|  |  |  |
| --- | --- | --- |
| |  |  | | --- | --- | | **Informaticien/-ne CFC**  Travail pratique individuel 2023 (TPI) |  | |
|  | |

Modèle de rapport v1.4

**Candidat No**

**145341**

Nom du candidat : Killian Pasche

Attention : Dernier délai de remise du rapport durant la réalisation du TPI le 18.06.2021 à 17h00 hormis la présentation (ce texte doit être effacé).

**Futnet Single Master**

### Sommaire

Résumé du rapport du TPI 2

1 Les grandes lignes du projet 3

1.1 Analyse de la situation initiale 3

1.2 Analyse de l’état désiré 3

1.3 Cahier des charges / exigences du système 3

1.4 Organisation du projet 3

2 Analyse préliminaire 4

2.1 Test technologique 4

2.2 Objectifs du système 4

2.3 Variantes 4

2.4 Choix de variante 5

2.5 Rentabilité 5

2.6 Analyse de risque 5

2.7 Sécurité de l’information et protection des données 5

3 Concept 6

3.1 Exigences du système 6

3.2 Architecture du système 6

3.3 Plan d’intégration des systèmes 6

3.4 Concept d‘implémentation 6

3.5 Concept de formation 6

3.6 Concept de tests 6

3.7 Moyens nécessaires 6

4 Réalisation 7

4.1 Spécifications détaillées 7

4.2 Design du système 7

4.3 Configuration xyz 7

5 Test 8

5.1 Procédure de test 8

5.2 Protocol de test 8

5.3 Signature du protocole de test 8

6 Conclusion 9

6.1 Améliorations possibles 9

6.2 Auto-évaluation 9

7 Bibliographie: liste des sources et références 10

8 Glossaire 11

9 Signatures 12

10 Annexes 13

# Résumé du rapport du TPI

Le Résumé du rapport du TPI est une présentation conceptuelle du travail effectué et du résultat attendu d’au maximum une page A4 qui permettent au lecteur une rapide compréhension du rapport de travail. . Il contient trois paragraphes : Situation de départ, mise en œuvre, résultats.

Pour plus de détails, veuillez vous référez au Manuel ICT - partie B : Documentation / rapport du TPI - Question 14 - page 81.

# Les grandes lignes du projet

## Analyse de la situation initiale

Avant de commencer mon Analyse sur le projet, le projet n’existe pas. Les seules informations sur le projet se trouve dans le cahier de charge.

## Analyse de l’état désiré

A la fin de mon TPI le projet devra correspondre au cahier des charges, c’est-à-dire qu’un utilisateur pourras faire ces actions :

Si l’utilisateur est un joueur :

* Visualiser le classement et les matchs qui ont eu lieux.
* Ajouter un match contre un adversaire et y inscrire le score.
* Valider les matchs inscrits par les adversaires.

Si l’utilisateur est un admin :

* Inscrire des joueurs dans le tournois.

L’application aura donc un login qui permettra aux joueurs / admins de se connecter.

## Cahier des charges / exigences du système

### Analyse

* Analyser les tâches complètes du projet et les maquettes préparées auparavant
* Crée le diagramme des cas d’utilisation global.
* Créer les différents diagrammes de séquences des systèmes pour les applications.
* Créer les différents diagrammes d’activités principaux pour les applications

### Conception

* Créer le modèle relationnel de la base de données utile pour l’application Backend.
* Créer le diagramme des classes des applications.
* Créer les différents diagrammes d’interactions des méthodes principales des applications.

### Réalisation

* Réaliser l’implémentation de la base de données.
* Réaliser les différentes requêtes SQL utiles pour la base de données.
* Réaliser l’implémentation des services REST (application BackEnd).
* Développer l’interface homme-machine (application FrontEnd).
* Réaliser l’implémentation des applications et leur communication.
* Tester le fonctionnement complet des applications et leur relation.

## Organisation du projet

Mon TPI sera stocké sur un git ce qui permet d’accéder au projet entier à n’importe quel moment et n’importe où, ça me permet également de garder une trace des diverse versions de mon projet ce qui me permet de revenir en arrière à n’importe quelle moment.

