Cégep André-Laurendeau

Travail synthèse
« Portail de jeux »

420-245-AL
Programmation web

Note: / 100 (équipe)
Note: / 75 (seul)

À remettre au plus tard aux dates indiquées dans LÉA

Table des matières

1. Performance finale attendue

« À partir d'un devis décrivant les exigences minimales à respecter pour la création d'un site web de petite envergure, l'étudiant développe et déploie un site web complet, fonctionnel, interactif et ergonomique qui répond aux exigences ».

En réalisant ce travail synthèse, l'étudiant exploitera différentes compétences acquises pendant la session :

- Utilisation du langage JavaScript :
  - Structures de contrôles
  - Fonctions
  - Tableaux
  - Programmation orientée-objet par prototype
  - Accès/Modification du DOM
  - Gestion d'évènements
  - Animations 2D
- Utilisation de l'API "Web Storage".
- Design adaptatif avec la librairie Bootstrap.
- Utilisation de la librairie jQuery.

2. Directives importantes

- Ce travail final peut être réalisé **seul ou en équipe de deux** (2). Les « clones » ou les copies trop semblables se verront attribuer la note 0 (plagiat).
- Le code source de départ est disponible dans Moodle.
- À remettre:
  - Dans **LÉA** : fichier **.zip** qui contient l'ensemble du projet (html, css, js, images, etc).
    - En **4 remises distinctes**, pour assurer un meilleur suivi du projet.
    - Voir les dates dans la section 5 de l'énoncé.
  - Sur **cal-info.org** (remise finale seulement)
    - Projet déployé sous le dossier « **tp-synthese** », puis **protégé** en changeant les droits d'accès au dossier.
- La **qualité du code** est importante. Après une année en programmation, vous devriez remettre du code en appliquant les standards d'écriture (indentation, commentaires, noms des identificateurs, paramètres des fonctions, etc.).
- La **qualité du français** est aussi un aspect à ne pas négliger (dans le code et dans le texte affiché dans les pages web). Prenez soin d'écrire des phrases bien structurées, en évitant les fautes d'orthographes, de syntaxe et de grammaire. Une pénalité de 10% pourra être retirée pour une mauvaise qualité du français.
- **Testez** votre code **au fur et à mesure** du développement, une fonctionnalité à la fois.
- N'oubliez pas que ce travail fait partie de l'évaluation certificative du cours, et qu'il y a un **seuil de réussite obligatoire spécifiquement pour ce travail**. Un étudiant qui obtient une note inférieure à 50% pour ce travail aura automatiquement un échec dans le cours, peu importe sa note globale.

3. Mise en situation

Pour ce travail final, on vous demande de compléter le code source de départ pour développer un portail de jeux.

Le portail comporte essentiellement 2 pages : la page d'authentification/inscription des abonnés, puis la page du catalogue des jeux.

Une troisième page sera affichée lors du lancement du jeu "Sauve qui peut", dans laquelle sera implémenté un jeu 2D codé en JavaScript.

Voici les principaux développements que vous devrez réaliser au cours du projet.

Page	Développement	Statut
Accueil / Catalogue / Jeu	Développer une interface adaptative ("responsive") avec le framework "Bootstrap".	À faire
Accueil	Valider les informations d'un futur abonné (nom d'utilisateur, pseudonyme, mot de passe) et afficher des messages d'erreur spécifiques selon la situation.	À faire
Accueil	Valider les informations d'authentification d'un abonné qui revient sur le portail.	À faire (si en équipe)
Jeu	Mettre en place la logique de déplacement du "personnage" (acteur), à partir de commandes entrées au clavier par le joueur.	À faire
Jeu	Mettre en place la création et le déplacement des ennemis.	Code fourni.
Jeu	Gérer les collisions entre le personnage et les ennemis.	À faire
Jeu	Mettre en place le chronomètre, qui servira aussi de donnée source pour gérer les "records".	À faire (si en équipe)
Jeu	Gérer les records (top 10).	À faire (si en équipe)
Accueil / Catalogue / Jeu	Améliorer l'ergonomie en gérant les focus et l'état de disponibilité des différents contrôles (boutons, liens, etc), selon le contexte.	À faire
Jeu	Mettre en place la possibilité de faire une "Pause" pendant la partie, puis de reprendre ensuite.	Bonus

Votre tâche consiste donc à compléter le code de départ fourni, tout en respectant une série d'exigences fonctionnelles et techniques qui sont décrites dans la prochaine section de cet énoncé.

4. Exigences fonctionnelles et techniques

Voici différentes **exigences** qui doivent être implémentées dans la solution finale. En plus de ces descriptions détaillées, vous pouvez également visionner la présentation vidéo (voir le lien dans Moodle).

4.1. Exigences générales

1. Le "look" final du site est professionnel.
2. Les 3 pages sont réalisées avec le "framework" Bootstrap.
3. Le site est adaptatif ("responsive"). Il s'adapte donc aux différentes tailles d'écran pour que les contenus soient toujours bien affichés et que le tout soit facilement lisible.
   N.B. : Pour ce faire, du code CSS est déjà fourni en partie dans les fichiers de départ.
4. Chaque page comporte un contenu spécifique (propre à cette page), mais aussi un contenu commun. La partie commune comprend :
   a. Un en-tête, avec image et titre, pour identifier le portail.
   b. Un menu pour naviguer entre les 3 pages.
   c. Une zone pour identifier si le visiteur est authentifié, et dans ce cas, on affiche son pseudonyme et un élément permettant de se déconnecter.
   d. Un pied de page pour identifier l'auteur et la date de création du portail.
5. Le code HTML/CSS/JavaScript du site est réparti dans plusieurs fichiers, vous devez respecter cette structure. Le code de départ contient déjà tous ces fichiers et les liens (<script> et <link>) sont déjà définis dans la section <head> des 3 pages HTML.
   a. Les 3 pages web sont développées dans les fichiers "index.html" (page d'accueil), "catalogue.html" (page du catalogue) et "sauveQuiPeut.html" (page du jeu développé).
   b. En complément au framework Bootstrap, vous aurez probablement un peu de code CSS personnel à développer, il sera ajouté dans le fichier "mesStyles.css".
   c. Chaque page web a son script JavaScript spécifique : "index.js", "catalogue.js" et "sauveQuiPeut.js".
   d. Certaines fonctionnalités plus spécifiques à certains traitements sont isolées dans des scripts JavaScript spécifiques :
      i) La gestion des abonnées est codée dans le script "gestionAbonnes.js".
      ii) La gestion du plan de jeu est codée dans le script "gestionPlan.js".
      iii) La gestion de l'acteur est codée dans le script "gestionActeur.js".
      iv) La gestion du chronomètre est codée dans le script "gestionChronometre.js".
      v) La gestion des records sera codée dans le script "gestionRecords.js".
      vi) La gestion des ennemis est déjà codée (en grande partie) dans le script "gestionEnnemis.js". Il ne manque que le traitement des collisions que vous y ajouterez.
      vii) Les traitements communs à plusieurs pages sont codés dans le script "commun.js".

