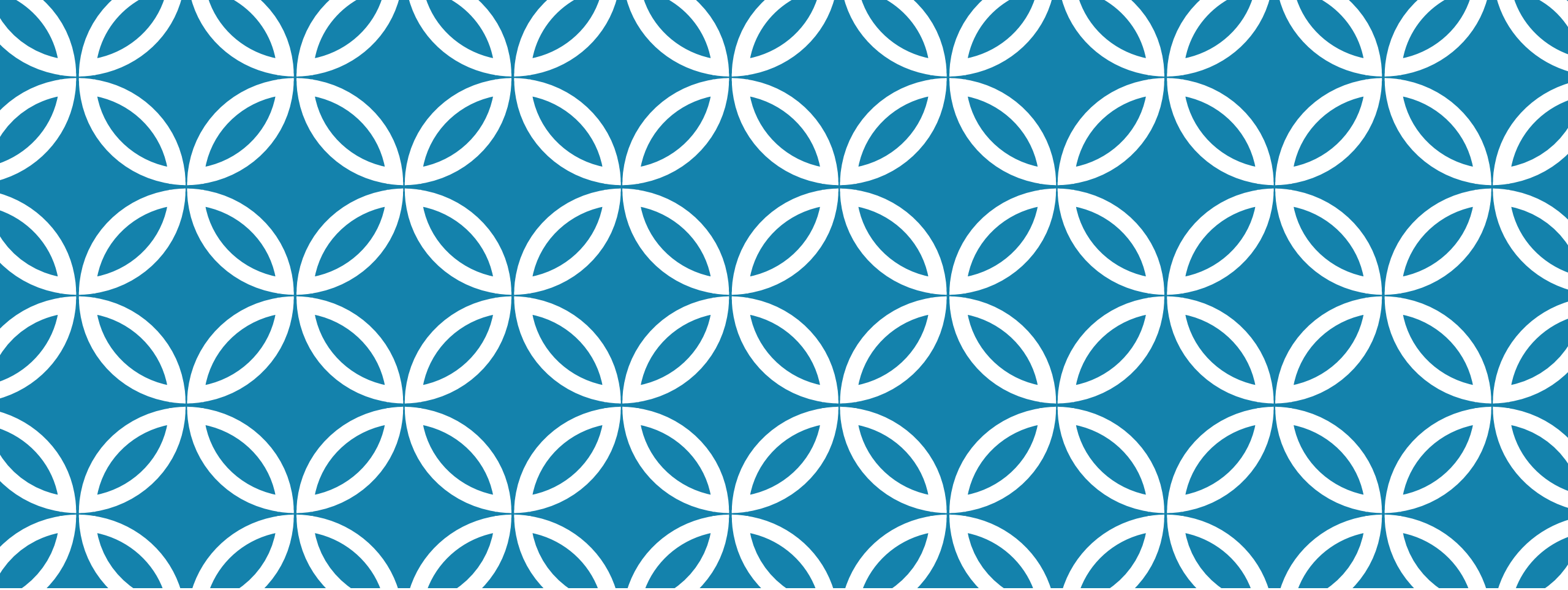
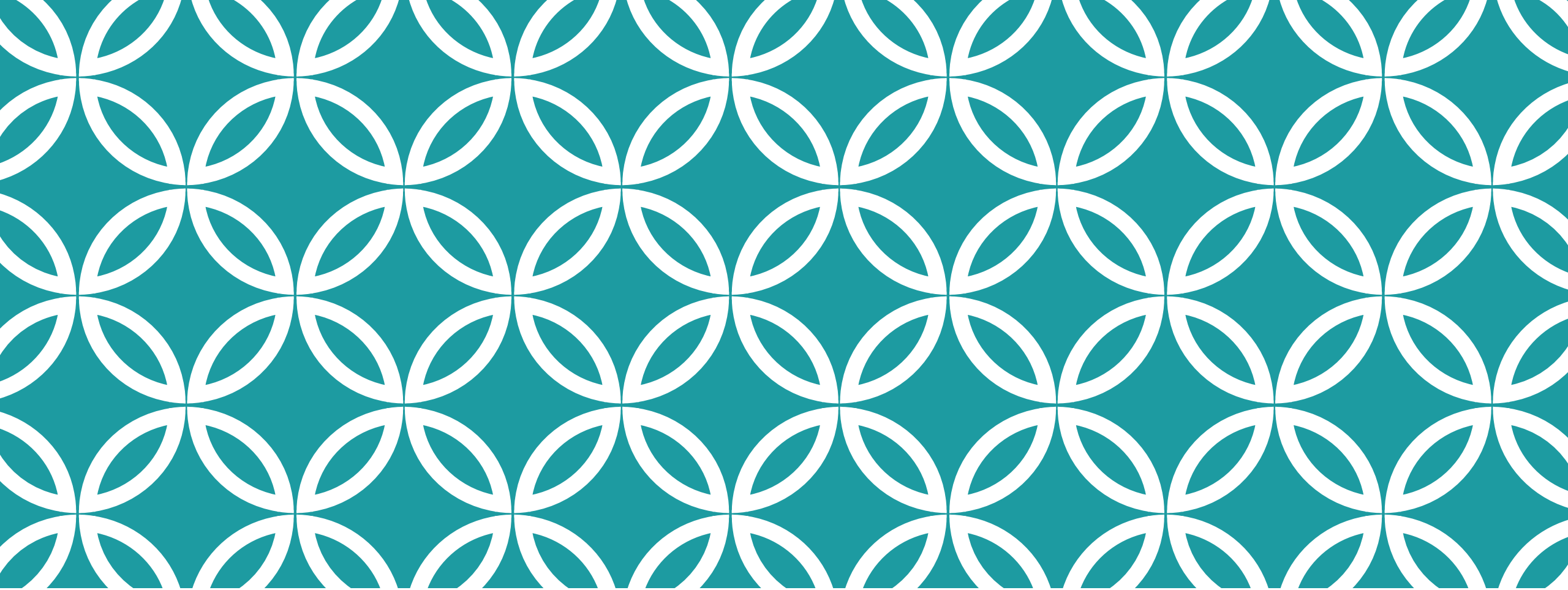


