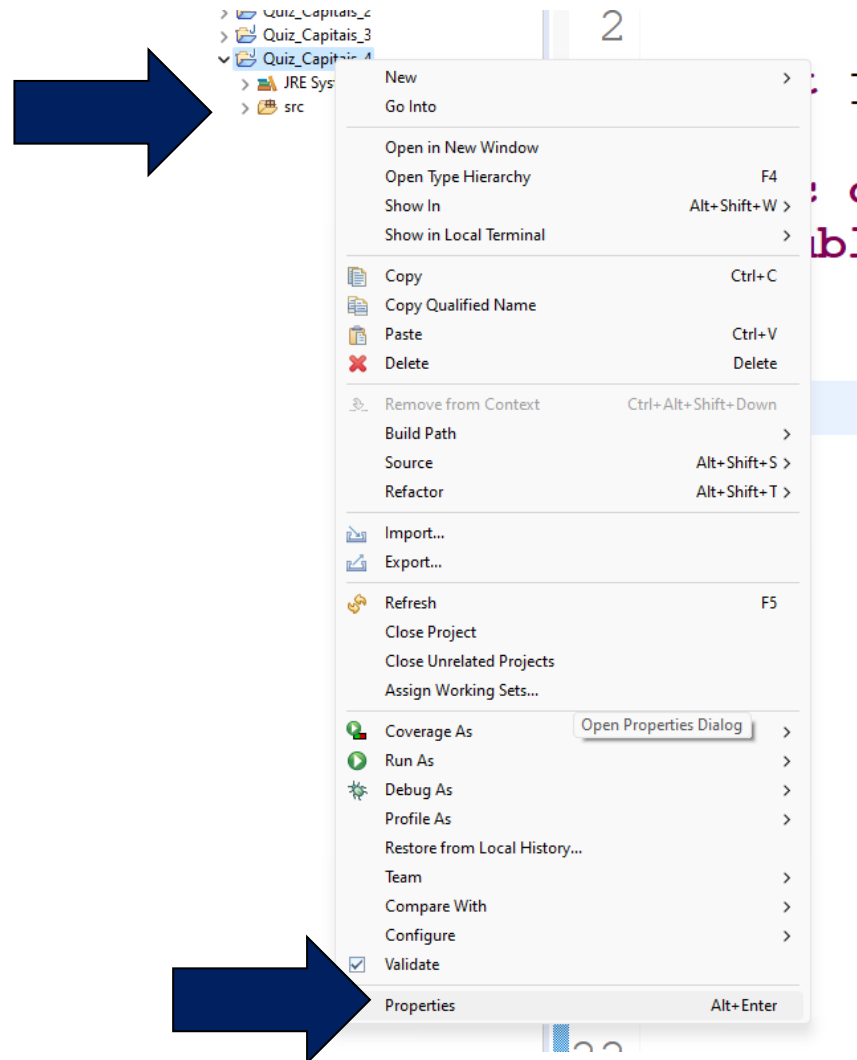
- Depois de escrever o código, execute o aplicativo para testar a conexão com o banco de dados.
- Se houver algum problema, verifique a URL de conexão, nome de usuário, senha e a configuração do driver JDBC.

Lembre-se de que esses passos são genéricos e podem variar um pouco dependendo da versão do Eclipse e do driver JDBC do MariaDB que você está usando. Se você encontrar problemas específicos, é útil consultar a documentação do MariaDB e do Eclipse para obter mais detalhes.



