



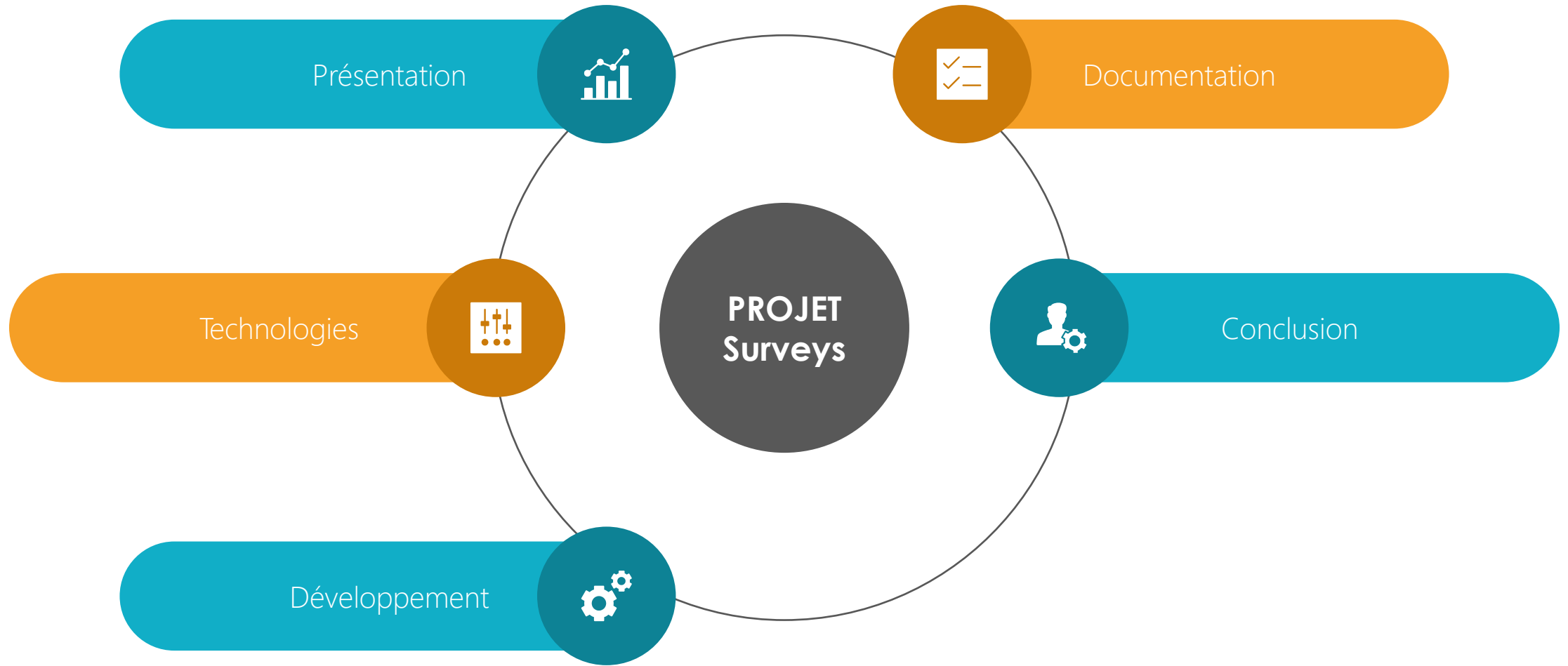
Production logicielle

Présentation

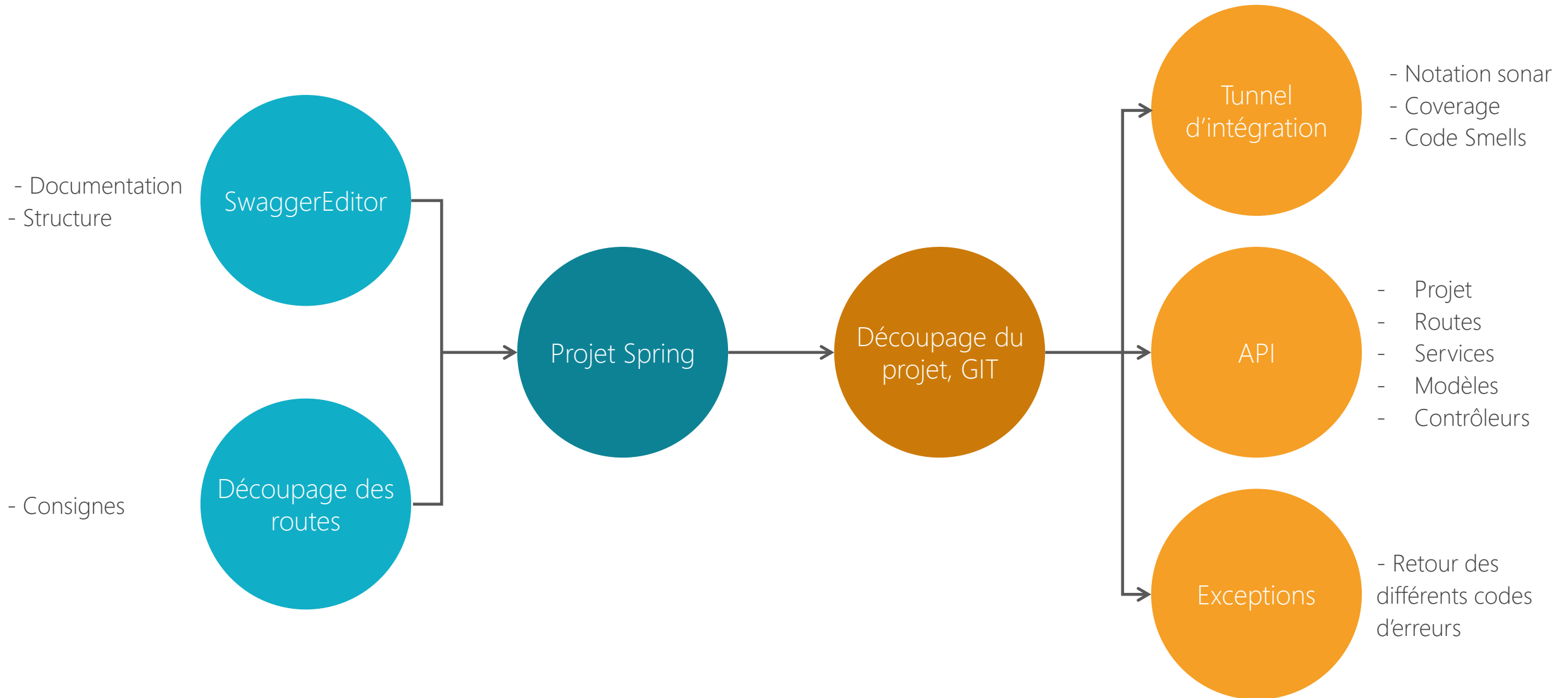
Réalisé par :

- Conrad Alexandre
- Liam Davies
- Chevalet Clément
- Bir Vincent

Sommaire du projet



Présentation



Présentation



Utilisateur

Utilisation de l'API
Swagger Editor.



Controller

Controller qui permet
de récupérer la
commande et la
rediriger vers le bon
service.



Services

Classe qui permet de
gérer toutes les
requêtes SQL.



Repository

Accéder aux données.



