ARCHITECTURE LOGICIELLE

Type :	**PROJET**
Formations :	Ynov Informatique
Promotions :	Bachelor 3
UF :	SPE INGENIERIE LOGICIELLE et BASE DE DONNEES

1. CADRE DU PROJET

Ce projet permet l'évaluation des compétences acquises grâce aux modules de l'UF « Développement Logiciels ». Pour ce faire, ce projet devra être réalisé en groupe de 2.

Vous pouvez soumettre un projet personnel dont le contenu et les fonctionnalités devront respecter des conditions décrites dans la partie « Projet personnel ». Ce projet devra être validé par l'établissement.

Si vous n'avez pas d'idée de projet, vous avez le choix parmi une liste de projets proposés dans la partie « Projets au choix ».

Un bonus sera apporté aux projets personnels et aux groupes qui se challengent en proposant des fonctionnalités plus poussées.

Vous êtes totalement libre quant aux choix technologiques. Nous vous conseillons d'utiliser les langages de programmations et les outils vus avec vos intervenants mais acceptons toute technologie de Développement logiciel. Ex : C++, Java, C#, Python, NodeJS + Electron...

Seul l'aspect fonctionnel sera pris en compte et non l'aspect graphique.

Date de début : ……..

Date de rendu : ……..

2. OBJECTIFS DE FORMATION VISÉS

Vous serez évalués sur les compétences suivantes : *UF SPÉ INGÉNIERIE LOGICIELLE et Base de données.*

ARCHITECTURE LOGICIELLE

- Effectuer des choix technologiques et respecter leurs conventions
- Maîtriser les Design Patterns
- Maîtriser l'utilisation d'un ORM

ARCHITECTURE MICRO-SERVICES

- Maîtriser la séparation des responsabilités
- Concevoir une API REST DEVELOPPEMENT ASYNCHRONE
- Détecter les situations ayant besoin de l'asynchrone
- Construire autour d'informations asynchrones

BAS NIVEAU / MACHINE LEARNING

- Réaliser un défi technique

3. PREREQUIS

Voici les modules d'enseignement prérequis à ce projet :

- Bas niveau
- Machine Learning
- Architecture micro-services
- Développement asynchrone

4. LIVRABLES

- dépôt GIT de votre logiciel à jour
- exécutable de vos logiciels
- un document de présentation de votre projet (rôles de chacun, technologies utilisées, structure algorithmique, fonctionnalités majeures, captures d'écran...)
- slides de votre présentation

5. MODALITÉS D'ÉVALUATION DU PROJET

Vous serez évalué sur l'ensemble des productions. L'évaluation prendra aussi la forme d'une présentation orale de synthèse d'environ 15 minutes accompagnée d'un support de présentation et d'une démonstration des fonctionnalités du site mises en place.

Le jury sera composé d'une partie des intervenants des cours de l'UF « Développement Logiciels ».

Un temps de questions-réponses d'une durée de 5 minutes sera prévu à l'issue des 15 minutes.

Des évaluations intermédiaires auront également lieu au cours du déroulement du Projet.

RÉCAPITULATIF DE LA GRILLE DE NOTATION

Le projet personnel décrit ci-dessous comporte 19 points de difficulté. A vous d'y ajouter des fonctionnalités pour atteindre les 28 points minimums requis. Vous pouvez vous référer aux projets proposés pour vous faire une idée des fonctionnalités et des points qui leurs sont attribuées. Les fonctionnalités devront être validées par l'établissement.

Un point bonus sera ajouté à la note finale si vous choisissez de réaliser un projet personnel.

Selon le nombre de points de difficulté total mis en place sur votre projet, des points bonus seront accordés selon les paliers suivants :

Points de difficulté : Entre 28 et 35	*0 point bonus*
Points de difficulté : Entre 35 et 40	*1 point bonus*
Points de difficulté : Plus de 40	*2 points bonus*

Les projets proposés devront être réalisés dans leurs intégralité pour obtenir la totalité des points.

6. DESCRIPTIF DU PROJET

Vous êtes totalement libre quant à l'apparence de vos interfaces. L'utilisation de librairies est autorisée et encouragée afin de gagner en rapidité de développement.

Vous avez la possibilité de choisir entre un projet personnel ou un projet proposé.

LISTE DES PROJETS AU CHOIX :

PROJET PERSONNEL :

Le projet personnel devra atteindre ou dépasser les 28 points en degré de difficulté. Les fonctionnalités obligatoires représentent déjà 19 points en degré de difficulté, vous devez alors trouver des fonctionnalités valant au minimum 10 points. Vous pouvez vous référer à la liste de « Projet au choix » pour vous faire une idée des points alloués par degré de difficulté.

Le projet choisit devra avoir au minimum :

- une architecture en services (API + IHM au minimum) *Difficulté : 4*
- l'utilisation d'au moins 3 Design Patterns de façon utile *Difficulté : 3*
- une base de données (SQL ou NoSQL au choix) *Difficulté : 2*
- l'utilisation d'un ORM *Difficulté : 4*
- réaliser au moins un défi technique (ex : génération de code, IA, machine learning, BI, simulation physique...). *Difficulté : 6*

Degré de difficulté total : 19 points

(Bonus si le projet personnel est choisi)

1er PROJET : LOGICIEL ADAPTATIF

Présentation

Ce projet contient 3 composantes : une API, une IHM et une BDD. Le but est de créer une IHM de type CRUD qui s'adaptera à tout modèle de données.