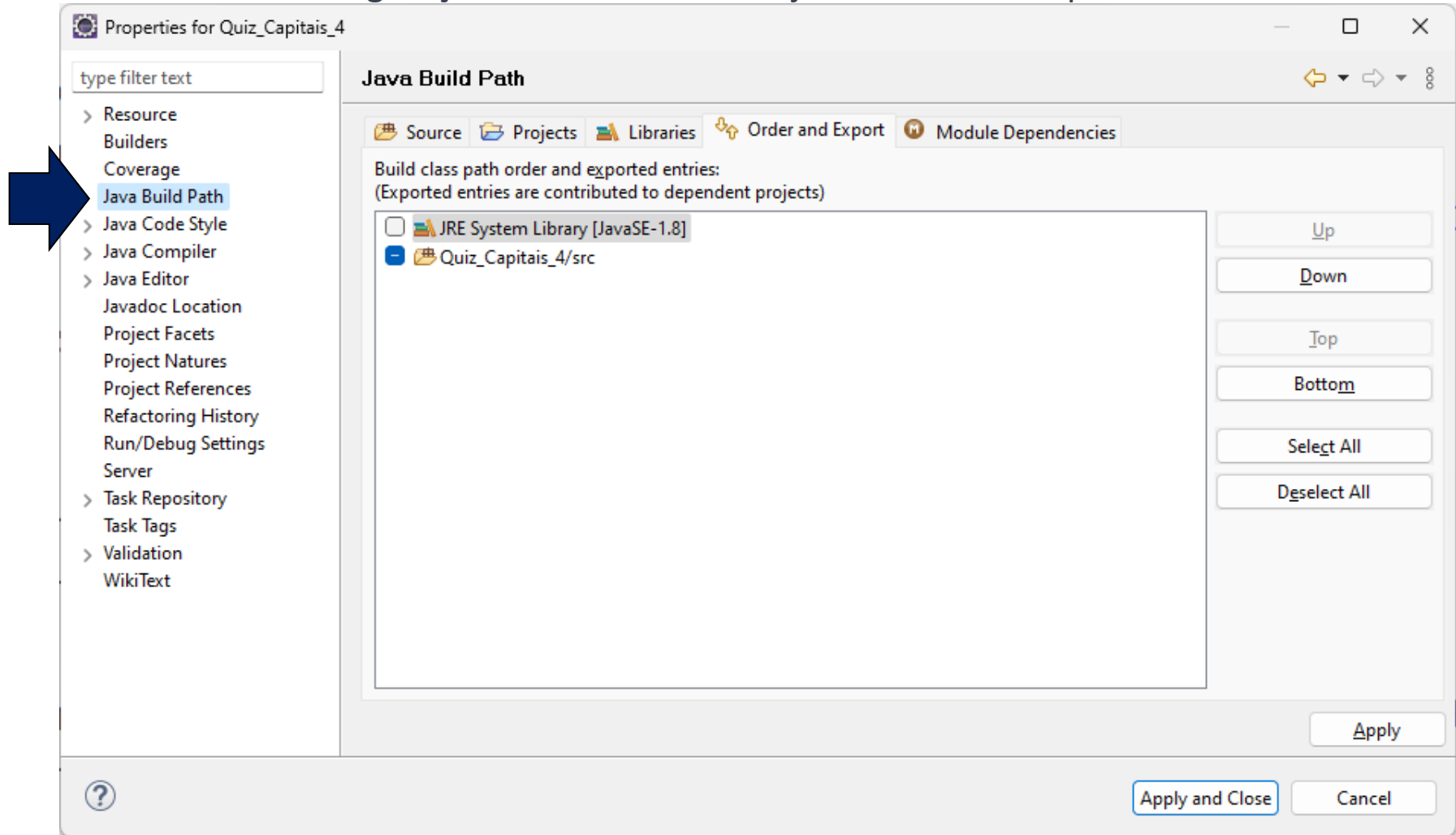
Arquitetura Cliente Servidor – Exemplo

❖ Configuração do Driver num Projeto Java com Eclipse



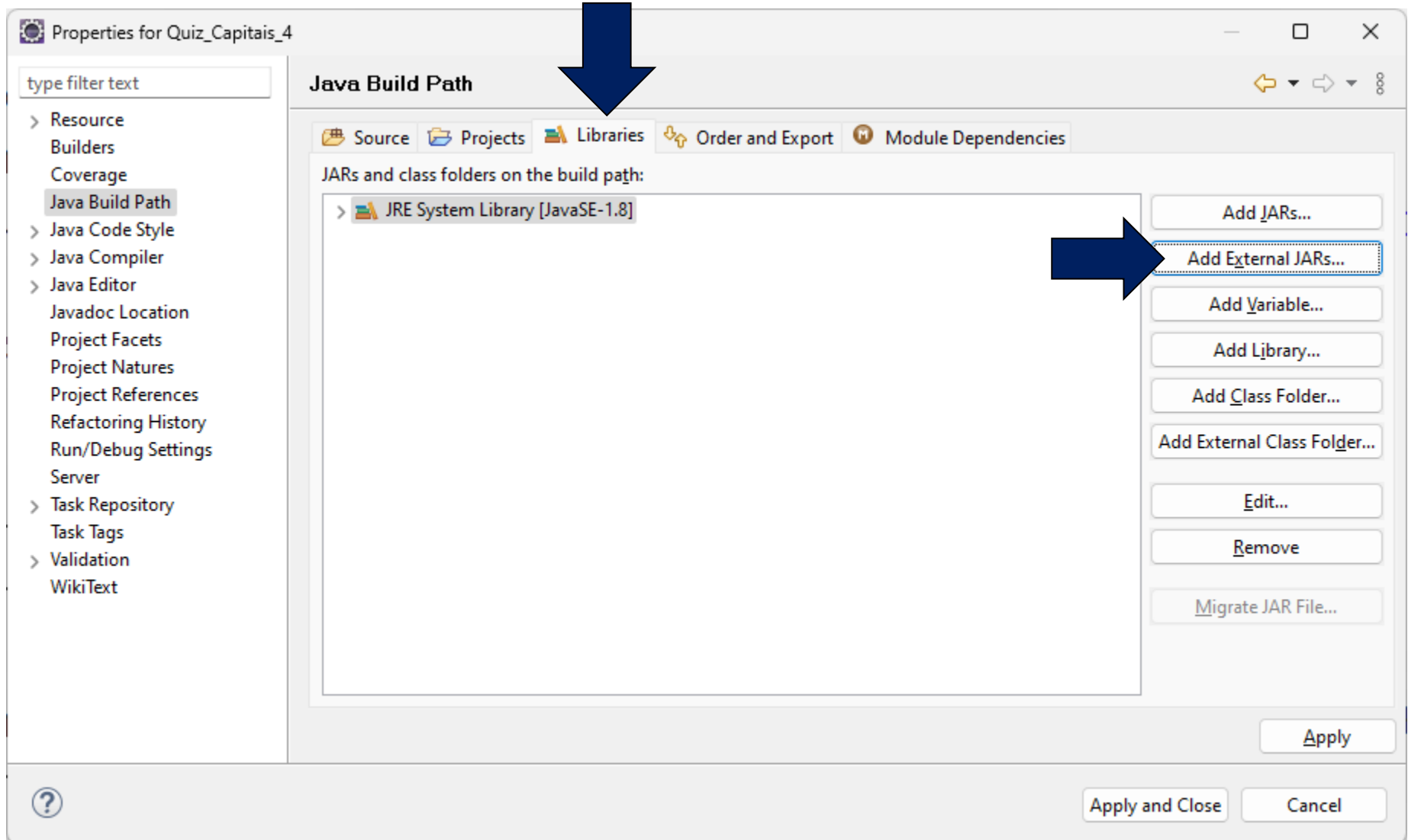
Arquitetura Cliente Servidor – Exemplo

❖ Configuração do Driver num Projeto Java com Eclipse



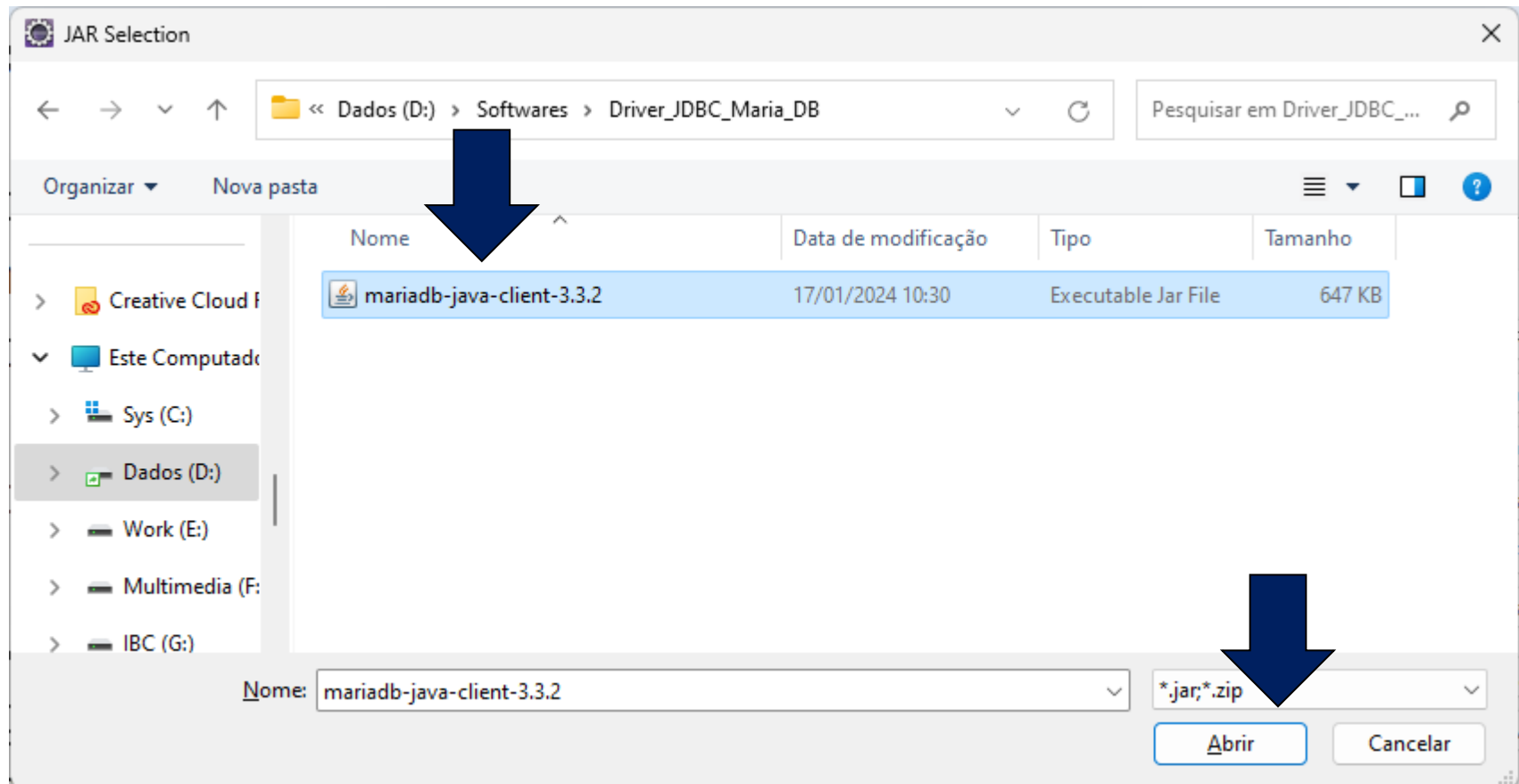
Arquitetura Cliente Servidor – Exemplo

❖ Configuração do Driver num Projeto Java com Eclipse



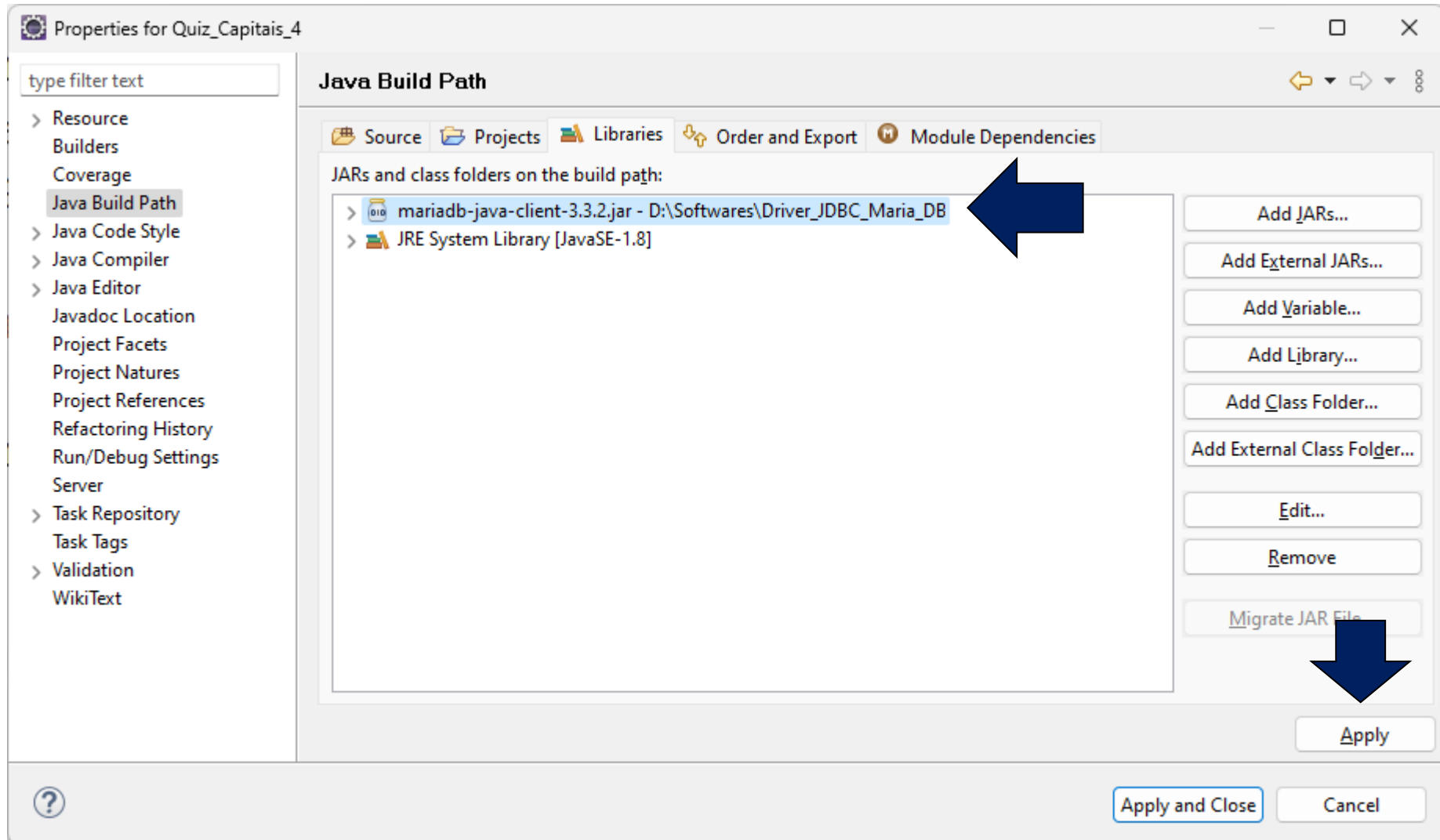
Arquitetura Cliente Servidor – Exemplo

❖ Configuração do Driver num Projeto Java com Eclipse



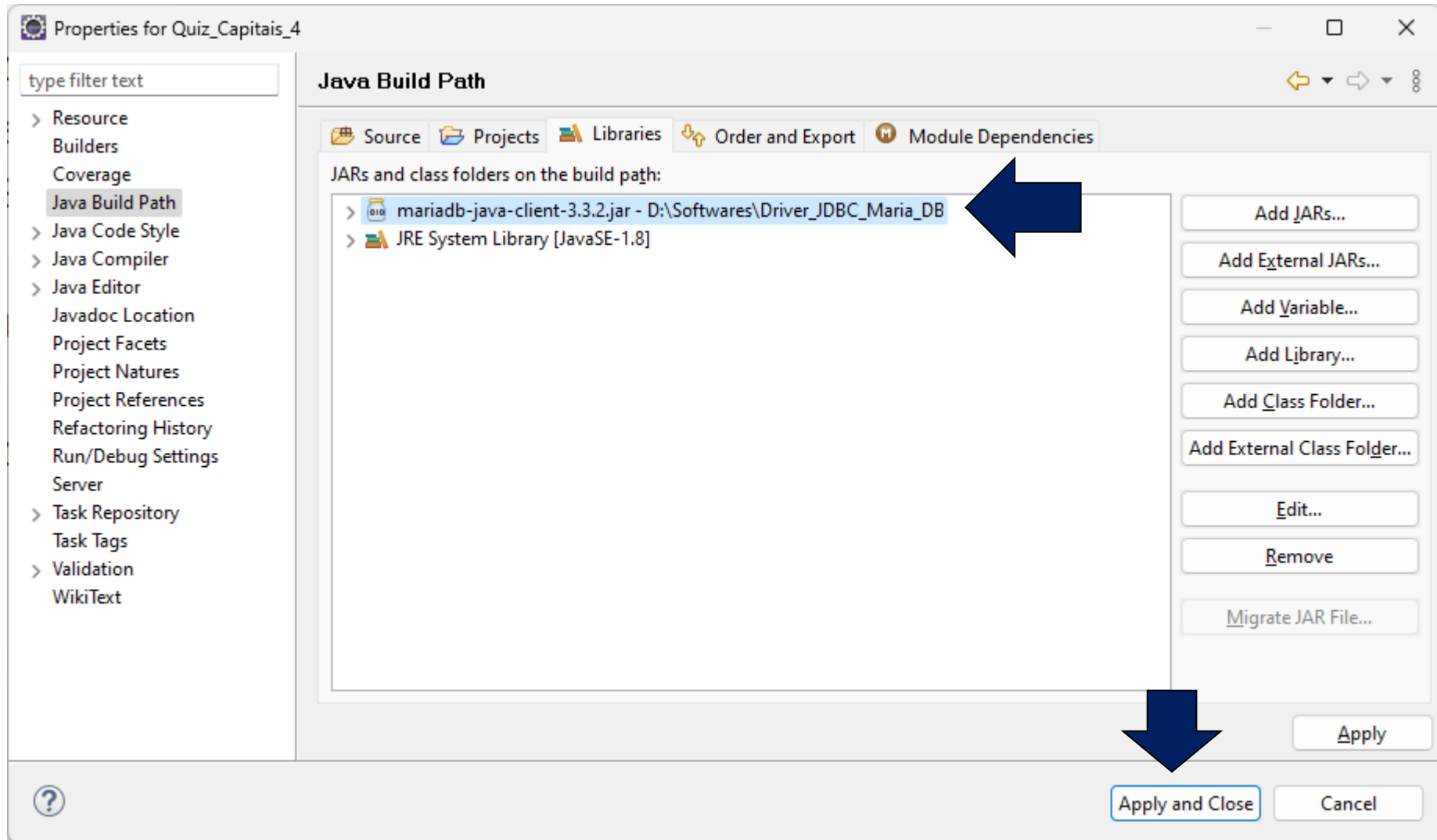
Arquitetura Cliente Servidor – Exemplo

❖ Configuração do Driver num Projeto Java com Eclipse



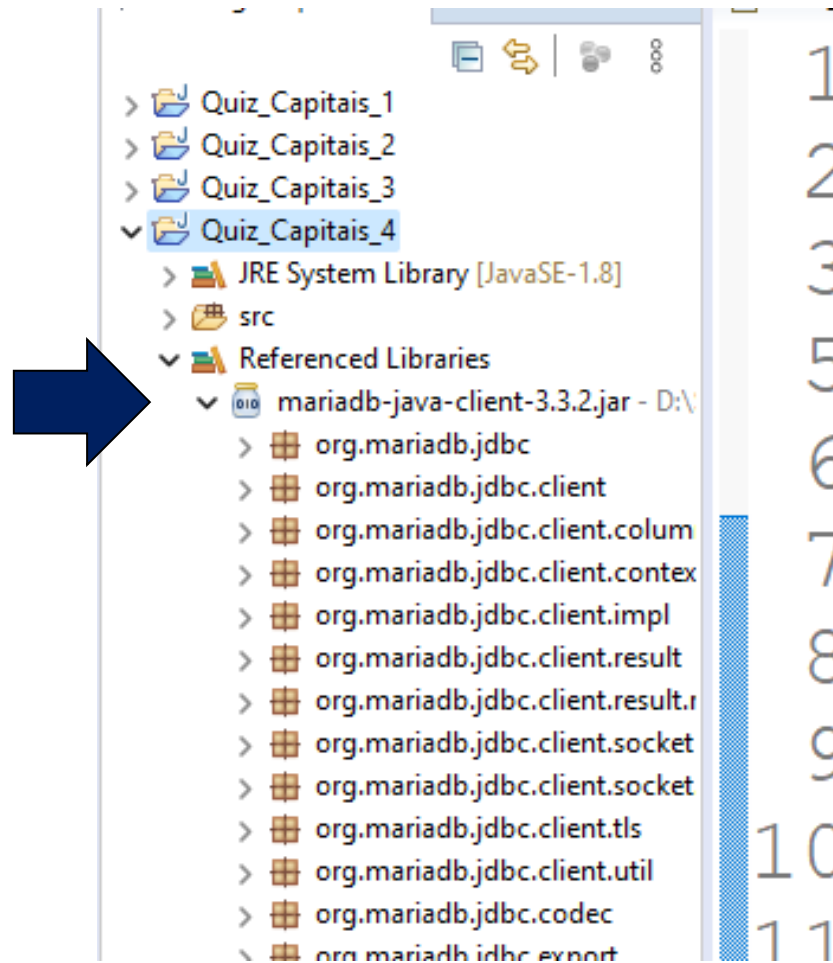
Arquitetura Cliente Servidor – Exemplo

