# INFORMATIONS GENERALES

|  |  |  |
| --- | --- | --- |
| Apprenti : | Nom: Premat | Prénom : Luca |
| Lieu de travail : | ETML / VENNES / 1010 LAUSANNE | |
| Chef de projet : | Nom: PIOT | Prénom: - |
| Du lundi 25.08.2025 au vendredi 27.10.2025 | |
| Dates de réalisation : | x périodes par semaine, x semaines (moins les jours fériés !) | |
| Horaire de travail : | x périodes | |
| Temps total : |  | |

# PROCÉDURE

* Tous les apprentis réalisent le projet sur la base d'un cahier des charges.
* Le cahier des charges est présenté, commenté et discuté en classe.
* Les apprentis sont entièrement responsables de la sécurité et sauvegarde de leurs données.
* En cas de problèmes graves, les apprentis avertissent le client au plus vite.
* Les apprentis ont la possibilité d’obtenir de l’aide externe, mais ils doivent le mentionner.
* Les informations utiles à l’évaluation de ce projet sont disponibles au chapitre 8.

# TITRE

***Snake en JS***

# SUJET

Faire un réplica du célèbre jeu du serpent.

# MATÉRIEL ET LOGICIEL À DISPOSITION

|  |
| --- |
| * Un PC ETML * Accès à Internet * Visual Studio Code * Node.js * Git et github |

# PRÉREQUIS

|  |
| --- |
| Modules de programmation de base |

# CAHIER DES CHARGES

## **Qualité**

1. Réaliser un programme informatique de qualité
   * Organisé (Le code est organisé en modules séparés pour une meilleure maintenabilité)
   * Compacté (pas de copié/collé,…)
   * Optimisé (utilisation de structures adaptées)
   * Testé
   * Commenté
2. Prouver que vous êtes digne de confiance lorsqu’on vous confie un projet
   * Pro-activité
     + **Poser des questions** au client
     + Faire des démonstrations
     + Utiliser un système de versioning de code (GIT)

## Description du projet

Ce projet consiste à réaliser un réplica du jeu classique du serpent en utilisant **JavaScript**.

Le jeu doit être dessiné sur un **canevas HTML** (<canvas>)

Le code **JavaScript** utilise la syntaxe des modules **ES** (ECMAScript Modules, ESM) pour organiser le code en plusieurs fichiers avec des responsabilités distinctes.

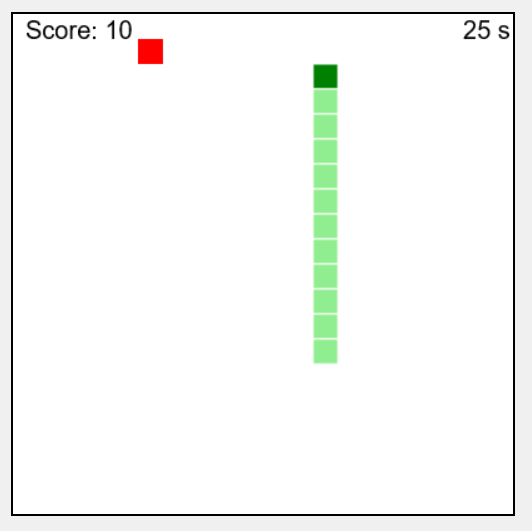
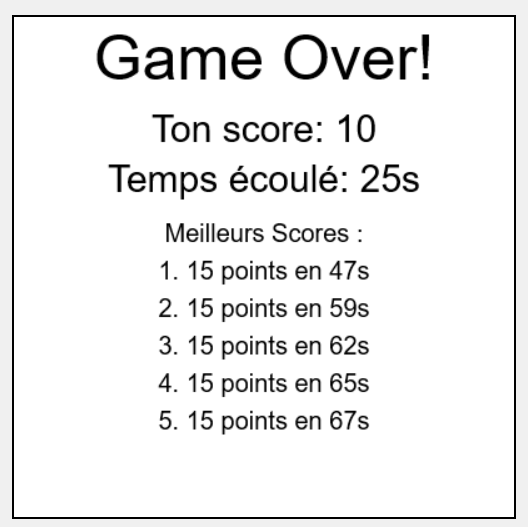
Le serveur de développement est géré par **Vite**, un outil moderne qui permet un rechargement rapide et une configuration minimale.

## Objectif du Jeu

Le but du jeu est simple :

* Contrôler un serpent qui se déplace sur une grille
* Manger de la nourriture (des pommes) pour grandir
* Eviter de se heurter aux murs ou à son propre corps.
* Le score augmente à chaque fois que le serpent mange de la nourriture.

## Fonctionnalités

**Déplacement du serpent :**

Le serpent peut être contrôlé à l'aide des touches directionnelles (haut, bas, gauche, droite).

Remarque : Si comme sur l’image ci-dessus, le serpent de déplace de bas en haut, la possibilité de descendre grâce à la flèche du bas est volontairement désactivée.

**Gestion des collisions :**

Si le serpent touche un mur ou son propre corps, la partie est terminée.

**Gestion du Score** : Un système de score affiche le nombre de fois où le serpent a mangé de la nourriture (représentée en rouge sur l’image ci-dessus).