DataBase

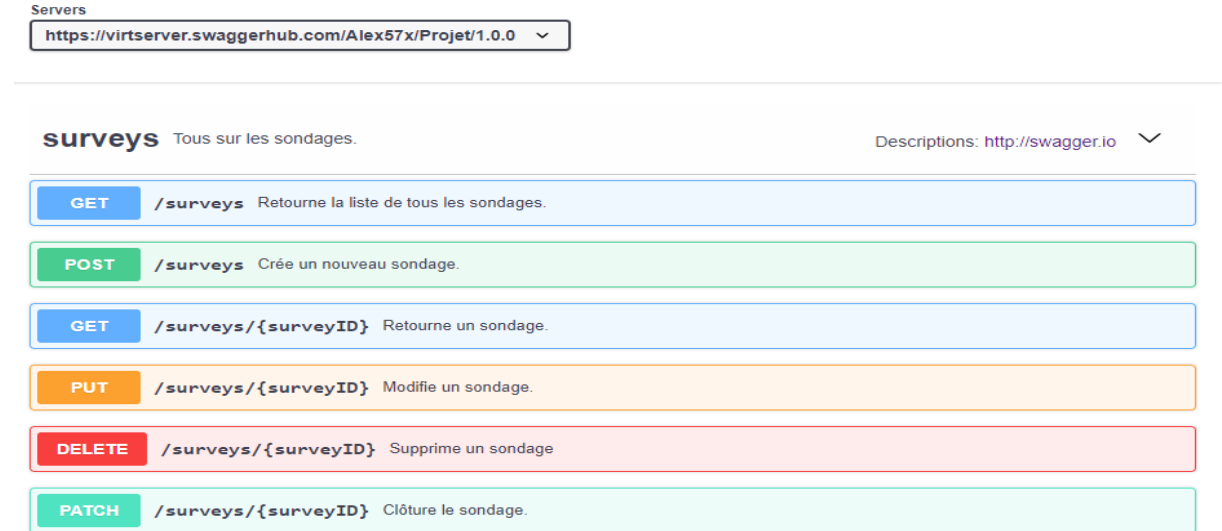
H2



Technologies



Swagger Editor



POSITIF

POINTS FORTS

- › Oblige à réfléchir clairement au découpage du cahier des charges.
- › Permet la construction de la structure du projet.
- › API et code d'erreur déjà codés.

NÉGATIF

POINTS FAIBLES

- › Code dispose de nombreux problèmes. Par exemple : Code smells, duplication et différentes vulnérabilités.
- › Problème lors de la création de certaines classes.

Technologie du projet



GitLab

The screenshot shows the GitLab project page for 'surveys'. At the top, there's a project header with the name 'surveys', a small icon, and a dropdown menu. Below this, it says 'Identifiant de projet : 9'. The main section displays project statistics: 103 commits, 5 branches, 2 étiquettes, 4,9 Mo fichiers, and 422,4 Mo Storage. A row of status indicators follows: 'pipeline passed', 'quality gate passed', 'reliability A', 'maintainability A', 'coverage 83.6%', 'Chat Slack', and 'Contributeurs 4'. Below these are more indicators: 'Le meilleur projet ? Oui, évidemment', 'code smells 7', and 'duplicated lines 0%'. A progress bar is shown below the statistics. The next section shows the current branch 'master' and a dropdown menu. To the right are buttons for 'Historique', 'Rechercher un fichier', 'EDI Web', a download icon, and a 'Clone' button. Below this is a merge commit summary: 'Merge branch 'alexandre_conrad' into 'master'', with a green checkmark and the commit hash '3a7914a0'. At the bottom, there are links for 'README', 'Configuration de l'intégration et de la livraison continues', and 'Aucune licence. Tous droits réservés'. A dashed box contains the text 'Ajouter une grappe de serveurs Kubernetes'.

POSITIF

POINTS FORTS

- › Permet un partage du code plus facilement.
- › Permet le développement dans nos branches de différentes fonctionnalités.
- › Historisation du code.

NÉGATIF

POINT FAIBLE

- › Lors d'un problème de merge.

Technologie du projet



GitLab - ReadMe

README.md

Surveys

Badges :



Projet de production logiciel réalisé par Monsieur Cédric Moschetta pour les Master 2 Génie informatique 2020/2021. L'objectif du projet est de réaliser une API pour des sondages en ligne. Le programme doit respecter certaines spécificités techniques comme un tunnel d'intégration, Spring Boot, etc.

Fabriqué avec

Liste les programmes/logiciels/ressources que nous avons utilisé pour développer notre projet :

- [Swagger Editor](#) - Création de l'API
- [Spring Boot](#) - Framework
- [H2](#) - Base de données
- [Hibernate](#) - Framework pour le mapping
- [Lombok](#) - Générateur de code
- [Maven](#) - Pour les dépendances
- [Maven Docker](#) - Pour la gestion de maven dans le pipeline
- [Java 11](#) - Version du JAVA
- [Git](#) - Projet Git
- [Jacoco](#) - Coverage
- [Jacoco](#) - Maven jacoco

Pipeline d'intégration avec :

