133 – Développer des applications WEB

Documentation

Une image contenant Graphique, clipart, symbole, blanc

Le contenu généré par l’IA peut être incorrect.Version 1 du 15.04.2025

Créé le 24.03.2025

Borgeat Nathan, Schmitt Maxime

|  |  |
| --- | --- |
| Une image contenant texte  Description générée automatiquement | Module du 17.03.2025 au 15.04.2025 |

[1 Introduction et contexte du project 4](#_Toc196672182)

[2 Analyse 5](#_Toc196672183)

[2.1 Fichier EA 5](#_Toc196672184)

[2.2 Use case client et use case Rest 5](#_Toc196672185)

[2.2.1 Use case user 5](#_Toc196672186)

[2.2.2 Use case admin 6](#_Toc196672187)

[2.2.3 Use case APIGateway 7](#_Toc196672188)

[2.2.4 Use case Rest user 9](#_Toc196672189)

[2.2.5 Use case Rest admin 11](#_Toc196672190)

[2.3 Diagrammes d’activité 11](#_Toc196672191)

[2.3.1 Diagramme d’activité user 11](#_Toc196672192)

[2.3.2 Diagramme d’activité admin 11](#_Toc196672193)

[2.4 Maquettes du projet 11](#_Toc196672194)

[2.4.1 Maquette user 11](#_Toc196672195)

[2.4.2 Maquette admin 12](#_Toc196672196)

[2.5 Diagrammes de séquence système 12](#_Toc196672197)

[2.5.1 Diagramme de séquence système user 12](#_Toc196672198)

[2.5.2 Diagramme de séquence système admin 14](#_Toc196672199)

[3 Conception 15](#_Toc196672200)

[4 Bases de données 16](#_Toc196672201)

[4.1 Schémas ER 16](#_Toc196672202)

[4.1.1 Schéma ER global 16](#_Toc196672203)

[4.2 Schémas relationnels (MySQL) 16](#_Toc196672204)

[4.2.1 Schéma relationnel user 16](#_Toc196672205)

[4.2.2 Schéma relationnel admin 16](#_Toc196672206)

[5 Descente de code 17](#_Toc196672207)

[6 Outils, langages (versions, définitions, installations) 18](#_Toc196672208)

[6.1 Logiciels 18](#_Toc196672209)

[6.2 Langages 18](#_Toc196672210)

[6.2.1 Backend 18](#_Toc196672211)

[6.2.2 Frontend 18](#_Toc196672212)

[6.3 Autre 18](#_Toc196672213)

[7 Tests de fonctionnement du projet 19](#_Toc196672214)

[8 Auto-évaluations et conclusions 22](#_Toc196672215)

[8.1 Auto-évaluation et conclusion de Nathan Borgeat 22](#_Toc196672216)

[8.2 Auto-évaluation et conclusion de Maxime Schmitt 22](#_Toc196672217)

# Introduction et contexte du project

Voici le contenu du module :

Une image contenant texte, capture d’écran, Police

Le contenu généré par l’IA peut être incorrect.

# Analyse

## Fichier EA



## Use case client et use case Rest

### Use case user

Voici le diagramme use case pour la partie utilisateur :

Une image contenant texte, capture d’écran, diagramme, cercle

Le contenu généré par l’IA peut être incorrect.

Voici un tableau résumant et expliquant les actions possibles :

|  |  |
| --- | --- |
| Acteur | Action |
| Visitor | Signin : Créer un nouveau compte |
| Login : Se connecter à un compte déjà existant |
| User | Logout : Se déconnecter du compte actif |
| Ajouter un hérisson : Ajouter un hérisson à l’utilisateur connecté |
| Modifier un hérisson : Modifier les caractéristiques d’un hérisson |
| Créer un voyage : Ajouter un nouveau voyage pour un hérisson |
| GetFusees : Récupérer toutes les fusées disponibles |
| GetVoyages : Récupérer tous les voyages de l’utilisateur |
| GetHerissons : Récupérer tous les hérissons de l’utilisateur |

### Use case admin

Une image contenant capture d’écran, texte, diagramme, ligne

Le contenu généré par l’IA peut être incorrect.

|  |  |
| --- | --- |
| Acteur | Action |
| Visiteur | Login : Se connecter à un compte déjà existant |
| Admin | Logout : Se déconnecter du compte actif |
| Rafraîchir les hérissons : recharger la liste des hérissons |
| Afficher les fusées : récupère les fusées pour les afficher |
| Afficher les vols : récupère les vols pour les afficher |
| Démonter fusée : démonte une fusée |
| Modifier fusée : modifier une fusée |
| Créer fusée : crée une fusée |

### Use case APIGateway

Voici le diagramme use case de la partie Gateway.

