

ALOM

ARCHITECTURES LOGICIELLES ORIENTÉES MICROSERVICES

INTRODUCTION

SALUT 🖐️

MOI C'EST JULIEN

- FREELANCE SOLUTION & SOFTWARE ARCHITECT 🏗️
- TEACHER 🎓 @UNIV-LILLE
- ASSOCIATE 🟩 @EKITE
- SPEAKER 🎤 (DEVOXX - DEVFEST LILLE - SUNNY TECH)
- TEAM ☁️ @CLOUDNORD
- WRITER 📖 @ENI



I LIKE



I LIKE



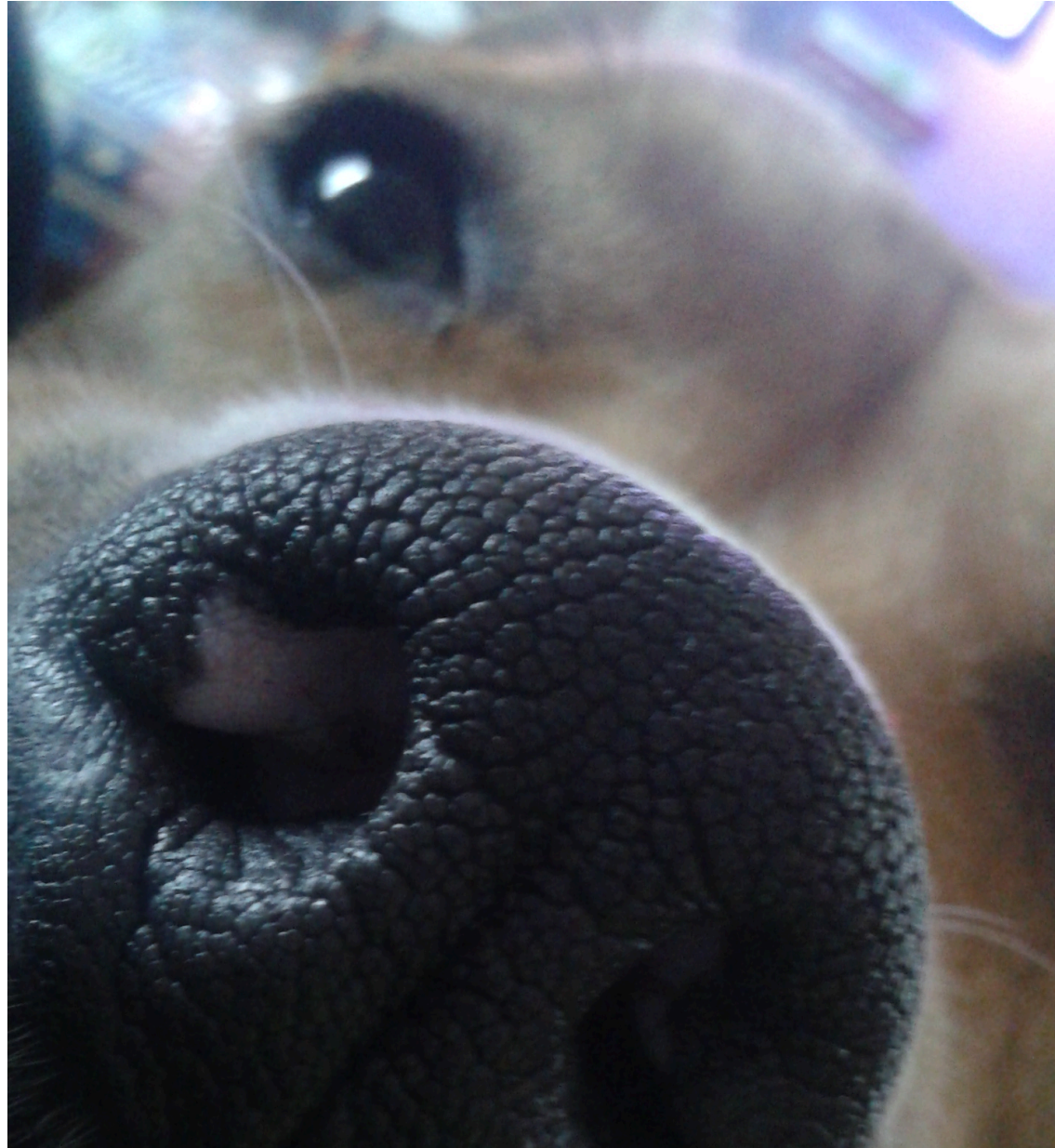
I LIKE



I LIKE



PROUD OWNER OF



I LIKE



ABOUT YOU

 iPhone vs Android 

 Cats vs Dogs 

 Coffee vs Tea 

 Java vs Node 

ALOM - OBJECTIFS

- Vision du développement Java/JEE en entreprise
- Principes fondateurs, besoins, contraintes, architectures
- Comment ça fonctionne ?
- Déploiements distribués, cloud
- Frameworks
- Outils / Middlewares

PLANNING DU COURS (PRÉVISIONNEL)

1. Introduction
2. Handcrafting everything - ❤️ servlets
3. Java 5 to 23
4. Spring Boot
5. Persistance des données
6. GUI
7. Intéropérabilité
8. Traitements asynchrones - Kafka
9. Clean Architecture - Hexa
10. Patterns avancés/Cloud/Containers...