4.2. Page d'accueil

6. Les contenus spécifiques suivants sont présents dans cette page :
    a. Une 1re section pour s'authentifier (nom d'utilisateur et du mot de passe).
    b. Une 2e section pour s'inscrire (au minimum un nom d'utilisateur, un pseudonyme et un mot de passe avec sa confirmation).
    c. Des boutons d'actions pour confirmer les 2 opérations (authentification et inscription).
    d. Une boîte de dialogue modale pour afficher les messages d'erreur (cachée initialement).
7. Les 2 sections ne sont pas visibles simultanément. Lorsque l'on visualise la section d'authentification, la section d'inscription est cachée, et l'inverse est aussi vrai.
8. Lors de la confirmation de l'inscription, les validations suivantes sont effectuées, et un message spécifique est affiché en cas d'erreur (apparition dans une fenêtre de dialogue modale) :
    a. Le nom d'utilisateur et le pseudonyme sont formés de 3 à 10 caractères, qui sont uniquement des lettres (minuscules ou majuscules, sans accents) ou des chiffres.
    b. Le mot de passe est formé uniquement de lettres minuscules non accentuées. De plus, la première et la dernière lettre doivent être identiques.
    c. Les contenus des champs du mot de passe et de sa confirmation sont identiques.
    d. Le nom d'utilisateur ne correspond pas au nom d'un autre abonné déjà inscrit.
    e. Les espaces non significatifs au début ou à la fin des champs de saisie ne sont pas considérés lors des validations et ils ne sont pas enregistrés.
9. Lors de la confirmation de l'authentification, les validations suivantes sont effectuées, et un message spécifique est affiché en cas d'erreur (apparition dans une fenêtre de dialogue modale) :
    a. La combinaison nom d'utilisateur et mot de passe correspond aux informations d'un abonné inscrit.
10. Si aucune erreur n'est détectée lors des validations :
    a. L'utilisateur est automatiquement authentifié et la page du catalogue est affichée.
    b. Si c'est une inscription, les informations de l'utilisateur sont stockées dans le "Local Storage). Voir les détails dans l'exigence 11b.
11. Gestion des abonnés :
    a. La gestion des abonnés est implémentée avec la POO par prototype, plus spécifiquement à l'aide d'une fonction constructeur pour créer un abonné.
        i) Un objet "Abonne" est défini au minimum avec 3 propriétés : son nom, son pseudonyme et son mot de passe
        ii) Le constructeur contiendra également au moins 3 méthodes (getter) pour obtenir les valeurs de ses propriétés.
        iii) Une autre méthode devra permettre de sauvegarder l’abonné dans le "Local Storage" (voir détail dans le point 11b).

b. Les informations des abonnés sont sauvegardées dans le "Local Storage" du navigateur.

   i) L'information d'un abonné est sauvegardée sous une clé formée du préfixe "ABONNE_", suivi du nom d'utilisateur de cet abonné.

   ii) Le contenu sauvegardé dans le "storage" est une transformation en chaîne de caractères de l'objet JSON représentant les propriétés de cet abonné.

c. L'état "authentifié" ou "non authentifié" du visiteur du site est géré en sauvegardant l'information dans le "Session Storage" du navigateur.

   i) Lorsqu'un visiteur est authentifié, une information est sauvegardée temporairement sous la clé "ABONNE_AUTHENTIFIE", et elle contient simplement la transformation en chaîne de caractères de l'objet JSON représentant cet abonné.

   ii) Lorsqu'un utilisateur authentifié se déconnecte, l'entrée du "Session Storage" est simplement supprimée.

d. Voici quelques propositions de fonctions JavaScript à développer pour la gestion des abonnés :

   i) La fonction constructeur, avec ses propriétés et ses méthodes.

   ii) validerNomOuPseudonyme(…) : Reçoit une chaîne en paramètre (un nom ou un pseudonyme) et retourne un booléen indiquant s'il est valide.

   iii) validerNouveauMotDePasse(…) : Reçoit une chaîne en paramètre (un mot de passe) et retourne un booléen indiquant s'il est valide.

   iv) validerNouvelAbonne(…) : Reçoit une chaîne en paramètre (le nom d'un futur abonne), et retourne un booléen indiquant si ce nom est autorisé (donc pas équivalent à celui d'un abonné déjà inscrit).

   v) validerMotDePasseIdentiques(…) : Reçoit deux chaînes en paramètre et retourne un booléen indiquant qu'elles sont identiques.

   vi) validerMotDePasse(…) : Reçoit un objet "Abonne" et une chaîne correspondant au mot de passe entré par l'utilisateur, et retourne un booléen indiquant si le mot de passe est conforme au mot de passe de cet abonné.

   vii) estAuthentifie(…) : Retourne un booléen indiquant si un utilisateur est actuellement authentifié dans l'application.

   viii) setAuthentifie(…) : Reçoit ou bien un objet "Abonne" en paramètre et entraîne alors les traitements requis à la suite de son authentification (ajout de cet état dans le "Storage" et navigation vers la page du catalogue), ou bien la valeur "false" et cela entraîne alors les traitements requis pour déconnecter l'utilisateur actuellement authentifié.

   ix) getPseudo(…) : Retourne le pseudonyme de l'abonné actuellement authentifié.

   x) afficherPseudo(…) : Met à jour l'interface (la page affichée), pour y ajouter le pseudonyme et le bouton de déconnexion si l'utilisateur actuel est authentifié ou pour les retirer dans le cas contraire.

   xi) getAbonne(…) : Reçoit un nom d'utilisateur en paramètre et retourne un objet JavaScript "Abonne" qui contient les informations de cet abonné (lues dans le "Local Storage").

4.3. Page du catalogue

12. Elle présente la liste des jeux sous la forme de fiches (card Bootstrap).
13. Chaque fiche présente au minimum le nom du jeu et l'image.
14. Chaque fiche est un lien "potentiel" vers une nouvelle page HTML qui s'affiche pour ce jeu.
15. Lors du clic sur le lien (ou l'image) menant à un jeu, un message d'avertissement est affiché si l'utilisateur n'est pas authentifié, et le la page du jeu n'est donc pas affichée.
16. Voici quelques propositions de fonctions JavaScript à développer pour cette partie du code :
    a. genererCatalogue(…) : Génère, à partir du contenu du tableau "LISTE_JEUX", le contenu HTML dynamique pour présenter le catalogue, et l'insère dans la bonne section de la page "catalogue.html".

