PROJET : TECHNOLOGIES WEB

Type : Projet de développement web
Formations : Ynov Informatique
Promotions : Bachelor 2
UF : Technologies WEB & Base de données

---

1. CADRE DU PROJET

Ce projet permet l'évaluation des compétences acquises grâce aux modules des UF «**Technologies Web**» et **« Base de données »**. Il devra être réalisé en **binôme**.

Exigences générales

- Le projet devra être un **site web dynamique** incluant un **système d'authentification** et une **gestion des données** via une base de données.
- Vous devez organiser le projet en suivant une **architecture logique et modulaire**.
- L'objectif est de concevoir une application respectant **les standards du développement web moderne**.
- L'utilisation d'un **framework web moderne** et d'un **ORM** (ou **ODM** en cas de base de données non relationnelle) est obligatoire.
- L'accent est mis sur **la maintenabilité, l'évolutivité et la qualité du code**.
- Vous devrez **documenter vos choix techniques** et fournir un **modèle de base de données clair**.

Date de début : 3 mars 2025
Date de rendu : 13 juin 2025

---

2. DÉTAILS DES ATTENTES TECHNIQUES

2.1. Technologies à choisir et justifier

Votre projet devra être conçu avec les technologies suivantes :

- **Framework web** : Laravel, Symfony, Django, Express.js...
- **Base de données** : MySQL (recommandé), PostgreSQL, MongoDB...

- **Langages** : PHP, Python, JavaScript, Node.js...
- **Gestion de version** : Git (obligatoire, GitHub ou GitLab recommandé).
- **Gestion des dépendances** : Composer (PHP), npm/yarn (JavaScript), pip (Python).
- **Front-end** : Bootstrap, Tailwind, Vue.js, React (optionnel).

Vous devez expliquer **les choix techniques effectués** dans votre rapport.

2.2. Conception et modélisation

- Vous devez réfléchir à l'**architecture** de votre application et la représenter sous forme de **schéma UML** (si vous ne connaissez pas UML, faites des recherches, ça vous servira énormément pour l'année prochaine).
- La **base de données** doit être modélisée avec **MERISE** si elle est relationnelle (même consigne, faites des recherches si nécessaire).

Si vous optez pour une base de données **non relationnelle**, vous devez concevoir un **modèle de données en utilisant les principes des bases NoSQL**. Vous pouvez vous documenter sur les schémas de données spécifiques aux bases NoSQL (documents, colonnes, graphes, clé-valeur) et choisir le modèle le plus adapté à votre projet. Une justification de votre choix devra être incluse dans votre rapport.

2.3. Fonctionnalités requises

L'application doit obligatoirement inclure :

1. **Une authentification sécurisée** (utilisateurs, rôles).
2. **Une base de données relationnelle** (stockage structuré et persistant des informations).
3. **Un CRUD (Create, Read, Update, Delete)** sur au moins une entité principale.
4. **Une interface utilisateur responsive** et ergonomique.
5. **Un dépôt Git bien organisé** avec commits fréquents et clairs.
6. **Un site web en ligne** (hébergé sur un serveur ou une plateforme comme Heroku, Vercel, AlwaysData...).

---

3. LIVRABLES ATTENDUS

1. **Spécifications fonctionnelles**
    - Décrire précisément les fonctionnalités attendues de l'application.

    - Définir les rôles des utilisateurs et leurs interactions avec le système.
    - Présenter les cas d'utilisation principaux avec des scénarios détaillés.
    - Lister les contraintes et exigences non fonctionnelles (performance, sécurité, accessibilité).
2. **Code source propre et bien structuré**, versionné sur GitHub/GitLab en PUBLIC.
3. **Documentation complète** (README, architecture du projet, choix techniques).
4. **Schéma UML et modèle MERISE** (ou équivalent) décrivant votre conception.
5. **Dépôt de la base de données** (fichier SQL ou instructions de migration).
6. **Présentation orale de 15 minutes** avec démonstration du projet.