PLANNING DU TP

1. Setup
2. Handcrafting everything - ❤️ servlets
3. Java 5 to 21
4. Spring Boot
5. Persistance des données, relationnel, NoSQL
6. Spring MVC, JSP
7. Web services
8. Intéropérabilité
9. Traitements asynchrones - Kafka
10. Clean Architecture - Hexa
11. Patterns avancés

OUTILS

- IDE : [IntelliJ IDEA](#) conseillé 😊
 - Eclipse, Netbeans, VSCode sinon
- Tomcat (Embedded)
- Maven
- Git -  GitLab
-  Java
 - Java 17 (release sept. 2021)
 - Java 21 (release sept. 2023)

RÉFÉRENCES

- Cours de CAR
- spring.io -  **spring**
by Pivotal.

NOTATION

PROJET + DÉMO



présentation le 11 décembre, rendu le mercredi 10
janvier

BESOIN DE ME CONTACTER ?



✉ julien.wittouck [at] gmail.com

✉ julien.wittouck [at] univ-lille.fr



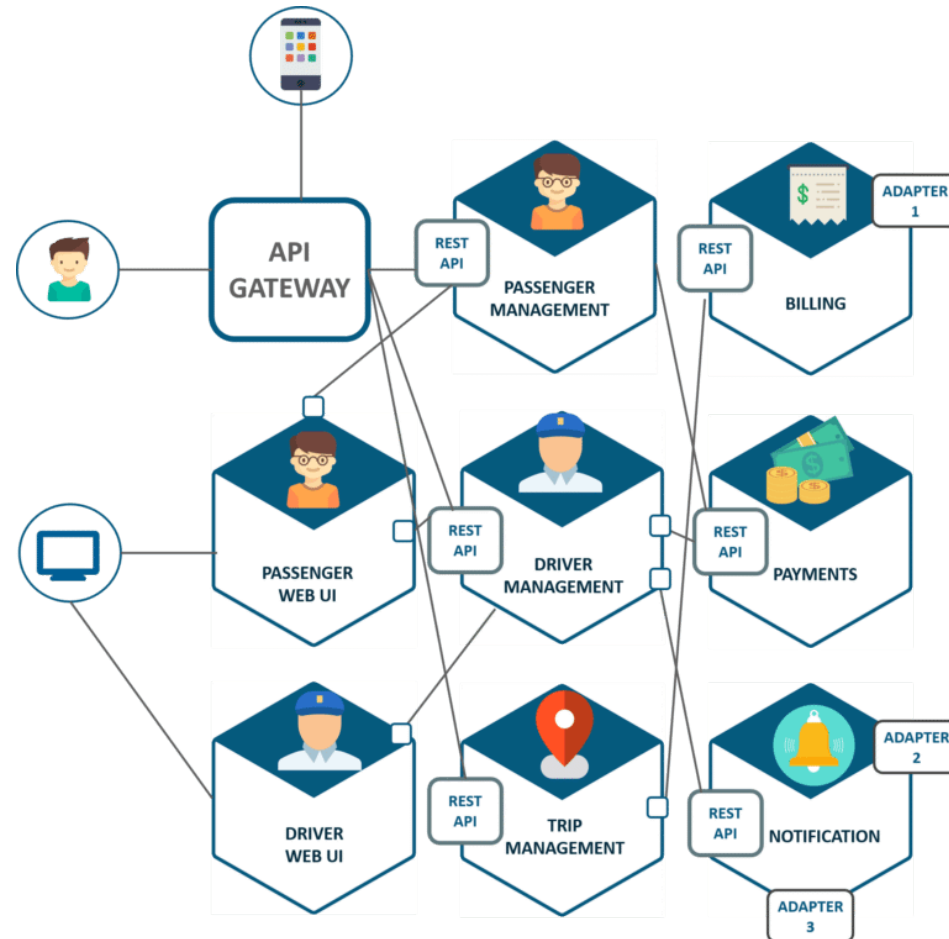
@CodeKaio











@codekaio.bsky.social



<https://alom-2024.gitlabpages.univ-lille.fr/cours/>



BESOINS ET CONTRAINTES

-  Gestion de session/persistance de données
-  Sécurité
-  Gestion d'erreur, de transactions 
-  Montée en charge (scalabilité)
 - Déploiement distribués
 - Load-balancing
-  Interopérabilité avec les partenaires
-  Intégration aux systèmes existants (legacy)
-  Compatibilité avec les produits cloud (Cloud Native)

NOTIONS DE FRAMEWORK

Fournit des services pour répondre aux besoins & contraintes.