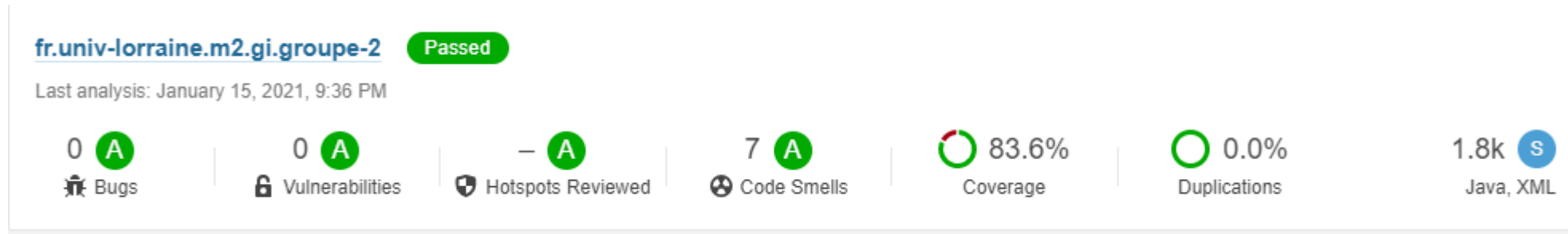
- [Gitea](#) - Build / Compile / Package
- [SonarQube](#) - Notation du code

Documentation

- [Swagger Editor](#) - Document réalisé en Swagger
- [Carnet de bord](#) - Carnet de bord du projet
- [Documentation fonctionnelle](#) - Documentation fonctionnelle
- [Documentation Technique](#) - Documentation technique
- [Rapport Test XML Jacoco](#) - Rapport des test sous le format xml
- [Rapport Test Image Jacoco](#) - Rapport des test sous le format image
- [Postman](#) - Document Postman
- [BDD](#) - Schéma de la base de données
- [BDD_verbose](#) - Verbose de la base de données

Versions

Technologie du projet



POSITIF

POINTS FORTS

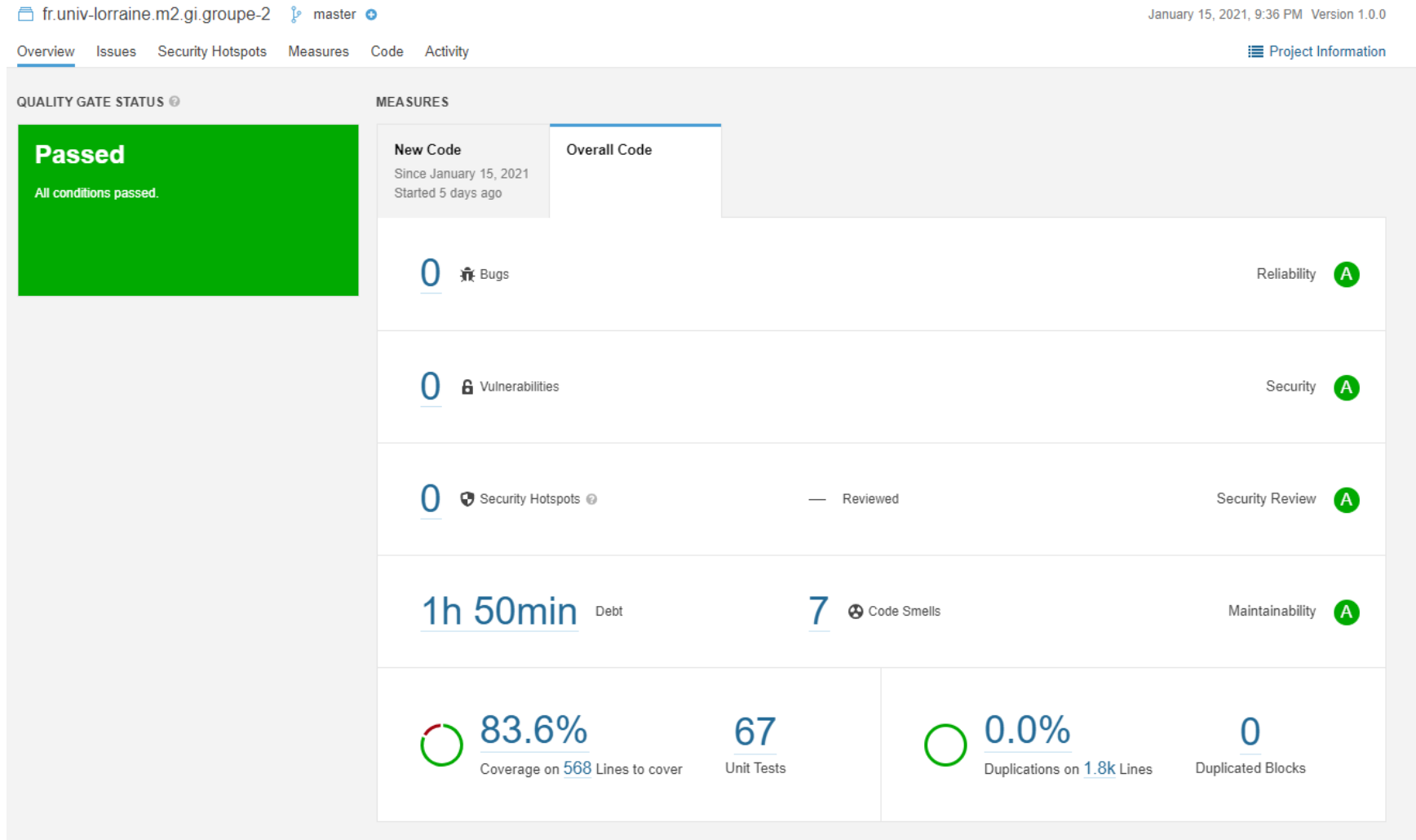
- › Permet la notation de notre code.
- › Permet de voir le pourcentage de coverage.
- › Le code smells et son apprentissage.
- › Création des badges.