4.4. Page du jeu

17. La partie spécifique de cette page contient les éléments suivants :
    a. Le plan de jeu (déjà défini dans le code HTML/CSS).
    b. Une zone pour afficher le chronomètre.
    c. Un composant pour accéder aux records (10 meilleurs temps).
    d. Un composant pour débuter une nouvelle partie (bouton ou autre).
    e. Un composant pour mettre une partie en pause (bonus du travail pour ceux/celles qui le font en équipe de 2).
18. Au départ et lors du recommencement, le plan de jeu est réinitialisé.
    a. Le plan de jeu est vidé de son contenu (code déjà présent, appel de la méthode "viderPlan()" de l'objet "plan").
    b. La taille de l'acteur est calculée pour qu'il occupe 5% de la largeur du plan de jeu.
    c. L'acteur est positionné au centre du plan de jeu (horizontalement et verticalement), donc en fonction de la taille actuelle du plan de jeu et de la taille de l'acteur.
    d. Le chronomètre est remis à "0:00:000".
    e. Les ennemis sont réinitialisés (code déjà présent, appel de la fonction "genererEnnemis").
19. Au démarrage d'une partie :
    a. Le chronomètre démarre et affiche la durée écoulée pendant le jeu en minutes, secondes et millisecondes.
    b. Le plan de jeu s'anime.
        i) Les ennemis se déplacent (code déjà présent, appel de la fonction "animerEnnemis(…)".
        ii) Les mouvements de l'acteur peuvent être contrôlés par le joueur.
20. À la fin d'une partie (perdue à cause de la collision avec un ennemi) :
    a. L'image ou la couleur de l'acteur change.
    b. Le chronomètre s'arrête, la durée écoulée reste affichée.
    c. L'acteur et les ennemis s'immobilisent.
    d. Les mouvements de l'acteur ne peuvent plus être contrôlés par le joueur.

e. Les records sont affichés si la durée totale fait partie des 10 meilleurs scores.

21. Déplacement de l'acteur :
    a. Le joueur contrôle l'acteur avec le clavier.
    b. Les 4 touches de direction (flèches gauche, droite, haut et bas) initient le mouvement dans la direction correspondante (évènement JavaScript "keydown"), qu'elles soient activées sur le clavier par les flèches ou via le pavé numérique en mode "Verr.Num." désactivé).
    c. Les 4 touches de direction diagonales ("Home", "PgUp", "Fin" et "PgDn") initient le mouvement dans les 4 directions respectives suivantes : "Supérieure gauche", "Supérieure droite", "Inférieure gauche" et "Inférieure droite".
    d. L'espace ou la touche centrale du pavé numérique immobilise l'acteur, tout en laissant les ennemis en mouvement.
    e. La touche "Echap" (Esc) met le jeu en pause (bonus du Travail synthèse)
    f. L'acteur s'immobilise automatiquement lorsqu'il atteint l'une des 4 frontières du plan de jeu.
    g. L'acteur s'immobilise également lorsqu'il entre en collision avec un ennemi, provoquant ainsi la fin de la partie.
22. Gestion des collisions :
    a. Le code pour gérer les collisions doit être ajouté dans le fichier "gestionEnnemis.js".
    b. À chaque milliseconde, lorsque les ennemis sont en mouvement, les collisions sont vérifiées (on vérifie si l'acteur est en collision avec chaque ennemi).
    c. Une collision entre l'acteur et un ennemi survient lorsque la distance entre le centre de la sphère de l'acteur ($Acteur_{Centre\text{-}X}$, $Acteur_{Centre\text{-}Y}$) et le centre de la sphère de l'ennemi ($Ennemi_{Centre\text{-}X}$, $Ennemi_{Centre\text{-}Y}$) à laquelle on soustrait le rayon de ces 2 sphères est inférieure à 0. Autrement dit, si la distance calculée suivante est inférieure ou égale à 0 c'est qu'il y a une collision. :

$$distance = \sqrt{(Acteur_{Centre-X} - Ennemi_{Centre-X})^2 + (Acteur_{Centre-Y} - Ennemi_{Centre-Y})^2} - Acteur_{Rayon} - Ennemi_{Rayon}$$

    d. Dès qu'une collision est détectée entre l'acteur et un ennemi, la partie se termine.
23. Gestion du chronomètre
    a. Le chronomètre doit être implémenté avec l'utilisation des objets "Date" de JavaScript, en calculant la durée écoulée entre le début de la partie et le moment présent.
    b. Le chronomètre doit afficher la durée écoulée en minutes, secondes et millisecondes, et doit être réaffiché automatiquement à toutes les millisecondes.
24. Gestion des records
    a. À la fin d'une partie, si le joueur a obtenu un score faisant partie des 10 meilleurs (10 durées totales les plus élevées) :
        i) On conserve son record et on affiche la liste des 10 meilleurs temps.
        ii) Les résultats affichés présentent au minimum l'ordre séquentiel du record, le pseudonyme de l'abonné et le temps obtenu lors de ce record.

b. Les informations des records sont sauvegardées dans le "Local Storage" du navigateur.

   i) L'information des records est sauvegardée sous une seule clé, qui contient les données des 10 meilleures parties (pseudonyme du joueur et temps obtenu).

   ii) Ces données sont stockées dans cette clé du "Local Storage" en ordre séquentiel (un tableau trié d'objets JavaScript transformé en String serait donc une bonne approche pour stocker ces informations).

c. Voici quelques propositions de fonctions JavaScript à développer pour la gestion des records :

   i) getRecords(…) : Retourne un tableau trié d'objets JavaScript qui contient les informations des 10 meilleures parties (lues du "Local Storage").

   ii) majRecords(…) : Reçoit une durée en paramètre, puis vérifie si cette durée est un résultat qui se classerait parmi les 10 meilleurs. Si c'est le cas, le "Local Storage" est mis à jour avec ce nouveau record.

   iii) AfficherRecords(…) : Provoque l'affichage des records. Sera appelé par le composant "manuel" de la page ou lors d'une partie qui résulte en un temps parmi les 10 premiers.

4.5. Exigences d'ergonomie

25. Des touches de raccourcis permettent de naviguer entre les onglets et d’activer les boutons d’action.
26. Sur les pages de saisies d'informations, le focus est placé automatiquement sur le champ de saisie le plus approprié selon le contexte.
27. Lors de la détection d’erreurs, le champ en erreur est mis en évidence.
28. L'état (disponible-actif/non disponible-inactif) des différents champs/boutons/onglets/liens est mis à jour selon le contexte, dans les 3 pages du portail.