SPRING

Jérémy PERROUAULT



SPRING TEST

Introduction à
Spring Test avec JUnit

MOCK – MOCK USER

Une classe qui va simuler l'exécution d'une « vraie » classe

- Mock User va simuler une connexion d'un utilisateur
  - Sans avoir spécifiquement besoin qu'il existe dans l'application concrète
  - On pourra lui attribuer des rôles et autorisations
  - **@WithMockUser**

MOCK – MOCK MVC

On injecte WebApplicationContext

```
@Autowired
private WebApplicationContext webApplicationContext;
```

On construit mockMvc avec MockMvcBuilders, en appliquant la sécurité

```
@BeforeEach
public void before() {
   this.mockMvc = MockMvcBuilders
        .webAppContextSetup(this.webApplicationContext)
        .apply(springSecurity())
        .build();
}
```

On annote les méthodes (ou la classe) de test de **@WithMockUser**

- On peut spécifier un username, password, role, ...

MOCK – MOCK MVC

Vérifier que l'adresse "/produit" est accessible (GET)

```
this.mockMvc.perform(get("/produit"))
  .andExpect(status().isOk());
```

Vérifier qu'on peut poster un produit (avec le CSRF)

```
this.mockMvc
  .perform(
    post("/produit/ajouter")
    .with(csrf())
    .param("libelle", "test")
    .param("prix", "50")
  )
  .andExpect(status().is3xxRedirection())
  .andExpect(model().hasNoErrors())
  .andExpect(redirectedUrl("./"));
```

MOCK – MOCK MVC

Il faut ajouter une dépendance supplémentaire

- **spring-security-test**

EXERCICE

Reprendre le tests contrôleur et implémenter la sécurité