Une image contenant capture d’écran, diagramme, texte, ligne

Le contenu généré par l’IA peut être incorrect.

Voici un tableau résumant et expliquant les actions possibles :

|  |  |
| --- | --- |
| Acteur | Action |
| Client | Signin : Créer un nouveau compte |
| Login : Se connecter à un compte déjà existant |
| Logout : Se déconnecter du compte actif |
| Ajouter un hérisson : Ajouter un hérisson à l’utilisateur connecté |
| Modifier un hérisson : Modifier les caractéristiques d’un hérisson |
| Créer un voyage : Ajouter un nouveau voyage pour un hérisson |
| GetFusees : Récupérer toutes les fusées disponibles |
| GetVoyages : Récupérer tous les voyages de l’utilisateur |
| GetHerissons : Récupérer tous les hérissons de l’utilisateur |
| Admin | GetVoyages : Récupérer tous les voyages de l’utilisateur |
| GetFusees : Récupérer toutes les fusées disponibles |
| Créer fusée : crée une fusée |
| Modifier fusée : modifier une fusée |
| Démonter fusée : démonte une fusée |
| Annuler vol : annule un vol en cours |

### Use case Rest user

Voici le diagramme des cas d’utilisation de l’API Rest client :

Une image contenant texte, capture d’écran, diagramme, ligne

Le contenu généré par l’IA peut être incorrect.

Voici un tableau résumant et expliquant les actions possibles :

|  |  |
| --- | --- |
| Acteur | Action |
| APIGateway | Signin : Créer un nouveau compte |
| Login : Se connecter à un compte déjà existant |
| Ajouter un hérisson : Ajouter un hérisson à l’utilisateur connecté |
| Modifier un hérisson : Modifier les caractéristiques d’un hérisson |
| Créer un voyage : Ajouter un nouveau voyage pour un hérisson |
| GetVoyages : Récupérer tous les voyages de l’utilisateur |
| GetHerissons : Récupérer tous les hérissons de l’utilisateur |

### Use case Rest admin

## Diagrammes d’activité

### Diagramme d’activité user

Voici un diagramme d’activité sur l’ajout d’un hérisson :

Une image contenant texte, capture d’écran, diagramme, nombre

Le contenu généré par l’IA peut être incorrect.

### Diagramme d’activité admin

Une image contenant texte, diagramme, ligne, capture d’écran

Le contenu généré par l’IA peut être incorrect.

## Maquettes du projet

### Maquette user

Page d’accueil une fois connecté :

Une image contenant texte, capture d’écran, nombre, conception

Le contenu généré par l’IA peut être incorrect.

### Maquette admin

Une image contenant capture d’écran, Rectangle, texte, diagramme

Le contenu généré par l’IA peut être incorrect.

## Diagrammes de séquence système

### Diagramme de séquence système user

Voici le diagramme de séquence système pour l’ajout d’un hérisson :

Une image contenant texte, capture d’écran, Rectangle, diagramme

Le contenu généré par l’IA peut être incorrect.

### Diagramme de séquence système admin

Une image contenant texte, capture d’écran, diagramme, Parallèle

Le contenu généré par l’IA peut être incorrect.

# Conception

Fichier EA point [2.1](#_Fichier_EA)

# Bases de données

## Schémas ER

### Schéma ER global

Dans le schéma ci-dessous, nous pouvons voir nos deux bases de données.

La première contient des utilisateurs qui peuvent créer des hérissons et des voyages.

Ces voyages utilisent des fusées qui se trouvent dans la base de données « Admin ». Cette DB contient aussi des administrateurs.



## Schémas relationnels (MySQL)

### Schéma relationnel user

Ce schéma représente les relations plus précisément que le schéma relationnel vu au-dessus.

