



Buenos
Aires
Ciudad

Agencia de Habilidades
para el Futuro

<Talento
Tech/>

Node.JS

Clase N° 11: REQUEST & RESPONSE

¡Les damos la bienvenida!

 Vamos a comenzar a grabar la clase.

Índice



1



2

REQUEST & RESPONSE

- Rutas
- CORS: solicitudes entre dominios
- Error Handle (404)
- Rutas parametrizadas
- POSTMAN

Capa lógica

- Express Router
- Controladores
- Servicios
- Arquitectura basada en capas

Objetivos de la Clase

1 Comprender rutas

Estructurar navegación y interacción entre secciones de una aplicación web.

2 Implementar CORS

Configurar correctamente solicitudes entre dominios para seguridad.

3 Manejar errores

Gestionar respuestas 404 para una experiencia de usuario consistente.

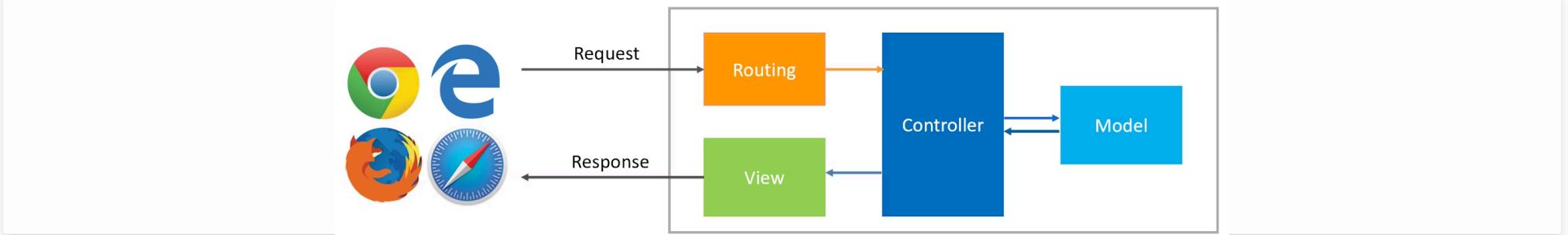
Rutas

Rutas en Aplicaciones Web

Las rutas son endpoints que definen cómo se entrega la información en una API Rest.

Funcionamiento

Cada request se filtra por la capa de routing, que apunta a un controlador específico.



Métodos HTTP en Rutas

GET POST PATCH PUT DELETE

1 GET

Para obtener información.

2 POST

Para crear nuevos recursos.

3 PUT/PATCH

Para actualizar recursos existentes.

4 DELETE

Para eliminar recursos.

CORS: Solicitudes entre dominios

CORS: Solicitudes entre Dominios

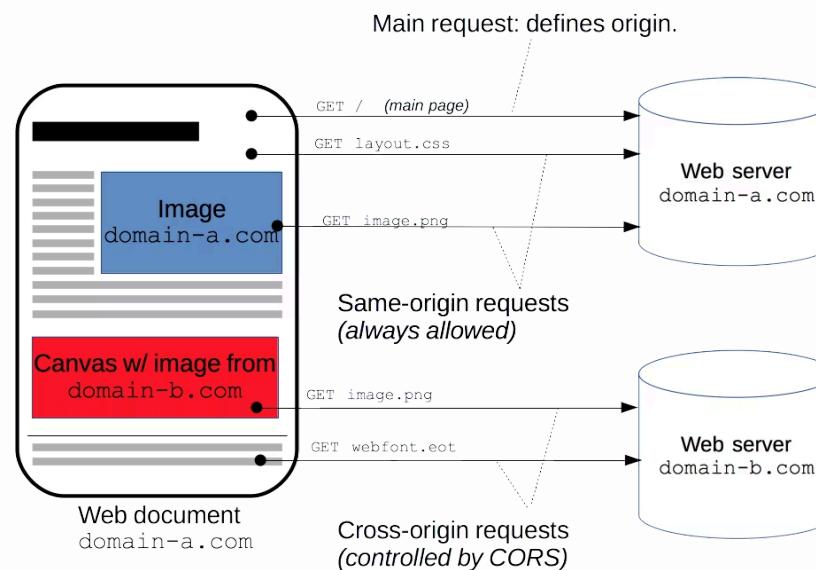
Mecanismo que permite acceso a recursos desde un dominio diferente al del documento.

Importancia

Crucial para la seguridad en aplicaciones web modernas.

Implementación

Se configura mediante headers HTTP adicionales.

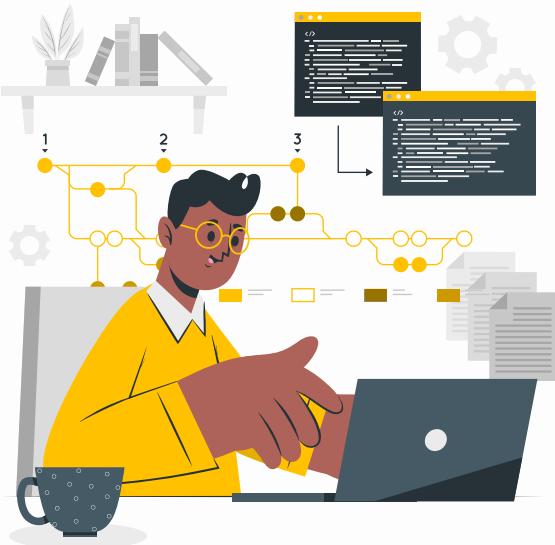


Configuración Manual de CORS

```
app.use((req, res, next) => {
  res.header('Access-Control-Allow-Origin', 'https://example.com');
  res.header('Access-Control-Allow-Methods', 'GET, POST, PUT, DELETE');
  res.header('Access-Control-Allow-Headers', 'Content-Type, Authorization');
  res.header('Access-Control-Allow-Credentials', 'true');
  next();
});
```

Error Handle (404)

Error Handle (404)



1 Propósito

Manejar solicitudes a rutas inexistentes de manera consistente.