5. Étapes à suivre pour la réalisation du TP-synthèse

1. Lire l'énoncé du TP au complet.
2. Récupérer le code source de départ dans Moodle et l'explorer pour s'y familiariser.
3. Développer la structure générale des 3 pages avec Bootstrap (Accueil : "index.html", Catalogue : "catalogue.html", Jeu : "sauveQuiPeut.html"), en vous assurant que l'affichage est adéquat peu importe la largeur de l'écran sur lequel elles seront affichées.
    a. Commencer par les éléments communs à toutes les pages :
        1. En-tête
        2. Menu de navigation entre les pages
        3. Pied de page
    b. Poursuivre avec les éléments spécifiques à chaque page.
    c. N.B. Initialement, certains trucs seront codés en dur ("hardcodé") :
        1. L'affichage du pseudonyme et du bouton de déconnexion sur toutes les pages.
        2. L'affichage des fiches (card) de différents jeux sur la page du catalogue.
        3. L'affichage du chronomètre sur la page du jeu.
        4. L'affichage de l'acteur centré verticalement sur la page du jeu.

REMISE PARTIELLE #1 dans LÉA (vendredi 22 avril)

4. Développer la logique d'inscription et d'authentification dans la page d'accueil (gestion des abonnés).
    a. D'abord les différentes fonctions de validation.
    b. Ensuite la logique d'enchaînement des appels lors du clic des boutons dans la page d'accueil.
    c. Ensuite la fonction constructeur pour créer un abonné.
    d. Et enfin la persistance d'un nouvel abonne dans le "Local Storage" ou d'un abonné connecté dans le "Session Storage".

REMISE PARTIELLE #2 dans LÉA (dimanche 1er mai)

5. Générer le catalogue dynamiquement à partir du tableau LISTE_JEUX.
6. Mettre en place l'acteur dans le plan de jeu et gérer son déplacement à l'intérieur du plan avec les contrôles clavier demandés.
7. Ajouter la gestion des collisions, incluant l'arrêt des animations et la désactivation des contrôles lorsque la partie se termine.

REMISE PARTIELLE #3 dans LÉA (mardi 10 mai)

8. Ajouter la gestion du temps, incluant l'affichage du chronomètre.
9. Ajouter la gestion des records.
10. Ajouter les traitements liés à l'ergonomie (voir section 4.5).
11. BONUS (défi) : Ajouter le processus de mise en pause du jeu
12. Tester à fond chacune des fonctionnalités (voir le plan de tests proposé dans la section 6).
13. Déployer le portail sur le serveur cal-info.org et retester le tout dans cet environnement.
14. Désactiver l'accès au code sur le serveur en changeant les droits d'accès au dossier racine de l'application.

REMISE FINALE (vendredi 20 mai)

6. Tests fonctionnels

Voici une proposition de plan de tests pour tester votre application.

#Cas	Page	Fonctionnalité	Description du test	Résultat attendu
G1	Accueil/ Catalogue/ Jeu	Pages "responsives"	Redimensionnement de la fenêtre	Les informations restent bien présentées pour les catégories de largeur d'écran de Bootstrap
G2	Accueil/ Catalogue/ Jeu	Déconnexion	L'abonné clique sur le composant de déconnexion lorsqu'aucune partie n'est en cours	• Les informations de l'utilisateur authentifié sont retirées du "Session Storage" • La page d'accueil est affichée • Les champs de la page d'accueil et les boutons redeviennent disponibles • Le focus est placé sur le 1er champ Le pseudonyme et le bouton de déconnexion ne sont pas affichés
G3	Accueil/ Catalogue/	Affichage du pseudonyme et du bouton de déconnexion	L'utilisateur n'est pas authentifié	Le pseudonyme et le bouton de déconnexion ne sont pas affichés
G4	Accueil/ Catalogue/ Jeu	Affichage du pseudonyme et du bouton de déconnexion	L'utilisateur est authentifié	Le pseudonyme et le bouton de déconnexion sont affichés
A1	Accueil	Validation du nom de l'utilisateur (inscription)	Nom d'utilisateur ne répond pas aux exigences (nombre de caractères et type de caractères)	• Message d'erreur spécifique pour cette erreur. • Mise en évidence du champ et focus sur le champ.
A2	Accueil	Validation du nom de l'utilisateur (inscription)	Nom d'utilisateur identique à un abonné déjà inscrit	• Message d'erreur spécifique pour cette erreur. • Mise en évidence du champ et focus sur le champ.
A3	Accueil	Validation du pseudonyme (inscription)	Pseudonyme ne répond pas aux exigences (nombre de caractères et type de caractères)	• Message d'erreur spécifique pour cette erreur. • Mise en évidence du champ et focus sur le champ.
A4	Accueil	Validation du mot de passe (inscription)	2 mots de passe non identiques	• Message d'erreur spécifique pour cette erreur. • Mise en évidence du 1er mot de passe et focus sur le champ.
A5	Accueil	Validation du mot de passe (inscription)	Mot de passe ne répond pas aux exigences (nombre de caractères et type de caractères et 1ère et dernière lettre)	• Message d'erreur spécifique pour cette erreur. • Mise en évidence du 1er mot de passe et focus sur le champ.

#Cas	Page	Fonctionnalité	Description du test	Résultat attendu
A6	Accueil	Inscription valide	Toutes les validations sont réussies	• Les informations sont ajoutées dans une nouvelle clé du "Local Storage" • Les champs de la page d'accueil et les boutons deviennent non disponibles • La page du catalogue est affichée • Le pseudonyme est affiché avec le composant permettant la déconnexion
A7	Accueil	Validation du nom de l'utilisateur (authentification)	Nom d'utilisateur ne correspond pas à un abonné inscrit	• Message d'erreur spécifique pour cette erreur. • Mise en évidence du nom et focus sur le champ.
A8	Accueil	Validation du mot de passe de l'utilisateur (authentification)	Mot de passe ne correspond pas au mot de passe de l'abonné inscrit	• Message d'erreur spécifique pour cette erreur. • Mise en évidence du mot de passe et focus sur le champ.
A9	Accueil	Authentification valide	Toutes les validations sont réussies	• Les informations de l'utilisateur authentifié sont ajoutées temporairement dans le "Session Storage" • Les champs de la page d'accueil et les boutons deviennent non disponibles • La page du catalogue est affichée • Le pseudonyme est affiché avec le composant permettant la déconnexion • Le focus est placé sur le lien permettant de démarrer le jeu « Sauve Qui peut »
A10	Accueil	Clic du lien vers la page du catalogue	L'utilisateur clique sur le lien permettant d'afficher la page du catalogue	• La page demandée s'affiche • Le lien vers la page d'accueil est actif tandis que celui vers la page du catalogue est inactif • Le focus est placé sur le lien permettant de démarrer le jeu « Sauve Qui peut »
C1	Catalogue	Clic du lien vers la page d'accueil	L'utilisateur clique sur le lien permettant d'afficher la page d'accueil	• La page demandée s'affiche, et c'est le formulaire d'authentification qui est affiché (le formulaire d'inscription est caché) • Les champs de saisie et les boutons d'action sont disponibles uniquement si l'utilisateur n'est pas authentifié. • À l'inverse, le pseudonyme et le composant de déconnexion sont affichés uniquement si l'utilisateur est authentifié. • Le lien vers la page du catalogue est actif tandis que celui vers la page d'accueil est inactif
C2	Catalogue	Clic du lien vers le jeu	L'utilisateur n'est pas authentifié	• Message d'erreur indiquant que l'authentification est requise