NÉGATIF

POINTS FAIBLES


- › Détection de code smells inutile.
- › Manques d'informations pour la configuration (Tunnel d'intégration / Jacoco).

Technologie du projet



Technologie du projet



 Alexandre CONRAD (Alexandre)

Pipeline #369 has passed in 00:47


Branch

master

Commit

Merge branch 'alexandre_conrad' into 'master'

surveys | 15 janv.

 Alexandre CONRAD (Alexandre)

Pipeline #370 has passed in 00:56

Tag

1.0.1

Commit

Merge branch 'alexandre_conrad' into 'master'

surveys | 15 janv.

POSITIF

POINTS FORTS

- › Permet une meilleure communication.
- › Informations en cas de changements.
- › Informations sur le tunnel d'intégration.

NÉGATIF

POINT FAIBLE

- › Un peut trop de messages.



Développement

Développement du projet



```
DROP TABLE IF EXISTS "VOTE";
DROP TABLE IF EXISTS "CHOICE";
DROP TABLE IF EXISTS "COMMENT";
DROP TABLE IF EXISTS "OPTION";
DROP TABLE IF EXISTS "SURVEY";

CREATE TABLE `SURVEY` ( `ID_SURVEY` int(11) NOT NULL AUTO_INCREMENT, `NAME` varchar(65) NOT NULL, `DESCRIPTION` text NOT NULL, `END_DATE` datetime NOT NULL, `IS_AVAILABLE` boolean NOT NULL) ENGINE=InnoDB;
INSERT INTO `SURVEY` (`ID_SURVEY`, `NAME`, `DESCRIPTION`, `END_DATE`, `IS_AVAILABLE`) VALUES (1, 'Anniversaire surprise pour', 'Je serai là !', '2021-01-31 12:00:00', 1), (2, 'Anniversaire surprise pour', 'Je serai là !', '2020-12-25 12:00:00', 1);

CREATE TABLE `CHOICE` ( `ID_CHOICE` int(11) NOT NULL AUTO_INCREMENT PRIMARY KEY, `DATE` datetime NOT NULL, `ID_SURVEY` int(11) NOT NULL) ENGINE=InnoDB;
INSERT INTO `CHOICE` (`ID_CHOICE`, `DATE`, `ID_SURVEY`) VALUES (1, '2021-01-31 12:00:00', 1), (2, '2020-12-25 12:00:00', 1);

CREATE TABLE `COMMENT` ( `ID_COMMENT` int(11) NOT NULL AUTO_INCREMENT, `COMMENT` text NOT NULL, `AUTHOR` varchar(65) NOT NULL, `ID_SURVEY` int(11) NOT NULL) ENGINE=InnoDB;
INSERT INTO `COMMENT` (`ID_COMMENT`, `COMMENT`, `AUTHOR`, `ID_SURVEY`) VALUES (1, 'Je serai là !', 'Théo', 1), (2, 'Je serai là !', 'Théo', 1);

CREATE TABLE `OPTION` ( `ID_OPTION` int(11) NOT NULL AUTO_INCREMENT, `NAME` varchar(65) NOT NULL) ENGINE=InnoDB;
INSERT INTO `OPTION` (`ID_OPTION`, `NAME`) VALUES (1, 'Disponible'), (2, 'Indisponible'), (3, 'Peut-être');

CREATE TABLE `VOTE` ( `ID_VOTE` int(11) NOT NULL AUTO_INCREMENT, `AUTHOR` varchar(65) NOT NULL, `ID_CHOICE` int(11) NOT NULL, `ID_OPTION` int(11) NOT NULL) ENGINE=InnoDB;
INSERT INTO `VOTE` (`ID_VOTE`, `AUTHOR`, `ID_CHOICE`, `ID_OPTION`) VALUES (1, 'Théo', 1, 1), (2, 'Théo', 2, 2), (3, 'Alex', 3, 3);

COMMIT;
```

POSITIF

POINTS FORTS

- › Permet de créer une base de données dans la RAM ou en document physique.
- › Démarre dans notre application.