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

Le contenu généré par l’IA peut être incorrect.

### Schéma relationnel admin

Voici simplement le schéma relationnel admin :

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

Le contenu généré par l’IA peut être incorrect.

# Descente de code

Cette descente de code va expliquer comment ajouter un hérisson pour un utilisateur.

Avant toute chose l’utilisateur doit être connecté sur le site utilisateur : [Lien du site](https://borgeatn.emf-informatique.ch/133/)

Une fois connecté, il faut remplir les champs « Nom » et « Caractéristique ».

L’utilisateur n’a plus qu’à appuyer sur le bouton « Nouveau herisson » pour ajouter un hérisson.

Passons maintenant au code en lui-même, le bouton a un écouteur qui appelle la méthode « nouveauHerisson » et qui est déclaré dans le constructeur :

$("#nouveauherisson").click(() => {

  this.nouveauHerisson();

});

Cette méthode obtient les données du nouveau hérisson et vérifie si elles sont valables, avant de les envoyer au service http.

Voici la méthode appelée dans le fichier serviceHttp.js :

addHerisson(nom, caracteristique, successCallback, errorCallback) {

  $.ajax({

    xhrFields: {

      withCredentials: true,

    },

    crossDomain: true,

    type: "POST",

    dataType: "text",

    url: BASE\_URL + "addHerisson",

    data: {

      nom: nom,

      caracteristique: caracteristique

    },

    success: successCallback,

    error: errorCallback,

  });

}

La requête ajax est envoyée en direction de l’API gateway qui l’intercepte dans son fichier « Douanier.java » :

    @PostMapping(path = "/addHerisson")

    public @ResponseBody ResponseEntity<String> addNewHerisson(HttpSession httpsession, @RequestParam String nom,

            @RequestParam String caracteristique) {

        if (httpsession.getAttribute("pk") != null && httpsession.getAttribute("admin") != null) {

            return userManager.addHerisson(nom, caracteristique, (Integer) httpsession.getAttribute("pk"));

        } else {

            return ResponseEntity.badRequest().body("Non connecté");

        }

    }

Une nouvelle requête est créée dans le userManager pour être transmise au service rest user :

public ResponseEntity<String> addHerisson(String nom, String caracteristique, Integer fk\_utilisateur) {

    MultiValueMap<String, String> herissonInfos = new LinkedMultiValueMap<>();

    herissonInfos.add("nom", nom);

    herissonInfos.add("caracteristique", caracteristique);

    herissonInfos.add("fk\_utilisateur", fk\_utilisateur.toString());

    ResponseEntity<String> response = restTemplate.postForEntity(url + "addHerisson", herissonInfos, String.class);

    return response;

}

Cette nouvelle requête est interceptée par le contrôleur du service comme montré avant.

L’ajout dans la base de données peut désormais être fait grâce à la méthode suivante :

public String addHerisson(String nom, String caracteristique, Integer fk\_utilisateur) {

    Herisson newHerisson = new Herisson();

    newHerisson.setName(nom);

    newHerisson.setCaracteristique(caracteristique);

    newHerisson.setUtilisateur(fk\_utilisateur);

    herissonRepository.save(newHerisson);

    return "saved";