#Cas	Page	Fonctionnalité	Description du test	Résultat attendu
C3	Catalogue	Clic du lien vers le jeu	L'utilisateur est authentifié	• La page du jeu est affichée • Le pseudonyme est affiché avec le composant permettant la déconnexion • Les ennemis et l'acteur sont positionnés • Le chronomètre est à 0 • Le focus est placé sur le composant permettant de débuter la partie
J1	Jeu	Déconnexion	L'abonné clique sur le composant de déconnexion lorsqu'une partie est en cours	• Une confirmation est demandée pour terminer la partie et si la réponse est positive, alors… • Les informations de l'utilisateur authentifié sont retirées du "Session Storage" • La page d'accueil est affichée • Les champs de la page d'accueil et les boutons redeviennent disponibles • Le focus est placé sur le 1er champ • Le pseudonyme et le bouton de déconnexion ne sont pas affichés
J2	Jeu	Changement de page	L'abonné clique sur le lien vers la page d'accueil ou la page du catalogue lorsqu'une partie est en cours	• Une confirmation est demandée pour terminer la partie et si la réponse est positive, alors… • La page demandée est affichée • Le pseudonyme et le bouton de déconnexion restent affichés
J3	Jeu	Début de partie	Le composant permettant de débuter la partie est actionné	• Le composant en question devient non disponible • Les ennemis du plan de jeu s'animent • L'acteur peut être animé avec les touches du clavier • Le chronomètre est démarré et est mis à jour chaque milliseconde
J4	Jeu	Fin de partie	Suite à une collision entre l'acteur et un ennemi	• Le composant pour réinitialiser une nouvelle partie redevient disponible • Les ennemis du plan de jeu s'arrêtent • L'acteur s'arrête et change de "look" • L'acteur ne peut plus être animé avec les touches du clavier • Le chronomètre est arrêté • La fenêtre modale des records s'affiche et le focus est placé sur le bouton permettant de refermer cette fenêtre (uniquement si le temps obtenu fait partie des 10 meilleurs résultats)

#Cas	Page	Fonctionnalité	Description du test	Résultat attendu
J5	Jeu	Entrée en pause	Suite à l'activation du composant pour mettre le jeu en pause	• Les ennemis du plan de jeu s'arrêtent • L'acteur s'arrête • L'acteur ne peut plus être animé avec les touches du clavier • Le chronomètre est arrêté • Le focus est placé sur le composant permettant de continuer la partie
J6	Jeu	Sortie de pause	Suite à l'activation du composant pour mettre le jeu en pause	• Les ennemis du plan de jeu reprennent leurs déplacements • L'acteur peut de nouveau être animé avec les touches du clavier • Le chronomètre redémarre
J7	Jeu	Déplacement vertical	Suite à la pression de la touche flèche vers le haut	L'acteur se déplace vers le haut et s'arrête lors qu'il touche la frontière supérieure du plan de jeu ou s'il entre en collision avec un ennemi
J8	Jeu	Déplacement vertical	Suite à la pression de la touche flèche vers le bas	L'acteur se déplace vers le bas et s'arrête lors qu'il touche la frontière inférieure du plan de jeu ou s'il entre en collision avec un ennemi
J9	Jeu	Déplacement horizontal	Suite à la pression de la touche flèche vers la gauche	L'acteur se déplace vers la gauche et s'arrête lors qu'il touche la frontière gauche du plan de jeu ou s'il entre en collision avec un ennemi
J10	Jeu	Déplacement horizontal	Suite à la pression de la touche flèche vers la droite	L'acteur se déplace vers la droite et s'arrête lors qu'il touche la frontière droite du plan de jeu ou s'il entre en collision avec un ennemi
J11	Jeu	Déplacement diagonal	Suite à la pression de la touche "Home"	L'acteur se déplace en montant vers la gauche et s'arrête lors qu'il touche la frontière gauche ou supérieure du plan de jeu ou s'il entre en collision avec un ennemi
J12	Jeu	Déplacement diagonal	Suite à la pression de la touche "PgUp"	L'acteur se déplace en montant vers la droite et s'arrête lors qu'il touche la frontière droite ou supérieure du plan de jeu ou s'il entre en collision avec un ennemi
J13	Jeu	Déplacement diagonal	Suite à la pression de la touche "Fin"	L'acteur se déplace en descendant vers la gauche et s'arrête lors qu'il touche la frontière gauche ou inférieure du plan de jeu ou s'il entre en collision avec un ennemi
J14	Jeu	Déplacement diagonal	Suite à la pression de la touche "PgDn"	L'acteur se déplace en descendant vers la droite et s'arrête lors qu'il touche la frontière droite ou inférieure du plan de jeu ou s'il entre en collision avec un ennemi

#Cas	Page	Fonctionnalité	Description du test	Résultat attendu
J15	Jeu	Arrêt du déplacement	Suite à la pression de l'espace ou la touche centrale du pavé numérique	L'acteur s'immobilise
J16	Jeu	Affichage des records	Suite à l'activation du composant pour visualiser les records	La fenêtre modale des records s'affiche et le focus est placé sur le bouton permettant de refermer cette fenêtre

7. Aperçu des pages

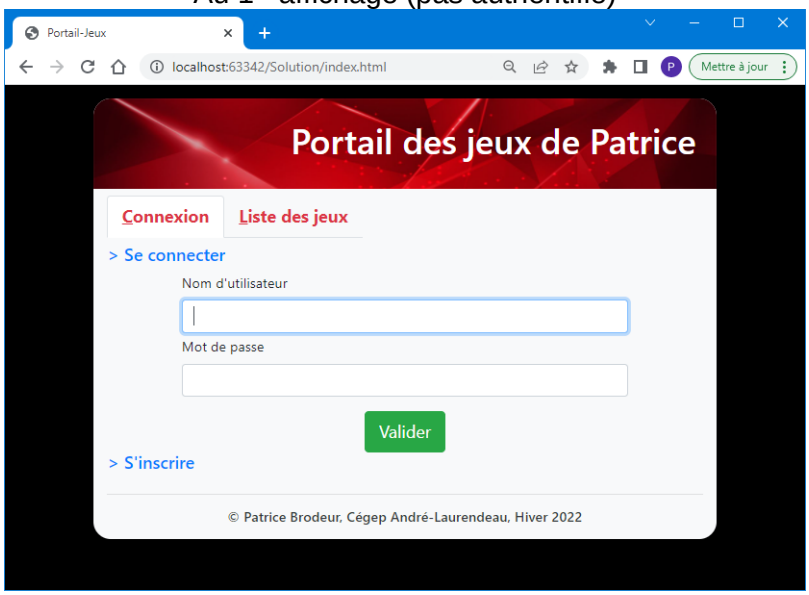
Voici quelques copies d'écran d'une solution parmi d'autres. Elles ne sont là que pour vous guider, mais l'objectif n'est pas de les reproduire, bien au contraire. Faites usage de votre créativité pour concevoir une interface personnalisée.