NÉGATIF

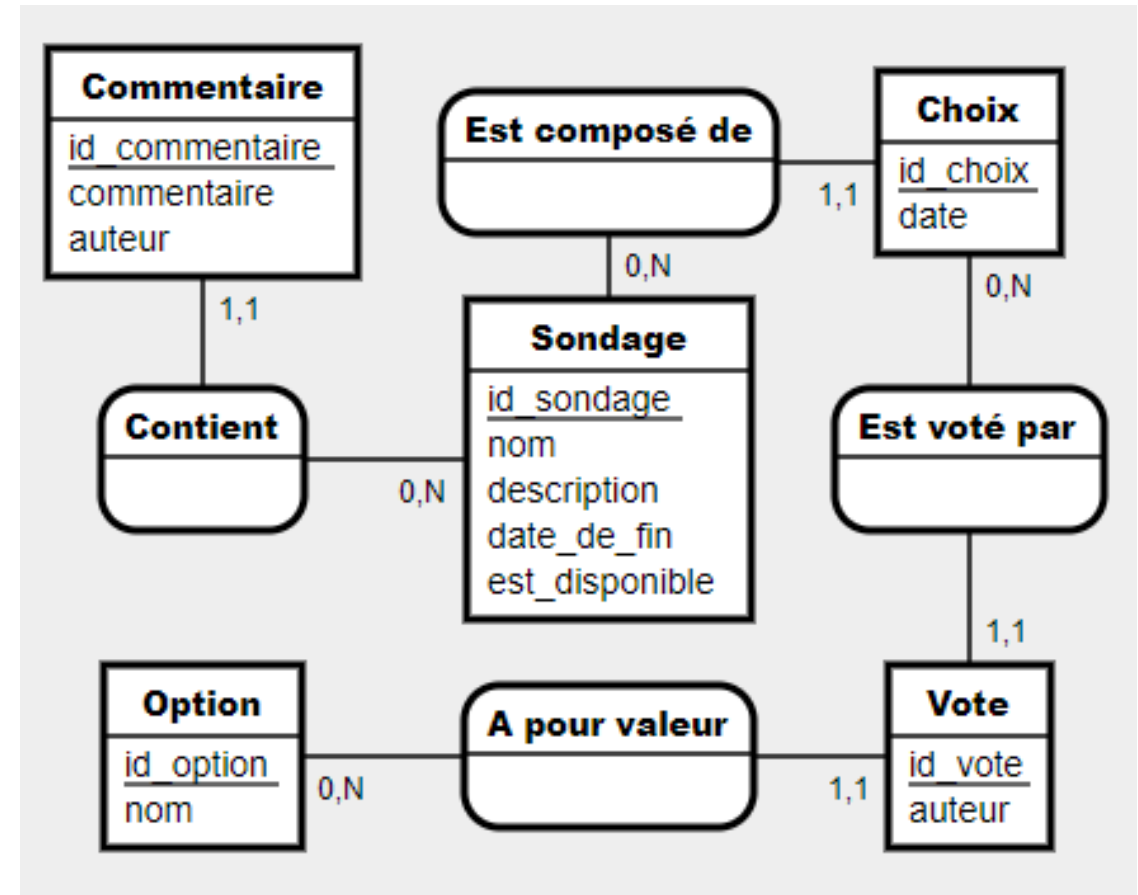
POINT FAIBLE

- › Problème avec le format de l'import.sql

Développement du projet



› Schéma de la base de données.



Développement du projet



```
/** Hibernate */
@Entity
@JsonIgnoreProperties({"hibernateLazyInitializer", "handler"})
public class Survey implements Serializable {

    @JsonProperty("idSurvey")
    @ApiModelProperty(hidden = true)
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Column(name = "ID_SURVEY")
    Long idSurvey;

    @JsonProperty("name")
    @NotNull
    @Column(name = "NAME")
    String name;

    @JsonProperty("description")
    @NotNull
    @Column(name = "DESCRIPTION")
    String description;
}
```

POSITIF

POINTS FORTS

- › Permet le mappage de la base de données avec nos classes.
- › Facile à mettre en place.

