TODO TIC

9

Buenas prácticas

¿Qué es refactoring?

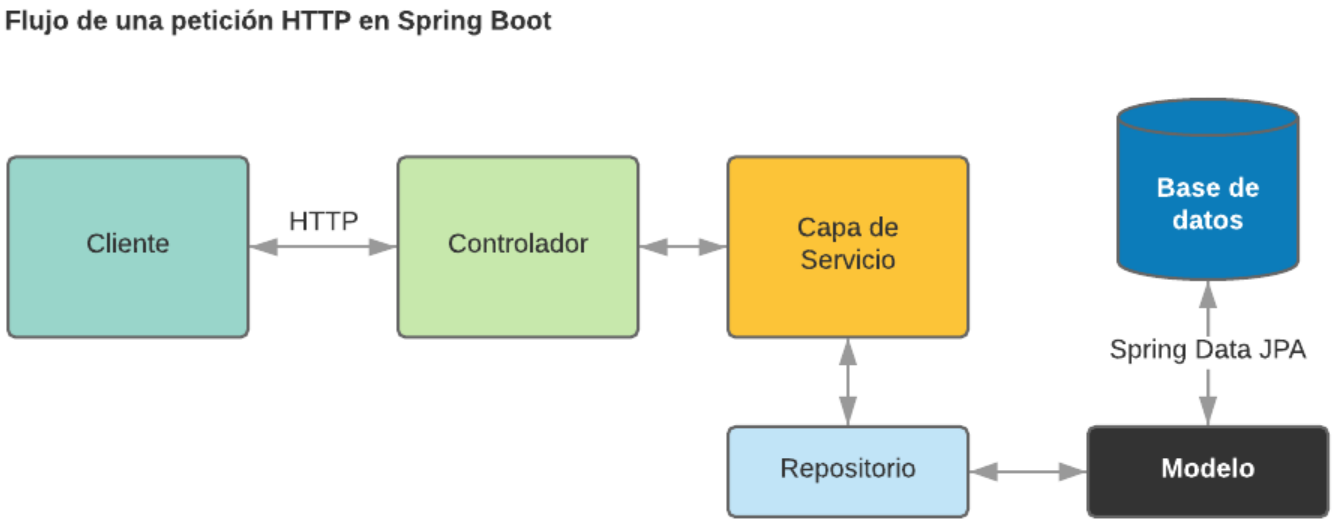
- ✓ “Es la mejora del diseño del código que ya existe” (Fowler, 1999).
- ✓ Es una técnica para reestructurar el código, alterando su estructura interna pero sin alterar el comportamiento externo (Fowler).
- ✓ Es una actividad que consiste en limpiar el código, para que sea más legible y organizado.

TODO TIC

Capa de servicio

- Es un patrón de software para aplicaciones empresariales.
- Encapsula la lógica empresarial de la aplicación, controlando las transacciones y coordinando las respuestas en la implementación de sus operaciones.
- Define el límite de una aplicación y su conjunto de operaciones disponibles desde la perspectiva de las capas de cliente interconectadas.

Capa de servicio: Spring Boot



TODO TIC

Capa de servicio

```java
@PostMapping("/users")
@PreAuthorize("hasAuthority(\"" + AuthoritiesConstants.ADMIN + "\")")
public ResponseEntity<User> createUser(@Valid @RequestBody AdminUserDTO userDTO) throws URISyntaxException {
    log.debug("REST request to save User : {}", userDTO);

    if (userDTO.getId() != null) {
        throw new BadRequestAlertException("A new user cannot already have an ID", "userManagement", "idexists");
        // Lowercase the user login before comparing with database
    } else if (userRepository.findOneByLogin(userDTO.getLogin().toLowerCase()).isPresent()) {
        throw new LoginAlreadyUsedException();
    } else if (userRepository.findOneByEmailIgnoreCase(userDTO.getEmail()).isPresent()) {
        throw new EmailAlreadyUsedException();
    } else {
        User newUser = userService.createUser(userDTO);
        mailService.sendCreationEmail(newUser);
        return ResponseEntity
            .created(new URI("/api/admin/users/" + newUser.getLogin()))
            .headers(HeaderUtil.createAlert(applicationName, "userManagement.created", newUser.getLogin()))
            .body(newUser);
    }
}
```

```java
public User createUser(AdminUserDTO userDTO) {
    User user = new User();
    user.setLogin(userDTO.getLogin().toLowerCase());
    user.setFirstName(userDTO.getFirstName());
    user.setLastName(userDTO.getLastName());
    if (userDTO.getEmail() != null) {
        user.setEmail(userDTO.getEmail().toLowerCase());
    }
    user.setImageUrl(userDTO.getImageUrl());
    if (userDTO.getLangKey() == null) {
        user.setLangKey(Constants.DEFAULT_LANGUAGE); // default language
    } else {
        user.setLangKey(userDTO.getLangKey());
    }
    String encryptedPassword = passwordEncoder.encode(RandomUtil.generatePassword());
    user.setPassword(encryptedPassword);
    user.setResetKey(RandomUtil.generateResetKey());
    user.setResetDate(Instant.now());
    user.setActivated(true);
    if (userDTO.getAuthorities() != null) {
        Set<Authority> authorities = userDTO
            .getAuthorities()
            .stream()
            .map(authorityRepository::findById)
            .filter(Optional::isPresent)
            .map(Optional::get)
            .collect(Collectors.toSet());
        user.setAuthorities(authorities);
    }
    userRepository.save(user);
    this.clearUserCaches(user);
    log.debug("Created Information for User: {}", user);
    return user;
}
```

@ConfigurationProperties

- Indica que un componente puede registrarse a uno o más perfiles activados.

```
# config ruta de almacenamiento de los archivos subidos
storage.location=uploaded-assets

app.security.jwt.access-token-validity=31104000
app.security.jwt.secret=chLhMF9w3mwDutysbQxsX8x4CGwZef4ma
```

```
@Value("${storage.location}")
private String storageLocation;
```

```
@Data
@ConfigurationProperties(prefix = "app.security.jwt")
public class JWTProperties {
    private long accessTokenValidity;
    private String secret;
}
```

@Autowired

```java
@Autowired
private CursoRepository cursoRepository;

@Autowired
private UsuarioRepository usuarioRepository;

@Autowired
private InscripcionRepository inscripcionRepository;
```

```java
private final CursoRepository cursoRepository;
private final UsuarioRepository usuarioRepository;
private final InscripcionRepository inscripcionRepository;

@Autowired
public UsuarioController(CursoRepository cursoRepository,
                         UsuarioRepository usuarioRepository,
                         InscripcionRepository inscripcionRepository) {
    this.cursoRepository = cursoRepository;
    this.usuarioRepository = usuarioRepository;
    this.inscripcionRepository = inscripcionRepository;
}
```


Conócete a ti mismo, y conocerás el universo
Oráculo de Delfos