Citer la méthode de gestion de projet utilisée (Méthode en phases, agile, scrum, …)

Etablir la liste des participants au projet avec leur rôle respectif.

Décrire comment sont gérées les sauvegardes de la documentation et du code (fréquence, supports utilisés, utilisation de logiciel de gestion de version, …)

## Technologies

Pour réaliser mon TPI j’ai le choix entre deux technologies : Angular ou VueJS.

### Angular

Angular est un framework robuste et puissant pour le développement d'applications web modernes. Il offre une architecture basée sur les composants, une liaison de données bidirectionnelle, une gestion avancée des routes, la gestion des dépendances, des outils de test et un écosystème solide.

### VueJS

Vue.js est un framework JavaScript progressif, réactif et basé sur les composants. Il offre une approche flexible pour la construction d'interfaces utilisateur interactives, tout en étant léger, performant et bénéficiant d'un écosystème dynamique. C'est un choix populaire pour les développeurs qui cherchent une solution moderne et conviviale pour leurs projets web.

### Choix de technologie

Pour mon TPI j’ai choisi le framework VueJS, voici les raisons pour lesquelles j’ai choisis VueJS plutôt qu’Angular :

1. VueJS est plus facile à apprendre que Angular, n’ayant aucune connaissance de base sur aucun des deux framework j’ai préféré choisir le framework sur lequel j’aurais le plus de facilité à apprendre.
2. VueJS est extrêmement flexible ce qui permet de choisir les fonctionnalités spécifiques en fonction du projet. La faite qu’il adopte une approche progressive est un plus car on adapter à n’importe quel moment en fonction des besoins.
3. VueJS est également beaucoup plus léger que Angular, ce qui permet à l’utilisateur une expérience plus rapide et où il y a le moins de chargement sur ordinateur comme sur téléphone.
4. Bien que la communauté d’Angular soit plus grande VueJS a également une communauté très active et évolutive avec de nombreux support et de ressources.

## Test technologique

### VueJs

#### Introduction

Vue.js est un framework JavaScript qui permet de créer des interfaces utilisateur interactives et réactives dans les applications web. Il offre une liaison de données bidirectionnelle entre le modèle de données et l'interface utilisateur, facilite la création de composants réutilisables et permet une intégration progressive dans les projets existants. Il est populaire pour sa facilité d'apprentissage, sa flexibilité et ses performances élevées.

#### Documentation

La documentation de VueJS est disponible ici : <https://vuejs.org/guide/introduction.html>

#### Création d’un projet VueJS

Pour crée un nouveau projet en vue il faut au préalable télécharger nodeJS afin de disposer de la commande « npm install » qui permet de télécharger des paquets de librairies.

Une fois que node est installé on peut utiliser la commande :

> npm init vue@latest

Cette commande installera et exécutera create-vue, l'outil officiel de création de projets Vue.Il faut ensuite répondre à des questions qui nous permettent de choisir d’installer certaine fonctionnalité de base :

✔ Project name: … <your-project-name>

✔ Add TypeScript? … No / Yes

✔ Add JSX Support? … No / Yes

✔ Add Vue Router for Single Page Application development? … No / Yes

✔ Add Pinia for state management? … No / Yes

✔ Add Vitest for Unit testing? … No / Yes

✔ Add Cypress for both Unit and End-to-End testing? … No / Yes

✔ Add ESLint for code quality? … No / Yes

✔ Add Prettier for code formatting? … No / Yes

Scaffolding project in ./< nouveau-projet >...

Done.

Il suffit ensuite de ce déplacé dans le projet qui viens d’être crée puis lancer le projet avec ces commandes.

