

**Package** com.mapmyjourney.backend.controller

## Class UserController

java.lang.Object<sup>↗</sup>  
com.mapmyjourney.backend.controller.UserController

```
@RestController
@RequestMapping("/users")
public class UserController
extends Object↗
```

Controlador REST para gestionar usuarios.

### Constructor Summary

Constructors
Constructor
Description
<code>UserController()</code>

### Method Summary

All Methods	Instance Methods	Concrete Methods
Modifier and Type	Method	Description
	<code>org.springframework.http.ResponseEntity&lt;Void<sup>↗</sup>&gt;</code>	
	<code>deleteUser(Long<sup>↗</sup> userId)</code>	
	7.	
	<code>org.springframework.http.ResponseEntity&lt;UserDTO&gt;</code>	
	<code>getUserByEmail(String<sup>↗</sup> email)</code>	
	4.	
	<code>org.springframework.http.ResponseEntity&lt;UserDTO&gt;</code>	
	<code>getUserById(Long<sup>↗</sup> userId)</code>	
	3.	
	<code>org.springframework.http.ResponseEntity&lt;LoginResponseDTO&gt;</code>	
	<code>login(LoginRequestDTO request)</code>	
	2.	
	<code>org.springframework.http.ResponseEntity&lt;LoginResponseDTO&gt;</code>	
	<code>registerUser(@Valid UserCreateRequestDTO request)</code>	
	1.	
	<code>org.springframework.http.ResponseEntity&lt;UserDTO&gt;</code>	
	<code>updateUser(Long<sup>↗</sup> userId, @Valid UserCreateRequestDTO request)</code>	
	5.	
	<code>org.springframework.http.ResponseEntity&lt;UserDTO&gt;</code>	
	<code>updateUserProfile(Long<sup>↗</sup> userId, UserUpdateRequestDTO request)</code>	
	6.	

Methods inherited from class java.lang.Object<sup>↗</sup>

[clone](#), [equals](#), [finalize](#), [getClass](#), [hashCode](#), [notify](#), [notifyAll](#), [toString](#), [wait](#), [wait](#), [wait](#)

## Constructor Details

### UserController

```
public UserController()
```

## Method Details

### registerUser

```
@PostMapping("/register")
@CrossOrigin(origins={"https://mapmyjourney-
4w93.onrender.com","http://localhost:4200","http://localhost:3000"})
public org.springframework.http.ResponseEntity<LoginResponseDTO> registerUser
(@Valid @RequestBody
 @Valid UserCreateRequestDTO request)
```

1. Registra un nuevo usuario. POST /api/users/register

### login

```
@PostMapping("/login")
public org.springframework.http.ResponseEntity<LoginResponseDTO> login(@RequestBody
                                                                    LoginRequestDTO request)
```

2. Inicia sesión con email y contraseña. POST /api/users/login

### getUserById

```
@GetMapping("/{userId}")
@PreAuthorize("hasRole(\"'USER'\")")
public org.springframework.http.ResponseEntity<UserDTO> getUserById(@PathVariable
                                                                    Long? userId)
```

3. Obtiene un usuario por ID. GET /api/users/{userId}

### getUserByEmail

```
@GetMapping("/email/{email}")
@PreAuthorize("hasRole(\"'USER'\")")
public org.springframework.http.ResponseEntity<UserDTO> getUserByEmail(@PathVariable
                                                                    String? email)
```

4. Obtiene un usuario por email. GET /api/users/email/{email}

### updateUser

```
@PutMapping("/{userId}")
@PreAuthorize("hasRole(\"'USER'\")")
public org.springframework.http.ResponseEntity<UserDTO> updateUser(@PathVariable
                                                                    Long? userId,
                                                                    @Valid @RequestBody
                                                                    @Valid UserCreateRequestDTO request)
```

5. Actualiza un usuario existente. PUT /api/users/{userId}

### updateUserProfile

```
@PutMapping("/{userId}/profile")
@PreAuthorize("hasRole(\"'USER'\")")
```

```
public org.springframework.http.ResponseEntity<UserDT0> updateUserProfile(@PathVariable
                                                                    Long↗ userId,
                                                                    @RequestBody
                                                                    UserUpdateRequestDT0 request)
```

6. Actualiza el perfil del usuario (versión flexible). PUT /api/users/{userId}/profile

## deleteUser

```
@DeleteMapping("/{userId}")
@PreAuthorize("hasRole(\"ADMIN\")")
public org.springframework.http.ResponseEntity<Void↗> deleteUser(@PathVariable
                                                                    Long↗ userId)
```

7. Elimina un usuario. DELETE /api/users/{userId}