Fonctionnalités

- Modèle de données (vous modifierez ce modèle de vous-même afin de tester la partie adaptative) : *Difficulté :* 2
  - Des clients, composés de :
    - Un nom, un prénom
    - Une adresse mail
    - Une date de création
  - Des factures, composés de :

        - Une référence à un client
        - Une date d'émission
        - Une donnée indiquant si elle a été payé ou non
        - Une date de paiement
        - Un prix
        - Des références à des produits
    - Des produits, composés de:
        - Un nom
        - Un stock
        - Une photo
        - Un prix
        - Des références à des factures
- L'API REST :
    - Doit faire le lien avec la BDD via un ORM (l'ORM peut même créer la BDD) *Difficulté :* 4
    - Doit fournir des routes HTTP pour toutes les actions CRUD de toutes les tables du modèle de données *Difficulté :* 5
    - Doit fournir (en une route HTTP ou plusieurs) toutes les informations relatives à la composition du modèle de données (comme un MCD mais en JSON) *Difficulté :* 3
- L'IHM :
    - Doit fournir des pages pour toutes les actions CRUD de toutes les tables du modèle de données (Ex : par table, une page liste / suppression et une page ajout / modification) *Difficulté :* 3
    - L'IHM doit utiliser la même page pour une action CRUD, quel que soit la table. La page devra se construire toute seule en fonction des informations de composition du modèle fournis par l'API *Difficulté :* 6
    - Pour ce faire, il faudra créer des composants génériques pour chaque type de champ *Difficulté :* 6

Défi technique : Réalisation d'un moteur de segmentation de clientèle

Vous implémenterez un (ou plusieurs algorithmes) de classification (apprentissage supervisé) ou de clustering (apprentissage non supervisé) pour segmenter vos clients/prospects en groupes présentant des caractéristiques similaires.

Voici le détail des étapes à réaliser pour :

- Nettoyer, sélectionner et agréger les données utiles *Difficulté 2*
- Entraîner l'algorithme utilisé sur les données sélectionnées *Difficulté 6*
- Evaluer la qualité de l'apprentissage *Difficulté 4*
- Prévoir un système pour mettre à jour l'entraînement de l'algorithme utilisé lorsque suffisamment de nouvelles données sont recueillies *Difficulté 2*
- Une fois la qualité de l'apprentissage validée, mettre en place un système renvoyant les résultats vers l'application (par exemple, via l'ORM) *Difficulté 3*

Degré de difficulté total : 44 points

2ème PROJET : IA DE JEU

Présentation

Ce projet est un jeu vidéo de type “Labyrinthe”. L’utilisateur va créer des niveaux, et voir comment l’IA du jeu s’en sort dessus. Il contient 3 composantes : deux IHM et une BDD

Fonctionnalités

- Modèle de données : *Difficulté :* 3
  - Des types d’obstacles, avec pour chacun :
    - S'ils sont traversables ou non
    - Leur effet sur l’IA si on les traverse
    - Leur nom
    - Leur apparence
    - Le nombre minimum contenu dans un niveau
    - Le nombre maximum dans un niveau
  - Des niveaux, avec pour chacun :
    - Un nom
    - Un créateur
    - Une date de création
    - Une date de modification
    - Une composition
  - Des tests de niveau, avec pour chacun :
    - Une référence au niveau
    - Une date de passage
    - Un résultat
- Pour les types d’obstacle, avoir au minimum :
  - Point de départ
  - Point d’arrivée
  - Mur
  - Piège (met en échec l’IA)
  - Boue (ralenti l’IA)
- Le lien avec la BDD doit se faire via un ORM *Difficulté :* 4
- Logiciel de test de l’IA :
  - L’utilisateur choisi un niveau crée *Difficulté :* 2
  - L’IA apparait sur l’élément Point de Départ *Difficulté :* 1
  - L’IA tente d’atteindre le Point d’arrivé, à l’aide d’un algorithme au choix (Ex : machine learning ou pathfinding A*) *Difficulté :* 8
  - Le résultat du test est entré en BDD *Difficulté :* 1
- Logiciel d’édition de niveau :
  - L’utilisateur doit pouvoir créer ou éditer un niveau existant *Difficulté :* 2

- Une boite à outils permettant de choisir parmi tous les types d'obstacle *Difficulté :* 2
- L'utilisateur doit pouvoir placer, déplacer, supprimer des obstacles sur une map de jeu 2D *Difficulté :* 5

Degré de difficulté total : 28 points

7. RESSOURCES COMPLEMENTAIRES

Un framework correspondant bien à l'API du projet n°1 :
https://spring.io/guides/gs/accessing-data-rest/

Une explication parfaite des algorithmes de pathfinding :
https://www.redblobgames.com/pathfinding/a-star/introduction.html

A vous de trouver les ressources nécessaires en fonction des technologies choisies.

8. BESOINS MATERIELS ET LOGICIELS

- Un IDE
- Un langage de programmation orienté objet
- GIT