Le projet consiste à créer un site web de trois pages pour s'initier aux technologies web (HTML, CSS, java-script) et à l'utilisation d'une API RESTful pour afficher des données dynamiques. Il vise à développer des compétences en conception, programmation, modélisation, gestion de projet et collaboration, avec l'utilisation de GitHub pour la gestion des versions.

Objectifs principaux :

1. Développement web : Apprendre les bases de HTML, CSS (pour un design responsif), et java-script (pour l'interactivité et la manipulation des données).
2. Interaction avec une API RESTful : Récupérer et afficher dynamiquement des données externes (météo, films, etc.), et permettre à l'utilisateur de les filtrer.
3. Modélisation : Utiliser des diagrammes SYSML pour planifier l'architecture du site web et visualiser les flux de données.
4. Collaboration avec GitHub : Travailler en équipe, gérer les versions du projet, et utiliser des pull requests.
5. Design et UX : Créer une interface agréable, avec une charte graphique, et un design responsif.

Organisation et Livrables :

- Code source complet.
- Diagrammes SYSML.
- Documentation GitHub.
- Charte graphique détaillée.

Pages à créer :

1. Page d'accueil : Présentation du site, navigation, et aperçu des données de l'API.
2. Page d'informations : Affichage des données de l'API avec possibilités d'interaction (filtrage, tri).
3. Troisième page : Au choix (contact, authentification, ou inscription) avec formulaires et validation côté client.

Cahier des charges pour un site web sur le thème des jeux vidéo

1. Contexte du projet

Le projet consiste en la création d'un site web interactif de trois pages ayant pour thème principal les jeux vidéo. Ce site servira à exposer les tendances du moment, à permettre aux utilisateurs de filtrer et d'interagir avec les données de jeux vidéo populaires, et à inclure une fonctionnalité d'authentification ou de contact. Il exploitera les technologies HTML, CSS, java-script et intégrera une API RESTful pour afficher des informations dynamiques sur les jeux vidéo. Ce projet vise également à développer des compétences en modélisation SYSML et en gestion de version avec GitHub.

2. Objectifs du site

- Présenter des informations en temps réel sur les jeux vidéo, comme les jeux les plus populaires ou les nouvelles sorties.
- Permettre à l'utilisateur de consulter et d'interagir avec les données (par exemple, en filtrant les jeux par plateforme, genre ou popularité).
- Offrir une interface moderne, intuitive et responsive pour s'adapter à toutes les tailles d'écrans (ordinateurs, tablettes, smartphones).
- Fournir une page d'accueil, une page d'informations utilisant l'API jeux vidéo et une troisième page au choix, soit une page de contact, soit une page d'authentification.

3. Technologies utilisées

-**Frontend** : HTML5, CSS3, java-script
- **Backend** : API RESTful (par exemple l'API RAWG pour les données de jeux vidéo)
- **Outils de modélisation** : SYSML pour modéliser l'architecture du site
- **Gestion de version** : Git et GitHub

4. Pages du site

Le site se compose de trois pages principales :

4.1. Page d'accueil

- **Description générale :** Présentation du site sur le thème des jeux vidéo.
-Contenu dynamique : Un aperçu des jeux vidéo les plus populaires récupérés via l'API (nom, image, date de sortie, plateforme, genre).
- **Menu de navigation** : Liens vers les autres pages du site (Page d'informations/Services et troisième page au choix).
- **Design responsive** : Mise en page fluide et adaptative selon la taille de l'écran, conformément à la charte graphique.

4.2. Page d'informations/services (intégration de l'API)

- **Données de jeux vidéo** : Affichage dynamique des données récupérées via l'API (liste de jeux vidéo, par exemple).
- Interactions utilisateur :
  - **Filtrer les jeux par plateforme** (PC, Xbox, PlayStation, etc.), genre (action, RPG, aventure), ou par note.
  - **Trier les résultats par popularité**, date de sortie, ou note des utilisateurs.
- F**onctionnalités java-script** : Ajout d'interactions pour améliorer l'expérience utilisateur (ex : actualisation automatique des données sans rechargement complet de la page).
- **Mise en page** : Les informations sont présentées sous forme de cartes ou de tableaux pour une meilleure lisibilité.

4.3. Troisième page au choix

- Page d'authentification :
  - Formulaire de connexion avec nom d'utilisateur et mot de passe.
  - Validation des champs avec java-script.
  - Simulation d'une connexion réussie ou d'un message d'erreur.
  - Option d'ajouter un lien pour "Mot de passe oublié" ou "Créer un compte".

5.Charte graphique

- Couleurs :
  - Palette de couleurs inspirée de l'univers des jeux vidéo (par exemple, noir, bleu électrique, violet).
  - Accentuation des boutons et éléments interactifs par des couleurs contrastées (orange ou vert).
- Typographies :
  - Utilisation de polices modernes et lisibles.
  - Taille de police adaptée pour la lisibilité sur tous les appareils.
- **Espacement** :
  - Utilisation d'espacements généreux pour structurer le contenu et offrir une navigation claire.
  - Respect d'un design minimaliste et ergonomique.

6. Modélisation SYSML

- Diagrammes de cas d'utilisation : Visualisation des interactions entre l'utilisateur et le site (navigation entre les pages, interaction avec les données de l'API).
- Diagrammes de séquence : Illustration des échanges entre le serveur, l'API, et le navigateur de l'utilisateur (affichage des données en temps réel).
- Diagrammes de blocs : Structure du site, des composants (front-end, back-end) et des flux de données entre les différents systèmes.

7. Fonctionnalités techniques

- **HTML/CSS** : Structuration et stylisation des pages avec un design responsive.
-java-script : Interaction avec l'API pour récupérer et manipuler les données de jeux vidéo. Ajout d'éléments interactifs (filtrage, tri, validation des formulaires).
- **API RESTful** : Connexion avec une API de jeux vidéo (par exemple RAWG) pour récupérer des informations dynamiques.
- **Gestion de version** : Utilisation de Git et GitHub pour le suivi des versions, les branches de développement et les pull requests.

8. Exigences fonctionnelles

- Réactivité : Le site doit être optimisé pour les ordinateurs, tablettes et smartphones.
- Performance : Temps de chargement rapide même avec l'intégration d'éléments dynamiques depuis l'API.
- Accessibilité : Le site doit respecter les normes d'accessibilité web pour tous les utilisateurs.

9. Livrables

- **Code source** : Fichiers HTML, CSS et java-script comprenant les trois pages du site.
- Diagrammes SYSML : Cas d'utilisation, séquence et blocs pour l'architecture du site.
-Documentation GitHub : README détaillant le projet, les technologies utilisées et les étapes de développement.
- Charte graphique : Explication des couleurs, typographies et principes de mise en page employés.

10. Gestion du projet

Le projet doit être livré en respectant les jalons suivants :
- Étape 1 : Définition de la charte graphique et des maquettes
- Étape 2 : Modélisation de l'architecture du site avec SYSML
- Étape 3 : Développement des pages avec intégration de l'API.
- Étape 4 : Tests, validation et corrections