Vous devez donc produire votre propre portail, tout en respectant les exigences énumérées dans la section 4 :

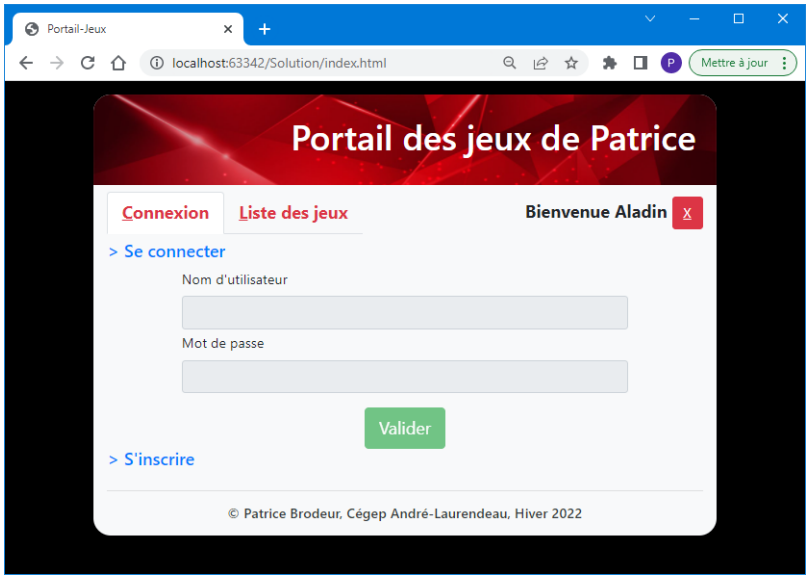
- Vos couleurs
- Vos images
- Votre propre "design"

A) Page d'accueil

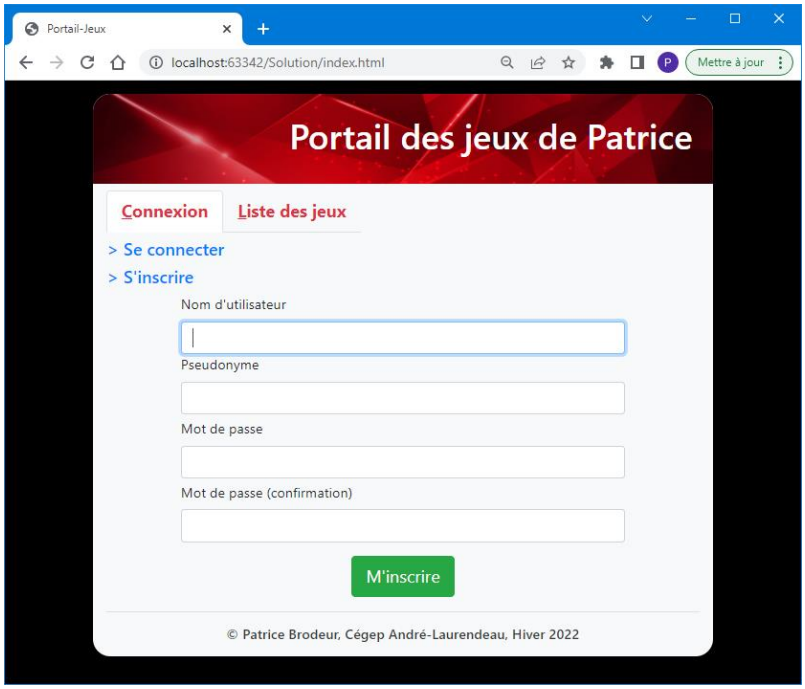
Au 1er affichage (pas authentifié)