}

# Outils, langages (versions, définitions, installations)

## Logiciels

Visual Studio Code (plateforme de développement) 1.88.1

MySQL Workbench (élaboration de la DB) 8.0.38

Postman (plateforme de test) 7.36.1

Docker (hébergement en local) 4.39.0

## Langages

### Backend

Java (Maven) 17.0.14

### Frontend

HTML 5.2

CSS 5.01

Javascript 22.12

## Autre

SpringBoot v3.4.4

JQuery 3.7.0

Apache Tomcat 10.1.39

# Tests de fonctionnement du projet

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Droit d’accès, inscription et permissions | | | | |
| N° | Description du test | Résultat attendu | Résultat reçu | VISA |
| 1 | Si l'utilisateur n'est pas connecté, toutes les fonctionnalités lui sont bloquées | Toutes les requêtes retournent "Non connecté" | Une image contenant texte, capture d’écran, Police, nombre  Le contenu généré par l’IA peut être incorrect. | OK |
| 2 | Si l’utilisateur n’est pas connecté en tant qu’administrateur, les fonctionnalités administrateur lui sont bloquées | Toutes les requêtes admin retournent « Compte admin requis » |  | OK |
| 3 | Au clic sur "Se connecter", l'utilisateur se connecte avec les identifiants présents dans les champs s'ils sont corrects | Avec des identifiants corrects, les méthodes utilisateur de base sont accessibles (test avec ajout de hérisson) |  | OK |
| 4 | Si les identifiants de la connexion sont ceux d'un administrateur, l'utilisateur obtient les droits d'un administrateur | Les méthodes administrateurs sont accessibles (test avec ajout de fusée) |  | OK |
| 5 | Au clic sur "Se connecter", l'utilisateur ne peut pas se connecter avec les identifiants présents dans les champs s'ils sont incorrects | Popup "Login incorrect" |  | OK |
| 6 | L’inscription crée un nouvel utilisateur avec lequel on peut se connecter et agir comme un utilisateur non-admin. | Les infos du « signin » fonctionnent pour le « login » après inscription. | Inscription :  Connexion : | OK |
| 7 | Tous les utilisateurs et admin connectés ont accès aux requêtes « GET » (sauf « getAllVoyages » et « getHerissons ») | Les requêtes « GET » (sauf « getAllVoyages » et « getHerissons ») sont accessibles aux admins comme aux utilisateurs | Admin :  Utilisateur : | OK |
| 8 | Si l’utilisateur se déconnecte, il n’a plus les accès d’un utilisateur connecté | Après une déconnexion, l’utilisateur revient à ses droits d’avant la connexion |  | OK |
| Requêtes | | | | |
| N° | Description du test | Résultat attendu | Résultat reçu | VISA |
| 1 | La requête « getHerissons » retourne tous les hérissons de l’utilisateur connecté | Tous les hérissons de l’utilisateur connecté s’affichent |  | OK |
| 2 | La requête « addHerisson » ajoute un hérisson à l’utilisateur connecté. | Le hérisson s’ajoute bien et est utilisable |  | OK |
| 3 | La requête « putHérisson » remplace les infos du hérisson | Les infos sont bien remplacées | Test avec le hérisson du test 2 : | OK |
| 4 | La requête « getVoyages » renvoie tous les voyages de l’utilisateur. | La liste de tous les voyages selon l’utilisateur |  | OK |
| 5 | La requête « getAllVoyages » renvoie tous les voyages de l’utilisateur. | La liste de tous les voyages |  | OK |
| 6 | La requête « addVoyages » ajoute un voyage à l’utilisateur en y ajoutant le hérisson prenant part au voyage. | Un nouveau voyage est ajouté et lié à un hérisson |  | OK |
| 7 | La requête « addFusee » ajoute une fusée avec son nom | La fusée est ajoutée avec son nom |  | OK |
| 8 | La requête « putFusee » modifie le nom de la fusée | Le nom de la fusée est modifié |  | OK |
| 9 | La requête « deleteFusee » supprime une fusee | La fusée est supprimée |  | OK |
| 10 | La requête « getFusee » affiche toutes les fusées | Toutes les fusées s’affichent |  | OK |

# Auto-évaluations et conclusions

## Auto-évaluation et conclusion de Nathan Borgeat

Personnellement, j’ai trouvé intéressant de voir la différence entre PHP et Java dans l’interaction avec une base données. Grâce à ça, j’ai pu voir les différents avantages et inconvénients des deux langages.

J’ai cependant moyennement apprécié le fait de devoir faire le projet en groupe, puisqu’il s’agit d’un projet noté, si une des personnes travaille moins vite que l’autre cela peut être pénalisant pour cette dernière. Finalement tout s’est bien passé pour ce module.

## Auto-évaluation et conclusion de Maxime Schmitt

Ce module m’a tout d’abord appris qu’il était possible d’utiliser java pour travailler avec du développement web, et j’ai trouvé intéressant de lier mes connaissances avec les principes de développement web déjà travaillés dans le module 151.

La communication inter-modulaire était également très plaisante, cela donne vraiment la sensation d’avoir effectué le travail de A à Z.

En conclusion, donc, module très enrichissant et très complémentaire en termes de compétences.