> cd <nouveau-projet>

> npm install

> npm run dev

Une image contenant texte, capture d’écran, conception

Description générée automatiquementEt voilà l’application vue a été créé :

### SpringBoot

#### Introduction

Spring Boot est un framework open source pour le développement d'applications Java. Il simplifie le processus de création d'applications en fournissant une configuration et une infrastructure par défaut, permettant aux développeurs de se concentrer sur la logique métier. Il propose une approche "convention plutôt que configuration" en fournissant des valeurs par défaut intelligentes et en simplifiant la configuration. Spring Boot permet de créer rapidement des applications autonomes, des services web RESTful et des microservices, en offrant des fonctionnalités telles que la gestion des dépendances, la surveillance de l'application et la gestion de la sécurité. Il s'intègre étroitement avec l'écosystème Spring, ce qui en fait un choix populaire pour le développement d'applications Java robustes et évolutives.

#### Documentation

La documentation de SpringBoot est disponible ici : <https://spring.io/quickstart>

#### Création d’un projet SpringBoot

Pour crée un nouveau projet SpringBoot leur site propose un outil pour générer un nouveau projet : <https://start.spring.io/>.

Depuis ce site on peut donc configurer et ajouter des dépendances à notre projet, une fois terminé il suffit de générer et télécharger le zip. Le projet est généré automatique il n’y a rien à faire de plus.

#### Installer swagger

Swagger est un outil qui permet de documenter une API de manière efficace, il génère également une page web qui regroupe les diverses requêtes possibles à l’api.

Pour l’installer sur SpringBoot il suffit d’ajouter ces librairies au fichier pom.xml

<!-- Swagger UI Dependency -->

**<dependency>**

**<groupId>**org.springframework.boot**</groupId>**

**<artifactId>**spring-boot-starter-validation**</artifactId>**

**</dependency>**

**<dependency>**

**<groupId>**org.springdoc**</groupId>**

**<artifactId>**springdoc-openapi-starter-webmvc-ui**</artifactId>**

**<version>**2.0.2**</version>**

**</dependency>**

Il faut ensuite définir un Controller et crée une méthode qui correspondra a la requête de l’api.

@RestController

@CrossOrigin(origins = "\*")

@RequestMapping("api/v1/rencontre")

public class MailController {

  @Autowired

  private EmailSenderService senderService;

  @GetMapping(path = "/sendMail")

  public String triggerMail(String *mailTo*, String *subject*, String *body*) throws MessagingException {

    senderService.sendSimpleEmail(*mailTo*, *subject*, *body*);

    return "Mail bien envoyé";

  }

Une image contenant texte, capture d’écran, logiciel, Page web

Description générée automatiquement}

Swagger détectera le Controller et on peut tester la requête depuis swagger après ça.

#### Envoyer des mails

Pour envoyer des mails avec Spring Boot la librairie JavaMail est recommandé. Il faut ajouter la dépendance dans le fichier pom.xml :

<!-- Java mail Dependency-->

**<dependency>**

**<groupId>**org.springframework.boot**</groupId>**

**<artifactId>**spring-boot-starter-mail**</artifactId>**

**</dependency>**

Une fois que la librairie est installer il faut configurer les propriétés du serveur d’envoie (SMTP) dans le fichier ‘application.properties’.

#Mail configuration

spring.mail.host=smtp.gmail.com

spring.mail.port=587

spring.mail.username=<mail>

spring.mail.password=<mot-de-passe>

spring.mail.properties.mail.smtp.auth=true

spring.mail.properties.mail.smtp.starttls.enable=true

Pour le test technologique j’ai utilisé les serveurs de Gmail, j’ai également utilisé le port 587 pour que ce soit le plus sécurisé possible. Il faut ensuite crée un service qui enverras les mails.

@Service

public class EmailSenderService {

  @Autowired

  private JavaMailSender mailSender;

  public String sendSimpleEmail(String *toEmail*, String *subject*, String *body*) {

    String result = "";

    try {

      MimeMessage mimeMessage = mailSender.createMimeMessage();

      MimeMessageHelper mimeMessageHelper = new MimeMessageHelper(mimeMessage, true);

      mimeMessageHelper.setFrom("killian.pasche7@gmail.com");

      mimeMessageHelper.setTo(*toEmail*);

      mimeMessageHelper.setSubject(*subject*);

      mimeMessageHelper.setText(*body*);

      mailSender.send(mimeMessage);

      result = "Mail envoyé !";