❖ Configuração do Driver num Projeto Java com Eclipse



Arquitetura Cliente Servidor – Exemplo

- ❖ Configuração do Driver num Projeto Java com Eclipse



Arquitetura Cliente Servidor – Exemplo

❖ Projeto Java com Eclipse

```
package br.uscs;

import java.sql.*;
import java.util.Scanner;

public class Quiz_Capitais_4 {
    public static void main(String[] args) {
        String url = "jdbc:mysql://localhost:3306/quiz_capitais";
        String user = "root";
        String password = "";
        int score = 0;

        try (Connection conn = DriverManager.getConnection(url, user, password);
            Statement stmt = conn.createStatement();
            ResultSet rs = stmt.executeQuery("SELECT pais, capital FROM capitais")) {
```

Arquitetura Cliente Servidor – Exemplo

❖ Projeto Java com Eclipse

```
Scanner scanner = new Scanner(System.in);

while (rs.next()) {
    String pais = rs.getString("pais");
    String capital = rs.getString("capital");

    System.out.println("Qual é a capital do país " + pais + "?");
    String respostaUsuario = scanner.nextLine();

    if (respostaUsuario.trim().equalsIgnoreCase(capital)) {
        System.out.println("Correto!");
        score++;
    } else {
        System.out.println("Errado! A resposta correta é: " + capital);
    }
}

scanner.close();
```

Arquitetura Cliente Servidor – Exemplo

❖ Projeto Java com Eclipse

```
Integer count = getRowCount(stmt);
System.out.println("\nO quiz terminou. Você acertou " + score +
    " de " + count + " perguntas.");
System.out.println("Sua nota é: " + (score * 10 / count));

} catch (SQLException e) {
