# Objectif

**L’objectif** de ce travail est de **valider l’attente des sous-objectifs** dans le cadre du cours :

1. Effectuer le développement d’applications Web transactionnelles. (Compétence 00SU)
   * 5 - Programmer la logique applicative du côté serveur.
   * 6 - Programmer la logique applicative du côté client.
   * 7 - Contrôler la qualité de l’application.
   * 9 - Rédiger la documentation.

# Mise en situation

Vous êtes **programmeur Web** et avez à faire la programmation d’un lecteur audio à l’aide de JavaScript pour en maximiser l’interactivité.

Vous devez **réaliser la programmation** du lecteur au complet selon les spécifications des **récits utilisateurs** et en respectant **les consignes.**

## Détails supplémentaires

1. Respecter les [bonnes pratiques de code](http://appweb.progwmj.ca/documentations/bonnes-pratiques/code)
2. Vous devez respecter les **récits utilisateurs fournis**
3. Vous devez respecter les **consignes énoncées** plus bas.
4. N’oubliez pas de remplir la **grille d’auto-évaluation Excel**!

# Récits utilisateurs

|  |  |
| --- | --- |
| En tant qu’utilisateur... | Précisions |
| 1. Je peux faire jouer une chanson. | Clique bouton Jouer => bouton Pause apparaît  Utiliser songPlayerControl |
| 1. Je peux mettre sur pause une chanson. | Clique bouton Pause => bouton Jouer apparaît  Utiliser songPlayerControl |
| 1. Je peux arrêter une chanson. | Clique bouton Stop => bouton jouer apparaît si en pause  Utiliser songPlayerControl |
| 1. Je peux lire le nom de la chanson et le nom de l’artiste. | Utiliser songPlayerDatas |
| 1. Je peux consulter la description de la chanson et de l’artiste en cours. | Utiliser songInfos |
| 1. Je peux voir la durée d’une chanson. | Temps au format 00:00  Utiliser songPlayerTime |
| 1. Je peux voir le temps courant d’une chanson en temps réel. | Temps au format 00:00  Utiliser songPlayerTime |
| 1. Je peux changer la chanson en cours. | Clique sur la liste des chansons  Utiliser songsList |
| 1. Je peux changer la chanson en cours pour la précédente ou la suivante | Utiliser songList |
| 1. Je peux jouer une chanson au hasard | Utiliser songList |

# Méthodologie

1. Prendre connaissance de tous les récits utilisateurs.
2. Prendre connaissance du code de départ.

Le fichier **app.vue** contient **6 composants** :

index.html

├─songPlayer

| ├─songPlayerDatas

| ├─songPlayerControls

| ├─songPlayerTime

├─songInfos

├─songsList

1. Afin de manipuler l'audio en JavaScript, de nouvelles connaissances sont nécessaires, plus particulièrement :

|  |  |
| --- | --- |
| [HTML Event Attributes](https://www.w3schools.com/tags/ref_eventattributes.asp)  [HTML Audio/Video DOM Reference](https://www.w3schools.com/tags/ref_av_dom.asp) | [Audio()](https://developer.mozilla.org/en-US/docs/Web/API/HTMLAudioElement/Audio)  [HTMLAudioElement](https://developer.mozilla.org/en-US/docs/Web/API/HTMLAudioElement/Audio) |
| [Json-server](https://www.freecodecamp.org/news/json-server-for-frontend-development/) | [Component Events](https://vuejs.org/guide/components/events.html) |

# Consignes

* Travail pratique **individuel (20% de la note)**
* Réalisé en-**dehors** **du** **cours et temps en classe**
* Durée approximative de **15-18 heures**.
* Le fichier de remise est un fichier compressé nommé : **appweb\_trpr01\_votrenom.zip.**
  + Les **fichiers complets** de votre site
  + **IMPORTANT** supprimer le dossier **node\_modules**
  + La **grille d’évaluation** au format **Excel dans le dossier doc du site**
* Remis sur **Léa**à la date mentionnée

# Critères d’évaluation (100 %)

## Qualité du code (40 %)

* **Utilisation correcte de Vue.js :** Application des concepts clés de Vue.js tels que les composants, les directives, et le système réactif.
* **Utilisation correcte de TypeScript :** Types bien définis pour les props, les états des composants, et les objets. Utilisation des avantages de TypeScript pour augmenter la sécurité et la maintenabilité du code.
* **Bonnes pratiques de programmation et code propre.** Exemples: noms significatifs, évitez la répétition, clarté et simplicité, respect des normes, etc.
* **Structure du projet :** Organisation logique des fichiers et des dossiers, utilisation appropriée des modules et des composants.
* **Gestion des états :** Bonne utilisation des états locaux du composant et réactivité.

# **Fonctionnalités (50%)**

* **Implémentation des fonctionnalités requises** : L'application doit implémenter toutes les fonctionnalités demandées dans les récits utilisateurs telles que la lecture, la pause, le changement de piste, etc. (10 récits \* 4 = 40%)
* **Gestion des erreurs** : L'application gère correctement les erreurs, par exemple, lors du chargement des fichiers audio ou en cas d'absence de fichier. (10 récits \* 1 = 10%)

## Interface utilisateur (10 %)

* **Bogues et performances :** Aucun bogue, aucune erreur d'exécution et aucun problème de performance.
* **Qualité de l'interface** : convivialité, notifications à l'utilisateur, alertes, messages de confirmation, etc. Ex : Message d’erreur si la chanson n’est pas trouvée

## Respect des consignes (-10 %)

* Les consignes du travail sont respectées

***La grille d’évaluation est fournie sur Léa.***