JEE (Jakarta EE) est un framework, Spring est un framework

- Inversion de contrôle, injection de dépendances
- Configuration
- Pools d'objets, montée en charge, gestion de session
- Transactions
- Web services
- Messaging

HARDWARE / OS / MIDDLEWARE / SOFTWARE

- Hardware : Couche physique (CPU/RAM/Disque), virtualisé ou bare-metal
- OS : Système d'exploitation. Primitives d'usage de la couche physique (POSIX)
- Middleware : Environnement d'exécution d'applications, et services (Java / Tomcat)
- Framework : Support du développement
- Software : c'est nous ! ☒

ECRITURE D'APPLICATIONS

 Monolith Vs Micro-services 

MONOLITH

Application "tout-en-un"

- IHM (web)
- Services
- Un seul "package"

ECRITURE D'APPLICATIONS

 Monolith Vs Micro-services 

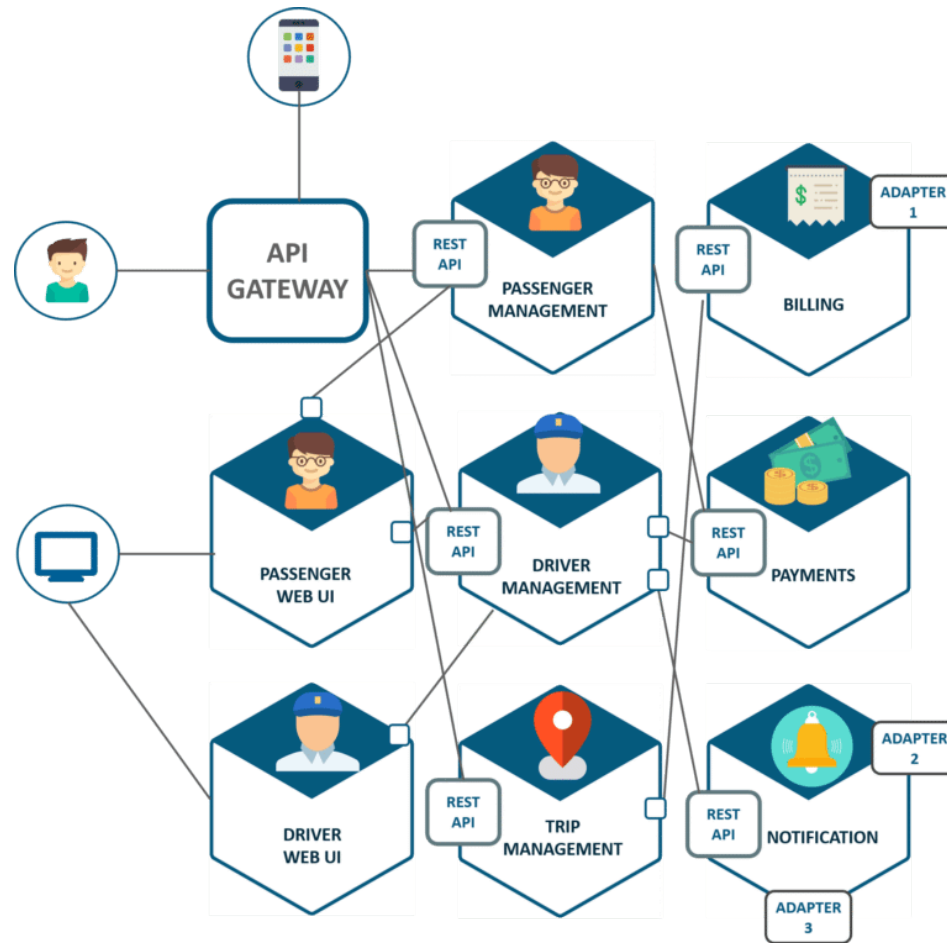
MICRO-SERVICES

- IHM
- Services métiers découpés
- Plusieurs "packages"

ECRITURE D'APPLICATIONS

- Développement de composant séparés
 - Composants métiers : EJB/Services
 - Composants IHM : JSP/Web
 - Web services
- Construction d'une application par assemblage de composants
 - Composants métier/IHM
 - Services middleware

ASSEMBLAGE D'APPLICATION



TP - SETUP DE VOS POSTES !



FIN DU COURS