NÉGATIF

POINT FAIBLE

- › Ne pas confondre avec JavaPersistence.

Développement du projet



```
/**
 * Lombok
 */
@Validated
//@javax.annotation.Generated(value = "io.swagger.codegen.languages.SpringCodegen", date = "2020-10-31T12:55:18.203Z")
@FieldDefaults(level = AccessLevel.PRIVATE)
@RequiredArgsConstructor // Constructeur par défaut impossible
@AllArgsConstructor
@NoArgsConstructor
@Data // annotation is the combination of @ToString, @EqualsAndHashCode, @Getter and @Setter.

/** Hibernate*/
@Entity
@JsonIgnoreProperties({"hibernateLazyInitializer", "handler"})
public class Survey implements Serializable {
```

POSITIF

POINTS FORTS

- › Permet de gagner du temps et des lignes de code.
- › Permet la création de nombreux accesseurs, hashCode, toString, constructeur, etc.
- › @Data

NÉGATIF

POINTS FAIBLES

Développement du projet



```
CriteriaBuilder builder = session.getCriteriaBuilder();
CriteriaQuery<Survey> criteria = builder.createQuery(Survey.class); //Récupération de tous les sondages
criteria.from(Survey.class);
List<Survey> surveys = session.createQuery(criteria).getResultList();
transaction.commit();
surveys.size(); // => survey not found
log.info("Fonction getSurveys => OK");
return surveys;
```

POSITIF

POINTS FORTS

- › Permet le changement de base de données.
- › Requêtes réalisées en JAVA.

NÉGATIF

POINT FAIBLE

- › Documentations pas toujours évidentes.

Technologie du projet



```
@ApiOperation(value = "Supprime un sondage", nickname = "deleteSurvey", notes = "Supprime définitivement un sondage.", response = Survey.class, tags = {"s"  
@ApiResponses(value = {  
    @ApiResponse(code = 200, message = "opération réussie", response = Survey.class),  
    @ApiResponse(code = 404, message = "Ressource introuvable"),  
    @ApiResponse(code = 500, message = "Echec de connexion à la base de données.")})  
@RequestMapping(value = "/surveys/{surveyID}",  
    produces = {"application/json"},  
    method = RequestMethod.DELETE)  
ResponseEntity<Survey> deleteSurvey(@ApiParam(value = "Identifiant du sondage à supprimer.", required = true) @PathVariable("surveyID") Long surveyID);
```

POSITIF

POINTS FORTS

- › Création d'une structure.
- › Code d'erreur déjà pré construit.
- › Modifications faciles.

NÉGATIF

POINT FAIBLE

- › Format sur la documentation.

Développement du projet



```
/**
 * Fonction getSurveysIsActive
 * Doit retourner les surveys Actifs.
 */
@Test
public void getSurveysIsActive() throws Exception {
    List<Survey> surveysTest = SurveyService.getSurveysIsActive();
    for (Survey s: surveysTest)
        Assert.assertTrue(s.getIsAvailable());
}
```

POSITIF

POINTS FORTS

- › Permet le coverage du code.
- › Vérifications des changements du code.

NÉGATIF

POINT FAIBLE

- › Gestions des Mocks pas simple.



Documentation

Documentation du projet



Toute la documentation sur le code.

Documentation

- [Swagger Editor](#) - Document réalisé en Swagger
- [Carnet de bord](#) - Carnet de bord du projet
- [Documentation fonctionnelle](#) - Documentation fonctionnelle
- [Documentation Technique](#) - Documentation technique
- [Rapport Test XML Jacoco](#) - Rapport des test sous le format xml
- [Rapport Test Image Jacoco](#) - Rapport des test sous le format image
- [Postman](#) - Document Postman
- [BDD](#) - Schéma de la base de données
- [BDD_verbose](#) - Verbose de la base de données

Versions

Dernière version stable : 1.0.0

Liste des versions : [Cliquer pour afficher](#)

