

# STD – BACK

.NETCORE | ENTITY FRAMEWORK

# Sommaire

1. Technologie
2. Cycle de Vie
3. Développement

**Annexe – Base Technique**

# 01

# Technologie

# Vue d'ensemble

## Technologie

### Base

- **Packet Mngt** : nuget
- **Langage** : C#, SQL

### Cycle de Vie

- **Dev** : Visual Studio Code
- **Unit Test** : ?
- **System Test** : ?
- **Code Quality** : ?
- **Host** :

### Architecture

- **Framework** : .NetCore
- **FRT / BCK** : HTTP | REST
- **ORM** : Entity Framework (LINQ)
- **Database**

### Example

- **Dev** : voir ici

# Library / Package (See « *webapi.csproj* »)

## Technologie

### .NetCore

- @angular/common => http

### Autre

- automapper
- mailKit
- jwtBearer
- entityFrameworkCore
- (DEV Only)

### Autre

- PDF Generation (CrystalReport / Power PDF)
- TODO

### Légende

- Intégré package
- Majeur package
- A creuser package

Designed By - K

# 02

## Cycle de Vie

(\* de l'Application)

- **Environnement de Dev**
- **Pipeline**
- **Déploiement**
  - Avec VS Code
  - Publier “dotnet publish -c Release -o ./publish” (génère ds dossier /publish)
  - Click Droit sur le dossier “/publish” puis “Deploy to web App”
- **Cookie Policy**
  - <https://docs.microsoft.com/en-us/aspnet/core/security/samesite?view=aspnetcore-6.0>
  - **Use : SameSite=None + Secure=True**
    - Parce que le BACK & Front sont hébergés sur des Env different (Azure / Firebase)
    - => Impose l'utilisation de HTTPS

# Configuration

## Cycle de Vie

### Fonctionnalité

- **Permettre une configuration selon un environnement**
- **Élément Configurable**
  - URL Service Business
  - URL Service Tiers (Map, GED, eSign...)
  - Gestion des Fichiers
  - Gestion des Timers
- **Ou stocker la configuration ?**
  - Dans un fichier du code source
  - BD\_Applicative
  - WS\_Dédié

### Mail SMTP (Sending Blue)

- **"SmtpHost"**: "smtp-relay.sendinblue.com",
- **"SmtpPort"**: 587,
- **"SmtpUser"**: "kevin.gellenoncourt@gmail.com",
- **"SmtpPass"**: "QwKNySB3Tt6bxD8A"



- **Tester c'est Douter ;p**
- **Tests Unitaires**
  - Test des méthodes « publiques » & « de plus d'une ligne »
  - Pas de test des librairies tierces
  - Couverture : XX%

# 03

## Développement

# Convention de Nommage

## Développement

- **Dossier / Fichier**

- Minuscule + Séparateur = « - »
- Nom : <nom-fichier>.<composant>.ts (Ex: )

- **Module**

- Alias : Préfixé par « @ »

- **Classe / Object / Type / Variable**

- Casse : CamelCase (maVariable, monObjet)
- Private -> Préfixé par « \_ » (\_maVar)

### Annotation

- **TODO** : Code à re-travailler
- **TO\_CONF** : Codes à adapter selon le besoin

### ALIAS

- @env, @enum, @material, @timer, @token
- @core, @account, @shoppingList, @product
- @package, @log, @deploy, @style, @conf

- **Interface** (JSON, WSDL ?)
- **Organisation du Code** (Règle métier / Appel Technique)
- **Exception** (Translator qui écoute toutes les couches)
- **Transaction**
  - **Utiliser “Using”**
  - Annotation @transaction
  - Gérée par le framework ?

# Merci !

### SQL / LINQ

- Database, table, requête, donnée, index
- Select, From Where, Join OrderBy

?

- Bonjour

### C#

- Variable, type, enum
- Class, propriété & fonction
- Interface & Héritage
- Public / privée
- Modifiers (readonly, optionnal...)
- Module
- Decorator (~Annotation de classe)
- Réflexion / Introspection
- Multi-threading