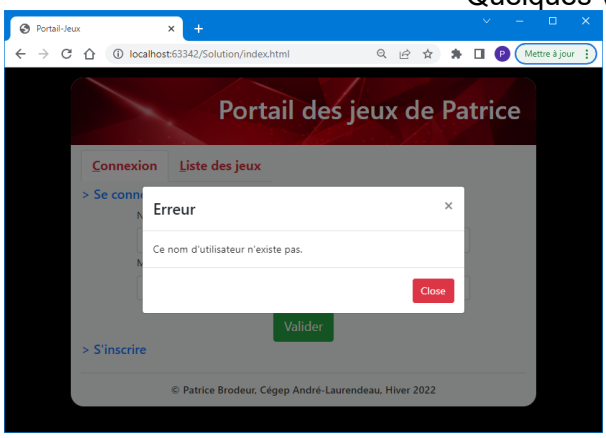
Si authentifié

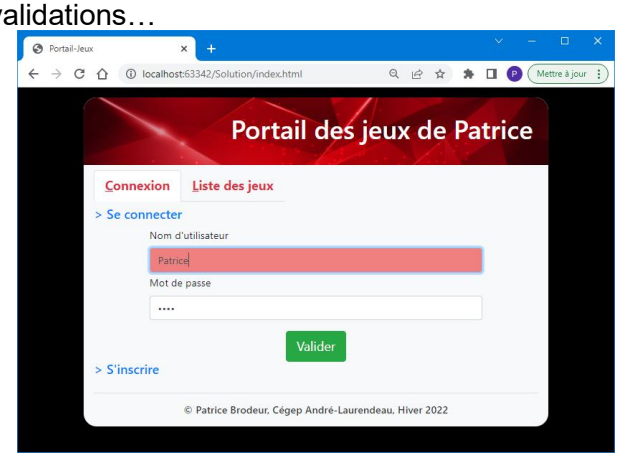


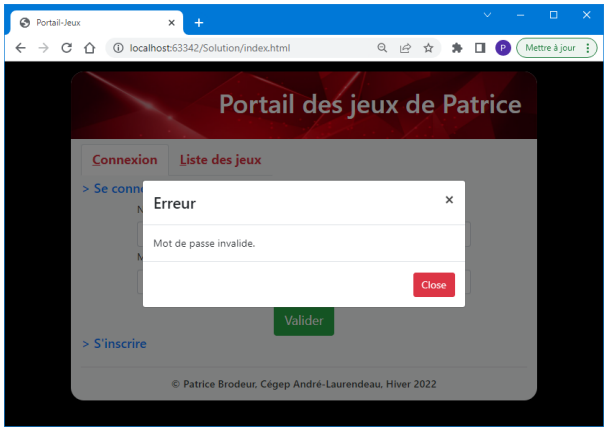
Au clic du lien « S'inscrire

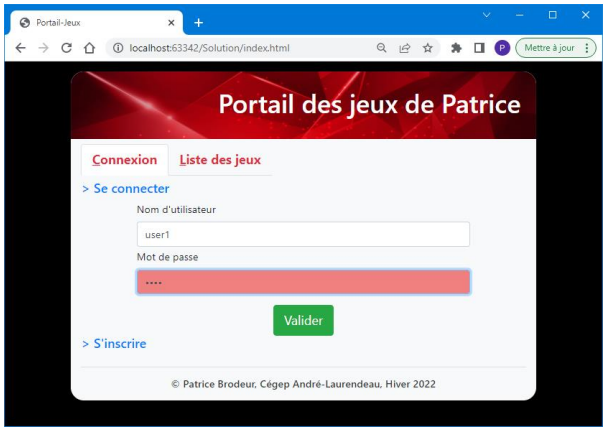


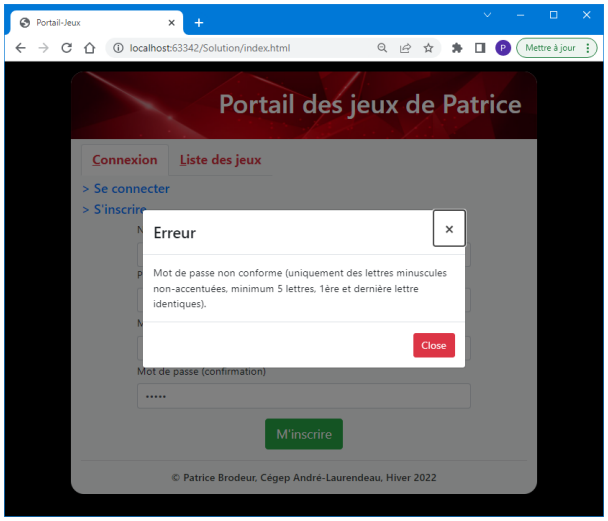
Quelques validations…

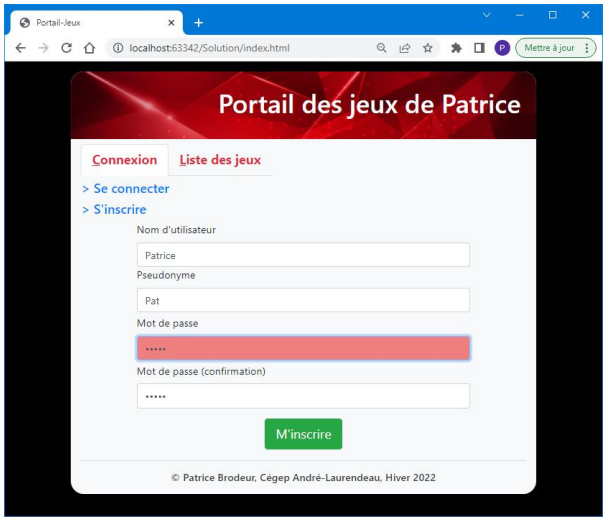






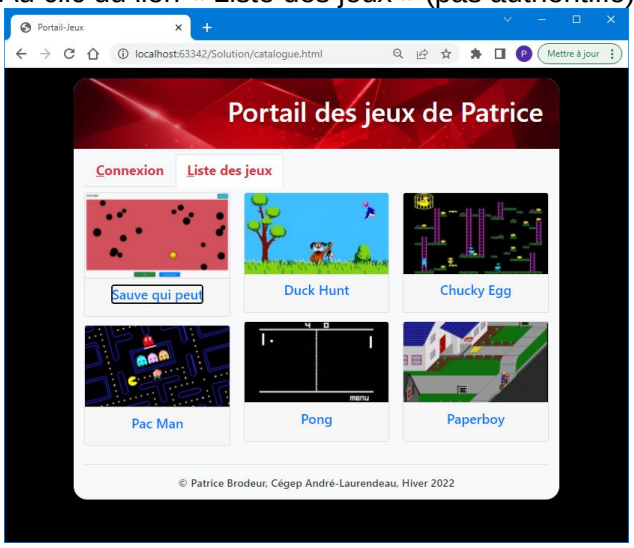



Portail-Jeux
localhost:63342/Solution/index.html
Mettre à jour
Portail des jeux de Patrice
Connexion
Liste des jeux
> Se connecter
> S'inscrire
Erreur
Mot de passe non conforme (uniquement des lettres minuscules non-accentuées, minimum 5 lettres, 1ère et dernière lettre identiques).
Close
Mot de passe (confirmation)
.....
M'inscrire
© Patrice Brodeur, Cégep André-Laurendeau, Hiver 2022

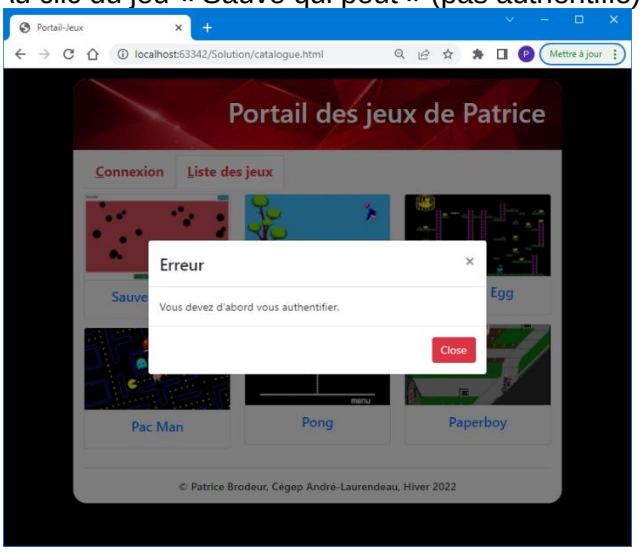

Portail-Jeux
localhost:63342/Solution/index.html
Mettre à jour
Portail des jeux de Patrice
Connexion
Liste des jeux
> Se connecter
> S'inscrire
Nom d'utilisateur
Patrice
Pseudonyme
Pat
Mot de passe
.....
Mot de passe (confirmation)
.....
M'inscrire
© Patrice Brodeur, Cégep André-Laurendeau, Hiver 2022

B) Page du catalogue

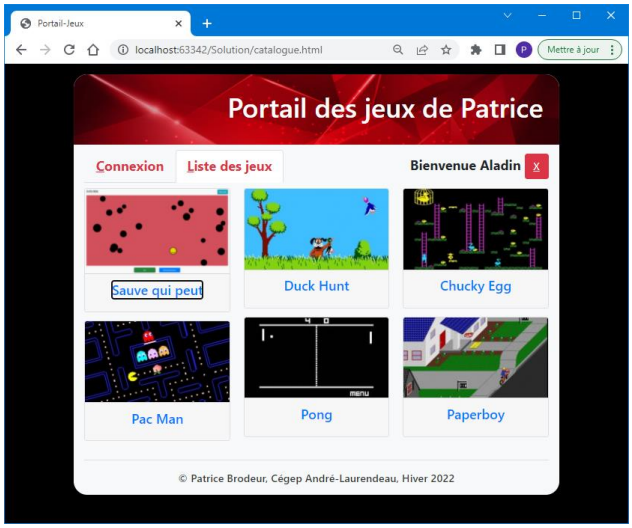
Au clic du lien « Liste des jeux » (pas authentifié)