    e.printStackTrace();
    System.out.println("Erro ao conectar ao banco de dados.");
}
}
```

Arquitetura Cliente Servidor – Exemplo

❖ Projeto Java com Eclipse

```
private static Integer getRowCount(Statement stmt) throws SQLException {  
    try (ResultSet rs = stmt.executeQuery("SELECT COUNT(*) FROM capitais")) {  
        if (rs.next()) {  
            return rs.getInt(1);  
        } else {  
            return 0;  
        }  
    }  
}
```

Arquitetura Cliente Servidor – Exemplo

❖ Projeto Java com Eclipse

```
20  
21 Scanner scanner = new Scanner(System.in);  
22  
23 while (rs.next()) {
```

Problems @ Javadoc Declaration Console

Quiz_Capitais_4 [Java Application] C:\Program Files\Java\jre-1.8\bin\javaw.exe (17 de jan de 2024 12:30:43)

Qual é a capital do país Alemanha?

Berlim

Correto!

Qual é a capital do país Argentina?





O arquitetura do Projeto que acabamos de desenvolver é Cliente-Servidor?

Cliente-Servidor

A arquitetura cliente-servidor é caracterizada pela divisão de responsabilidades entre dois tipos de entidades: o cliente, que solicita serviços ou recursos, e o servidor, que fornece esses serviços ou recursos. Vamos analisar seu código sob essa perspectiva:

1. Cliente (Aplicação Java):

- Seu código Java atua como o cliente. Ele é responsável por solicitar informações (como dados de países e capitais) e por enviar essas solicitações ao servidor (o banco de dados MySQL).
- O cliente (sua aplicação Java) também processa as respostas recebidas do servidor, implementando a lógica do quiz e interagindo com o usuário através do terminal.