Documentation du projet



SwaggerEditor

Response Body

```
[
  {
    "idSurvey": 1,
    "name": "Anniversaire surprise pour Alexandre ?",
    "description": "On fait une surprise, ne lui dites pas !!",
    "isAvailable": true,
    "endDate": "2020-12-31T11:00:00.000Z"
  },
  {
    "idSurvey": 2,
    "name": "Projet X après le déconfinement ?",
    "description": "Ça va être mortel !!",
    "isAvailable": false,
    "endDate": "2020-12-31T11:00:00.000Z"
  },
  {
    "idSurvey": 3,
    "name": "Soirée Netflix & Chill ?",
    "description": "On regardera Star Wars :)",
    "isAvailable": true,
```

Response Code

200

GET

/Alex57x/Projet/1.0.0/surveys

Retourne la liste de tous les sondages.

Implementation Notes

Retourne la liste de tous les sondages en cours ou qui ont eu lieu depuis la base de données.

Response Class (Status 200)

Ressource trouvé

Model

Example Value

```
[
  {
    "description": "string",
    "endDate": {
      "date": 0,
      "day": 0,
      "hours": 0,
      "minutes": 0,
      "month": 0,
      "nanos": 0,
      "seconds": 0
    }
  }
]
```

Response Content Type

application/json

Response Messages

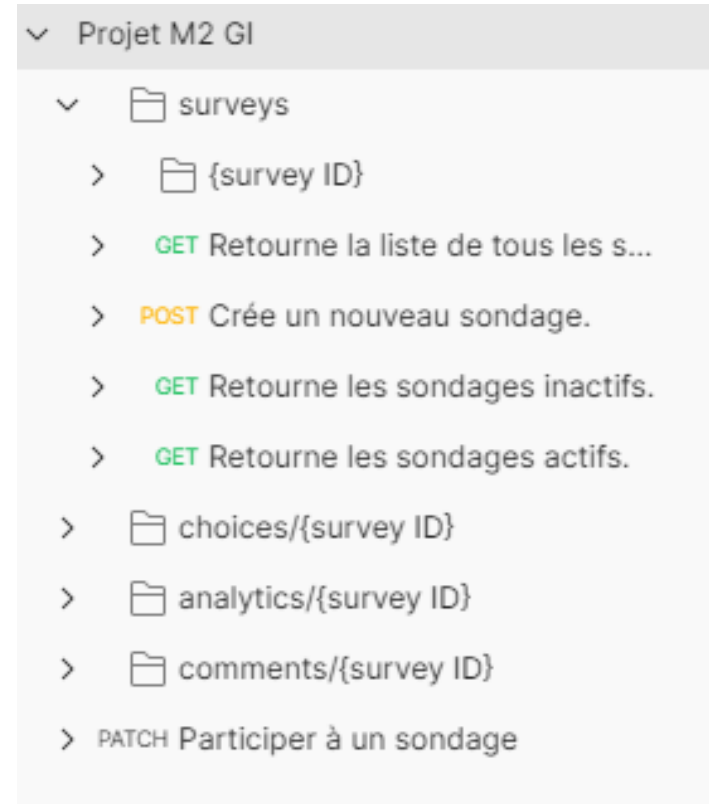
| HTTP Status Code | Reason | Response Model | Headers |
|------------------|------------------------------------------|----------------|---------|
| 401 | Unauthorized | | |
| 403 | Forbidden | | |
| 404 | Ressource introuvable. | | |
| 500 | Echec de connexion à la base de données. | | |

Try it out!

Documentation du projet

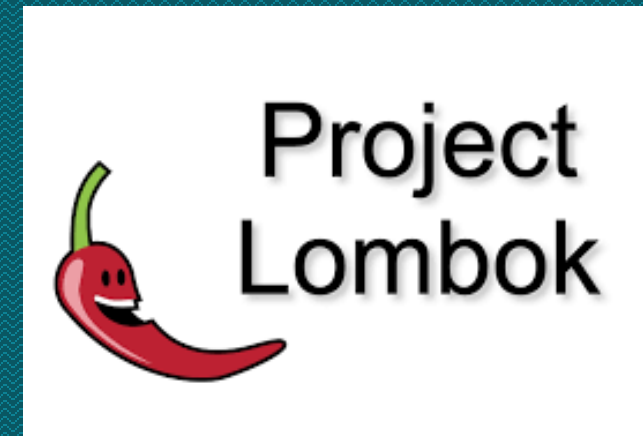


**Toute la documentation sur le code.
Identique à SwaggerEditor.**





Conclusion





Merci