    } catch (MessagingException *e*) {

      result = "Erreur envoie de mail : " + e.getMessage();

    }

    return result;

  }

}

Pour tester la méthode j’ai créé une requête qui permet d’envoyer un mail depuis le swagger :

@RestController

@CrossOrigin(origins = "\*")

@RequestMapping("api/v1/mail")

public class MailController {

  @Autowired

  private EmailSenderService senderService;

  @GetMapping(path = "/sendMail")

  public String triggerMail(String *mailTo*, String *subject*, String *body*)

{

    return senderService.sendSimpleEmail(*mailTo*, *subject*, *body*);

  }

}

Une image contenant texte, capture d’écran, logiciel, Page web

Description générée automatiquementJ’ai ensuite lance l’application est entrée une adresse mail ainsi qu’un sujet et un message pour tester :

Une image contenant texte, capture d’écran, Police, Graphique

Description générée automatiquementComme on peut le voir ci-dessous j’ai bien reçu le mail :

# Analyse préliminaire

## Objectifs du système

### Analyse de l’état actuel

### Analyse de l’état désiré

### Objectifs

## Sécurité de l’information et protection des données

Quelles sont les menaces/dangers pour la protection des données et des systèmes et par quelles mesures peuvent-ils être contournés/supprimés. Les données sont-elles soumises à la protection des données?

# Concept

## Exigences du système

## Architecture du système

## Plan d’intégration des systèmes

## Concept d‘implémentation

## Concept de formation

## Concept de tests

Comment les tests sont-ils effectués? Blackbox, Whitebox … .

## Moyens nécessaires

Conseil: D’autres chapitres peuvent s’ajouter à cette partie en fonction du projet.

# Réalisation

## Spécifications détaillées

## Design du système

## Configuration xyz

Conseil: D’autres chapitres peuvent s’ajouter à cette partie en fonction du projet.

# Test

## Procédure de test

Représentation du processus de test..Qui teste quoi à l’aide de quels moyens ?

## Protocol de test

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Nr. | Objet testé | Description du test | Attente | Résultat | Visa |
| 1.0 |  |  |  |  |  |

Protocole de test détaillé avec Visa.

## Signature du protocole de test

|  |  |  |
| --- | --- | --- |
| Date | Nom | Signature |
|  |  |  |

# Conclusion

Conclusion sur le projet en général. Les objectifs sont-ils atteints ? Si non, pourquoi ?

## Améliorations possibles

Y a-t-il des améliorations possibles (nouvelles fonctionnalités, amélioration et/ou modification du design, modification du support de données…) ?

## Auto-évaluation

Quels sont mes sentiments vis-à-vis du travail effectué. Enoncé des facteurs qui expliquent le succès ou l’échec de la réalisation des objectifs du projet.

# Bibliographie: liste des sources et références

Conseil: Cette partie peut contenir une liste des sources et éléments littéraires utilisés.

# Glossaire trier par ordre alphabetique

|  |  |
| --- | --- |
| Terme | Signification |
|  |  |
|  |  |
|  |  |

# Signatures

Je soussigné déclare que les informations contenues dans ce rapport de travail pratique individuel rendu ce jour le XX.XX.20XX dans le cadre de la procédure de qualification de mon CFC d’informaticien/-ne, ne sont pas plagiées. Toutes les informations de sources extérieures ainsi que les informations fournies par des tiers durant le déroulement du travail sont consignées.

L’apprenti/-e doit signer la documentation avant la remise afin de témoigner de la justesse des informations qui y figurent.

|  |  |  |
| --- | --- | --- |
| Date | Nom | Signature |
|  |  |  |

# Annexes

Code imprimé, protocole de séances, etc.: spécifier complètement et annexer à la documentation.

Listings de scripts et programmes. Le travail fourni par la candidate ou le candidat doit être entièrement documenté. Omettre le code généré automatiquement s’il n’est pas nécessaire à la compréhension.

Dans le cas où des manuels ont été créés, vous pouvez les ajouter ici en tant qu’annexe.