Cliente-Servidor

2. Servidor (Banco de Dados MySQL):

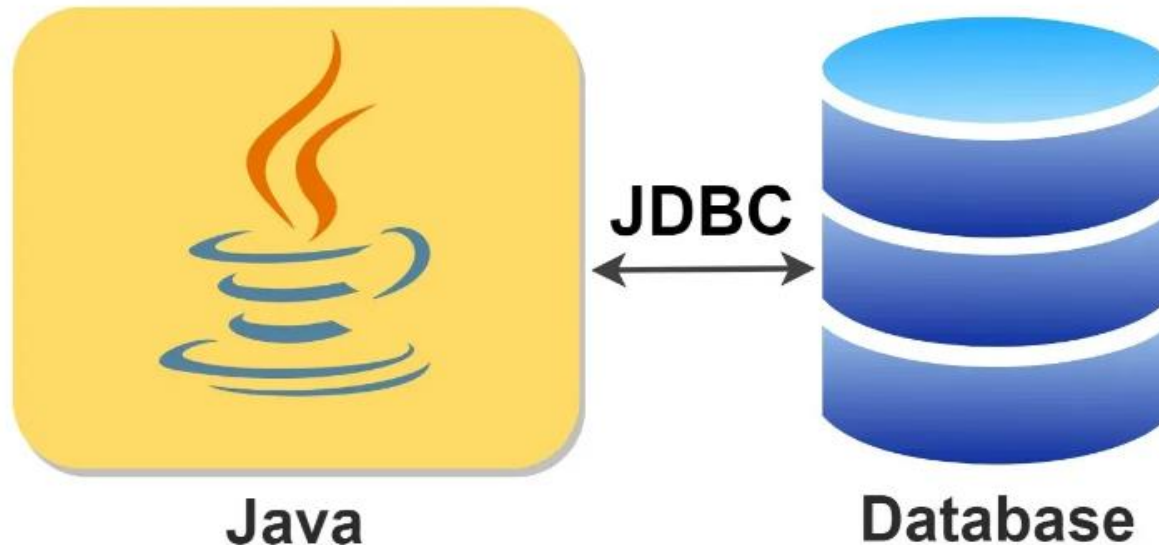
- O servidor no contexto da sua aplicação é o MySQL, um sistema de gerenciamento de banco de dados.
- O servidor é responsável por armazenar os dados (neste caso, informações sobre países e capitais) e processar as solicitações enviadas pelo cliente. Ele executa as consultas SQL recebidas e retorna os resultados ao cliente.



Cliente-Servidor

3. Comunicação entre Cliente e Servidor:

- A comunicação entre o cliente (sua aplicação Java) e o servidor (MySQL) ocorre através do protocolo JDBC (Java Database Connectivity).
- O cliente envia comandos SQL ao servidor, e o servidor retorna os resultados dessas consultas. Esse fluxo de solicitação e resposta é uma característica fundamental da arquitetura cliente-servidor.



Cliente-Servidor

4. Independência e Interação:

- O cliente e o servidor são independentes um do outro, embora interajam para completar tarefas. O cliente pode ser modificado sem alterar o servidor, desde que a interface de comunicação (neste caso, as consultas SQL) permaneça consistente.

Em resumo, a arquitetura do seu código é classificada como cliente-servidor porque envolve uma aplicação cliente (o programa Java) que solicita e recebe dados de um servidor (o banco de dados MySQL). Essa separação de responsabilidades e a comunicação entre cliente e servidor são características fundamentais dessa arquitetura.

Design Modular – Tarefa 5

```
1 var now = new Date();
2 var hours = now.getHours();
3 var minutes = now.getMinutes();
4 var seconds = now.getSeconds();
5
6 var ampm = "am";
7 var colon = '<IMG SRC="images/colon.gif">';
8
9 if (hours == 12) {
10     ampm = "pm";
11     hours = hours - 12;
12 }
13
14 if (hours == 0) hours = 12;
15
16 if (hours < 10) hours = "0" + hours;
17 else hours = hours + '';
18
19 if (minutes < 10) minutes = "0" + minutes;
20 else minutes = minutes + '';
21
22 if (seconds < 10) seconds = "0" + seconds;
23 else seconds = seconds + '';
```



Código **Java**, com arquitetura **monolítica**, que cria uma lista com **10 perguntas** sobre as capitais de diferentes países;

A aplicação, **via console**, envia a pergunta e o usuário digita a resposta e envia para o programa. No final do quiz, são mostradas as respostas corretas, erradas e a nota do usuário (de 0 a 10). A aplicação é monolítica uma vez que qualquer alteração irá requerer recompilação de todo o código;

Reescreva o código desenvolvido na **Tarefa 4_01** e implemente a aplicação com o emprego de Arquitetura Cliente Servidor (3 Camadas).



O arquitetura do Projeto que acabamos de desenvolver pode também ser considerada como Monolítica ????