Au clic du jeu « Sauve qui peut » (pas authentifié)



Lors d'une authentification réussie



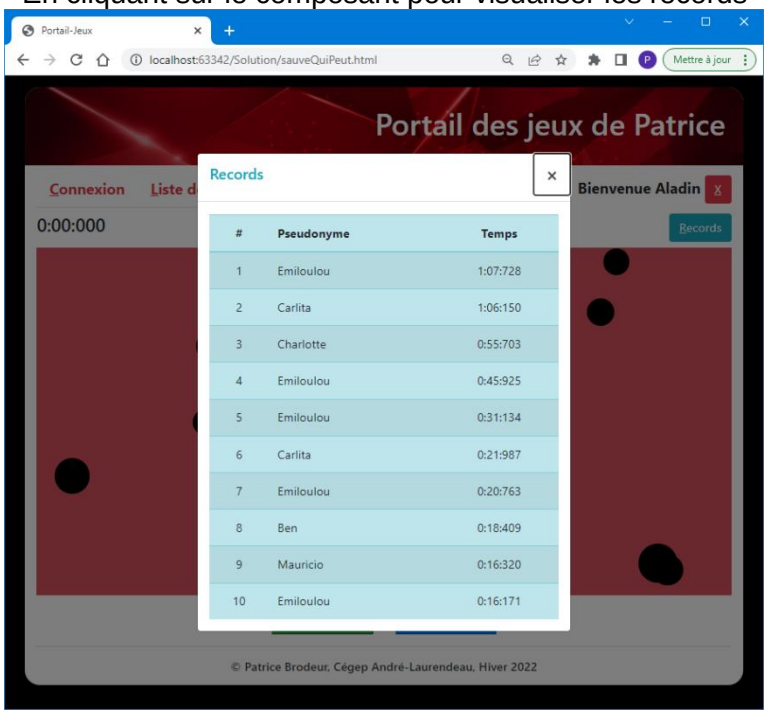
C) Page du jeu

Au clic du jeu « Sauve qui peut » (authentifié)

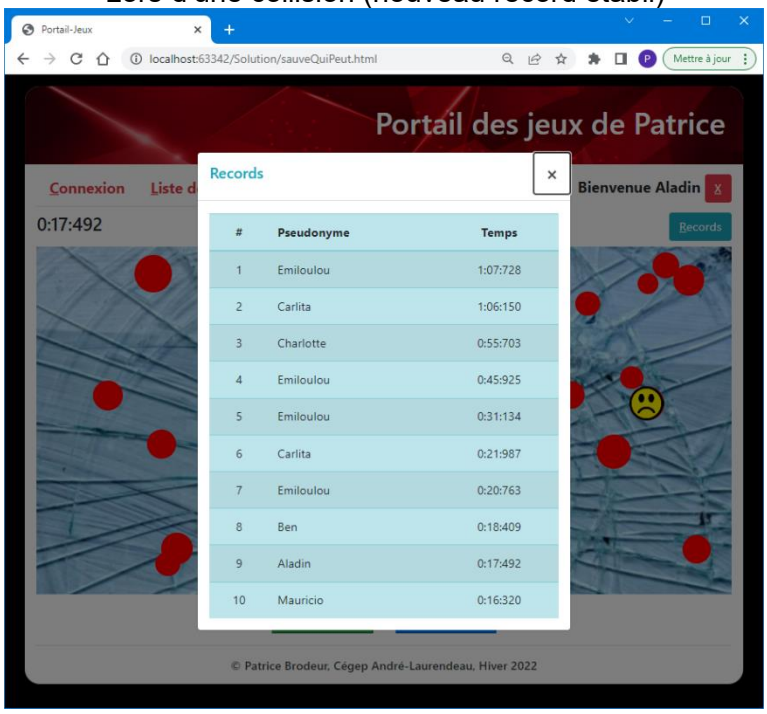
En cours de partie

Lors d'une collision (pas de nouveau record établi)

En cliquant sur le composant pour visualiser les records



Lors d'une collision (nouveau record établi)



8. Barème et critères de correction[1]

	Barème						
	Absent ou nettement insuffisant	**Faible**	**Insuffisant**	**Passable**	**Bien**	**Très bien**	**Excellent**
Design des pages[2] (Bootstrap)	0	2	4	6	7	8	10
Ergonomie[3]	0	2	4	6	7	8	10
Qualité du code[4] et déploiement sur cal-info.org	0	2	4	6	7	8	10
FONCTIONNALITÉS							
Inscription d'un abonné	0	3	6	9	10.5	12	15
Authentification d'un abonné[5]	0	2	4	6	7	8	10
Jeu - Déplacement de l'acteur	0	3	6	9	10.5	12	15
Jeu - Collisions / Gestion de l'état du jeu	0	3	6	9	10.5	12	15
Jeu - Chronomètre	0	1	2	3	3.5	4	5
Jeu - Records	0	2	4	6	7	8	10

	Seul	**En équipe de 2**
Sous-total	**/ 75**	**/ 100**
Pénalité (qualité du français)	**/ 7.5**	**/ 10**
Bonus (fait la partie authentification)	**/ 3**	**-**
Bonus (fait la partie chronomètre)	**/ 1.5**	**-**
Bonus (fait la partie Records)	**/ 3**	**-**
Bonus (mise en pause du jeu)	**-**	**/ 10**
Total	**/ 75**	**/ 100**

[1] Les critères de correction correspondent essentiellement au respect des différentes exigences fonctionnelles et techniques décrites dans l'énoncé.

[2] Look professionnel, réagit correctement aux différentes tailles d'écrans, dispositions et choix de couleurs harmonieux.

[3] Touches de raccourcis, action par défaut de la touche <Entrée>, focus sur les bons champs/boutons, disponibilité/indisponibilité des contrôles selon le contexte.

[4] Indentation du code, choix des noms des identificateurs (nomenclature et noms significatifs), commentaires pertinents dans le code (description des fonctions et méthodes, identification des références lorsque du code est copié ou inspiré de code trouvé sur Internet, identification des auteurs), validation du code HTML et CSS avec les validateurs du W3C.

[5] Les zones en jaune correspondent aux parties retirées du projet pour ceux/celles qui le font seuls. Dans l'éventualité où ces parties seront quand même réalisées, l'étudiant pourra obtenir un bonus pouvant aller jusqu'à 10%.