---

4. EXEMPLES DE PROJETS ATTENDUS

4.1. Projet E-Commerce

- Site de vente de jeux vidéo en ligne.
- Gestion des utilisateurs (clients et administrateurs).
- Paiement fictif et gestion des commandes.
- Système de facturation (PDF et envoi par mail).
- Gestion des stocks et catalogue de produits.

4.2. Projet Livraison de Repas

- Plateforme permettant aux utilisateurs de commander des repas auprès de restaurants.
- Gestion des utilisateurs (clients, restaurateurs, administrateurs).
- Paiement factice, validation et suivi des commandes.
- Gestion du stock des plats et affichage dynamique des restaurants.

4.3. Projet Plateforme de Réservations

- Une plateforme permettant aux utilisateurs de réserver des créneaux horaires pour des services (ex. salles de sport, coiffeurs, consultations médicales, etc.).
- Gestion des utilisateurs (clients et administrateurs).
- Gestion des disponibilités et des créneaux en temps réel.
- Paiement en ligne (factice) et notifications de rappel.

- Interface dynamique avec un calendrier interactif.

4.4. Projet Réseau Social Thématique

- Un mini-réseau social basé sur un thème spécifique (ex. passionnés de cinéma, musique, jeux vidéo, etc.).
- Système d'authentification avec profils utilisateurs.
- Publication de contenus (posts, commentaires, likes).
- Système de suivi (follow/unfollow).
- Gestion des interactions via un fil d'actualité.

4.5. Projet Gestion de Bibliothèque Numérique

- Un système permettant aux utilisateurs d'accéder à une bibliothèque numérique.
- Catégorisation et recherche avancée de documents.
- Gestion des droits d'accès (lecture, téléchargement, etc.).
- Intégration d'un système de favoris et de recommandations.
- Interface utilisateur fluide et responsive.

---

5. ÉVALUATION & NOTATION

Votre projet sera évalué sur :

Critères	**Points attribués**
Choix technologiques et justification	5
Architecture et conception (UML, MERISE)	10
Fonctionnalités de base (CRUD, Auth)	20
Qualité du code (organisation, bonnes pratiques)	10
Interface utilisateur et expérience utilisateur	10
Base de données et gestion des requêtes SQL	5
Hébergement et déploiement	10
Présentation et démonstration	20

Critères	Points attribués
Réponses aux questions	10
Bonus pour fonctionnalités avancées	+10
Total	**100**

---

6. RESSOURCES RECOMMANDÉES

- Laravel : https://laravel.com/docs/.
- Git : https://www.grafikart.fr/formations/git.
- UML : https://www.uml-diagrams.org/.
- MERISE : https://merise.developpez.com/.
- Bootstrap : https://getbootstrap.com/.
- VueJS : https://vuejs.org.
- ReactJS : https://react.dev
- Angular : https://angular.io
- Svelte : https://svelte.dev
- Alpine.js : https://alpinejs.dev
- Node.js : https://nodejs.org/en/docs
- Express.js : https://expressjs.com/en/starter/installing.html
- Django : https://docs.djangoproject.com/en/stable/
- Flask : https://flask.palletsprojects.com/en/latest/
- PostgreSQL : https://www.postgresql.org/docs/
- MySQL : https://dev.mysql.com/doc/
- MongoDB : https://www.mongodb.com/docs/
- Tailwind CSS : https://tailwindcss.com/docs
- TypeScript : https://www.typescriptlang.org/docs/

---

7. MATÉRIEL ET LOGICIELS UTILES

- **IDE** : Visual Studio Code, PhpStorm, PyCharm, etc.
- **Serveur web local** : WAMP, MAMP, LAMP…
- **Serveur de production** : Heroku, Vercel, AlwaysData…
- **Système de gestion de version** : Git (repository PUBLIC).