Aplicação também ser considerada Monolítica ?

Sim, a arquitetura do código que você forneceu pode ser considerada monolítica. Em uma arquitetura monolítica, componentes de software, como a interface do usuário, o código de negócios, e o acesso a dados, são combinados em um único programa a partir de um único ponto de execução. Vamos analisar o seu código sob essa perspectiva:



A aplicação também ser considerada Monolítica ?

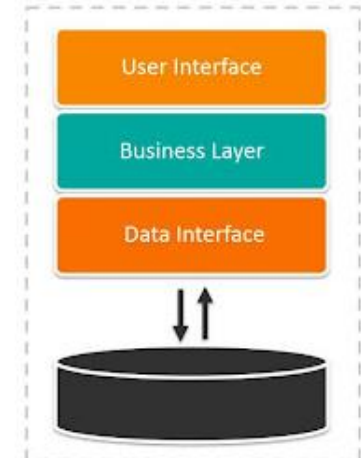
1. Código de Negócios e Acesso a Dados Juntos:

Seu código contém tanto a lógica de negócios (o quiz e o cálculo da pontuação) quanto o acesso direto ao banco de dados (executando consultas SQL e processando os resultados).

2. Execução em Um Único Processo:

Tudo é executado como parte de um único processo (um único programa Java), o que é uma característica típica de uma aplicação monolítica.

Monolithic Architecture



A aplicação também ser considerada Monolítica ?

3. Falta de Modularização:

Em arquiteturas mais modernas, como microserviços, diferentes aspectos da aplicação (como interface do usuário, lógica de negócios, e acesso a dados) seriam separados em diferentes serviços ou módulos. No seu código, esses aspectos estão todos misturados.

4. Simplicidade de Desenvolvimento e Implantação:

Um benefício de uma arquitetura monolítica é a simplicidade. É mais fácil de desenvolver, testar, implantar e executar, especialmente para aplicações menores ou mais simples, como parece ser o caso aqui.

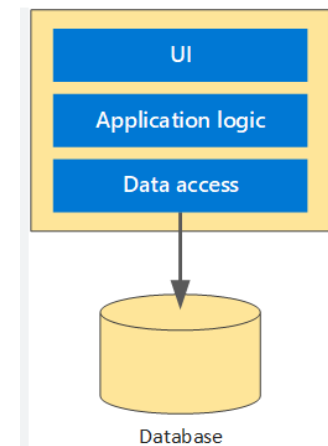


A aplicação também ser considerada Monolítica ?

5. Escalabilidade e Manutenção:

Enquanto a arquitetura monolítica é simples, ela pode se tornar difícil de manter e escalar à medida que a aplicação cresce. Cada mudança, por menor que seja, requer a construção e implantação do aplicativo inteiro.

Em resumo, sua aplicação pode ser classificada como monolítica devido à sua estrutura integrada e execução em um único processo. Esta abordagem é comum para aplicações menores e mais simples, mas pode apresentar desafios à medida que a aplicação cresce e se torna mais complexa.





Então, posso dizer que a arquitetura do Exemplo é tanto Cliente-Servidor quanto Monolítica ?

Então, posso dizer que a arquitetura do Exemplo é tanto Cliente-Servidor quanto Monolítica ?

Sim, você pode considerar que a arquitetura da sua aplicação é tanto monolítica quanto cliente-servidor. Esses dois termos descrevem aspectos diferentes de uma aplicação:

1. Monolítica:

- Este termo descreve como os componentes da aplicação são organizados e como eles interagem entre si.
- Em uma arquitetura monolítica, a lógica de negócios, a interface do usuário, e o acesso a dados estão todos integrados em uma única unidade de software.
- Isso significa que a aplicação é desenvolvida, implantada, e escalada como um todo.

Então, posso dizer que a arquitetura do Exemplo é tanto Cliente-Servidor quanto Monolítica ?

2. Cliente-Servidor:

- Este termo descreve a relação entre as diferentes partes da aplicação em termos de como elas se comunicam e interagem.
- Em uma arquitetura cliente-servidor, a aplicação é dividida em clientes e servidores, que comunicam-se uns com os outros através de uma rede.
- O servidor geralmente gerencia recursos e serviços, como acesso a um banco de dados, enquanto o cliente interage com o usuário final (por exemplo, através de uma interface gráfica).

Então, posso dizer que a arquitetura do Exemplo é tanto Cliente-Servidor quanto Monolítica ?

No seu caso, a aplicação Java que você desenvolveu é monolítica porque combina várias funcionalidades em um único software. Ao mesmo tempo, ela segue o modelo cliente-servidor porque interage com um servidor de banco de dados (MySQL, neste caso). O seu código Java age como o cliente, enviando solicitações de consulta ao servidor MySQL, que responde com os dados solicitados.

Portanto, é correto dizer que sua aplicação tem uma arquitetura monolítica implementada em um ambiente cliente-servidor.