2 Implementación

Middleware que captura solicitudes no manejadas previamente.

3 Respuesta

Envía código de estado 404 con mensaje personalizado.

Ejemplo de Error Handle (404)

```
app.use((req, res, next) => {
  res.status(404).send('Recurso no encontrado o ruta inválida');
});
```

Rutas Parametrizadas

Rutas Parametrizadas: Path Params

Permiten crear rutas dinámicas en lugar de rutas fijas.

Ejemplo

/item/:id captura cualquier valor en lugar de :id



Uso de Path Params

```
app.get('/item/:id', (req, res) => {
  const itemId = req.params.id;
  res.send(`Devolviendo el ítem con ID: ${itemId}`);
});
```

Query Params

Parámetros enviados después del signo de interrogación en la URL.

Uso

Útiles para filtros o configuraciones adicionales.

Ejemplo

/items?category=electronics&price=low

HTTP Request

<https://api.example.com/products>



page = 1
First Parameter

&

limit = 10
Second Parameter

&

sort = name
Third Parameter

Complete URL:

<https://api.example.com/products?page=1&limit=10&sort=name>

Uso de Query Params

```
app.get('/items', (req, res) => {
  const category = req.query.category;
  const price = req.query.price;
  res.send(`Categoría: ${category}, Precio: ${price}`);
});
```

Path Params vs Query Params

Path Params

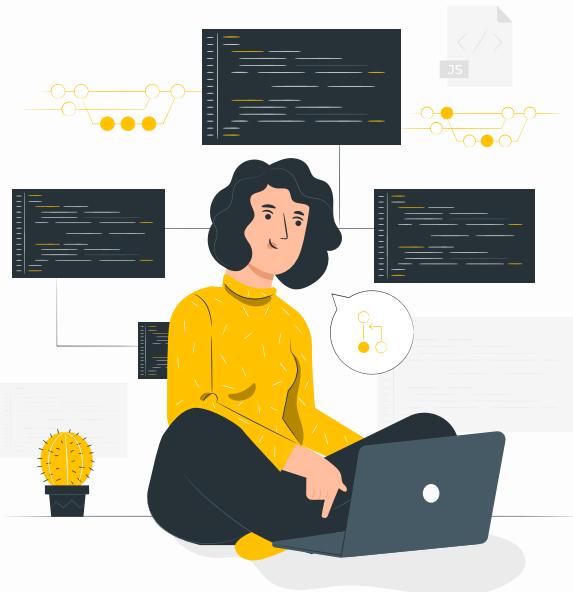
Para identificar recursos específicos.
Ejemplo: /item/:id

Query Params

Para filtros o configuraciones opcionales. Ejemplo:
/items?category=electronics

POSTMAN

POSTMAN: Cliente HTTP



Función

Enviar solicitudes HTTP y recibir respuestas.

Ventajas

Interfaz gráfica para operaciones complejas.

Usos

Probar API, manejar autenticación, trabajar con JSON.

Uso Básico de POSTMAN

Instalación

Descarga desde postman.com o usa la versión web.

Enviar y Revisar

Haz clic en "Send" y observa la respuesta.

1

2

3

Crear Solicitud

Selecciona método HTTP y ingresa URL.

Recursos Adicionales



Documentación Express

expressjs.com



Postman Learning Center

learning.postman.co



Artículo CORS

[MDN Web Docs](https://developer.mozilla.org/en-US/docs/Web/HTTP/CORS)

Explora estos recursos para profundizar en los conceptos aprendidos y mejorar tus habilidades de desarrollo web.



Ejercicio Práctico



Ejercicio Práctico

Obligatorio

Configuración de Rutas y Manejo de Parámetros

Sabrina y Matías se acercan a revisar tu progreso. Matías sonríe con aprobación: "Has avanzado muchísimo, pero ahora es momento de subir un nivel. Necesitamos que configures la capa de rutas de manera profesional."

Sabrina se cruza de brazos y asiente. "Exacto. Queremos que aprendas a organizar tus rutas, además de implementar un par de configuraciones clave como **CORS** y un manejo de errores para rutas inexistentes. Esto te ayudará a preparar tu servidor para escenarios reales."



Matías continúa: "Pero eso no es todo. También queremos que practiques cómo manejar datos dinámicos que llegan a través de las solicitudes. Deberás configurar rutas que lean tanto **path params** como **query params** y luego devolver esa información en las respuestas. Y no te olvides de usar **POSTMAN** para validar todo."

Ejercicio Práctico

Obligatorio

Misión:

1. Establece 4 rutas en el archivo de entrada de la aplicación. Asegúrate de:
 - Permitir solicitudes entre dominios configurando **CORS**.
 - Agregar un middleware que capture las rutas no encontradas y devuelva una respuesta 404.

Usa **POSTMAN** para probar las rutas que configuraste. Asegúrate de capturar correctamente los parámetros enviados y mostrar la información de manera clara en las respuestas.

Sabrina sonríe y agrega: "Recuerda, la organización es clave. Cuando logres que todo funcione, será el momento perfecto para revisar cómo cada pieza encaja en el proyecto. ¡Buena suerte!"

Tu desafío está claro. Ahora es momento de implementar lo aprendido y demostrar tus habilidades.

2. Define rutas que utilicen:

- **Path params:** Por ejemplo, una ruta que devuelva información basada en el ID de un producto o usuario.
- **Query params:** Una ruta que filtre datos con base en parámetros como categoría o precio.