**Chargement Asynchrone de la Configuration**

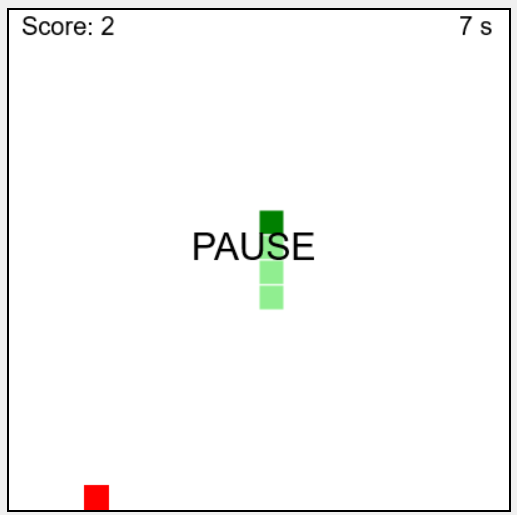
Externaliser les paramètres essentiels du jeu, tels que la taille des cases et la vitesse du jeu, dans un fichier de configuration **config.json**, qui sera chargé de manière asynchrone au démarrage du jeu. Cette approche permet de centraliser et de gérer facilement les configurations, tout en préparant le code à des évolutions futures.

**Gestion du Top 5 des Scores**

Ajouter une fonctionnalité de classement des scores, où seuls les 5 meilleurs scores sont conservés. Les scores doivent être sauvegardés de manière persistante en utilisant une API distante. À la fin de chaque partie, le score du joueur doit être comparé aux scores existants, et ajouté au top 5 si nécessaire, avant d'être synchronisé avec le serveur.

**Gestion du Temps et Intégration au Top 5 des Scores**

Ajouter un chronomètre pour mesurer le temps écoulé pendant chaque partie, affiché en haut de l'écran. Utiliser ce temps comme critère de départage en cas d'égalité de score dans le top 5, favorisant le joueur ayant terminé la partie le plus rapidement.



**Fonctionnalité de Pause et Reprise du Jeu**

Permettre à l'utilisateur de mettre le jeu en pause à tout moment en appuyant sur la touche "Espace", et de reprendre la partie avec la même touche. Pendant la pause, toutes les animations et le chronomètre doivent être suspendus, et un message "PAUSE" doit être affiché à l'écran pour indiquer l'état du jeu.

## Technologies Utilisées

**Node.js et npm**:

**npm** (Node Package Manager) est le gestionnaire de paquets de **Node.js**. Il permet aux développeurs d'installer, de partager et de gérer facilement des librairies et des outils JavaScript.

**npm** est inclus avec l'installation de Node.js. Cela signifie que lorsque vous installez **Node.js** sur votre machine, **npm** est automatiquement installé en même temps.

**Vite :**

Le projet utilise **Vite** comme serveur de développement. **Vite** est un outil moderne qui offre une expérience de développement rapide et fluide. Il permet de servir le projet avec un rechargement automatique à chaque modification du code.

**ECMAScript Modules (ESM)**

Le projet est structuré en utilisant la syntaxe ESM, permettant d'importer et d'exporter des fonctions et des variables entre les différents fichiers JavaScript. Cela permet de séparer les différentes responsabilités du code et de maintenir une structure propre et modulable.

**JSDoc**

Le code est documenté avec **JSDoc**, un standard de documentation pour JavaScript. **JSDoc** permet de générer automatiquement une documentation à partir des commentaires du code, ce qui améliore la lisibilité et la maintenabilité du projet.

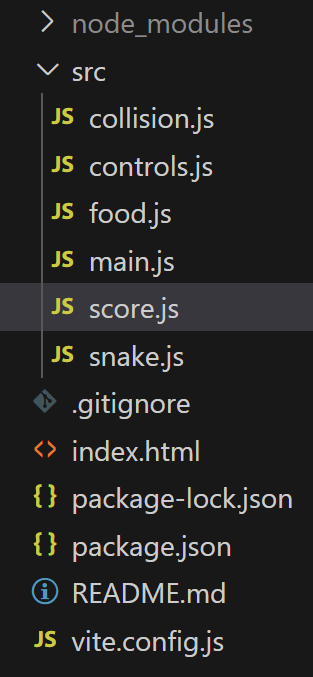
**API** [**jsonbin.io**](https://jsonbin.io/)

Cette API sera utilisée pour persister les 5 meilleurs scores au format **JSON**.

## Structure du Projet

Comme il s’agit certainement de votre 1er développement en **JavaScript** vous ne partirez pas de zéro !

Le code source de départ est organisé dans un répertoire **src/** pour séparer les fichiers de code des autres fichiers de configuration.



Vous devrez bien sûr ajouter de nouveaux fichiers à cette structure de départ pour développer l’ensemble des fonctionnalités demandées.

**Fichiers et Responsabilités :**

**main.js** :

Point d'entrée principal du jeu. Il initialise le jeu, configure les paramètres, et contrôle le cycle de rendu.

**snake.js** :

Contient les fonctions pour initialiser le serpent, le déplacer, et le dessiner sur le canevas.

**food.js** :

Gère la génération et le dessin de la nourriture sur la grille.

**controls.js** :

Gère les entrées utilisateur pour contrôler la direction du serpent.

**collision.js** : Contient les fonctions pour vérifier les collisions du serpent avec les murs ou avec lui-même.

**score.js** : Gère l'affichage du score sur le canevas.

Installation et Exécution

## Installation et serveur de développement

**Prérequis :**

Node.js : Assurez-vous que Node.js est installé sur votre machine.

**Installation :**

Récupérer le code de départ du **snake** puis :

* Placez-vous dans le répertoire snake avec la commande :

cd snake

* Installer les dépendances js grâce à la commande :

npm install

**Lancer le serveur de développement :**

Pour démarrer le serveur de développement avec Vite, exécutez :

npm run dev

Le projet sera servi localement à l'adresse <http://localhost:3000>.

## Contrôle de version

Votre code sera placé sous contrôle de version dans Github.

Créez un repository privé dans votre compte et invitez votre enseignant en lecture seule.

Vous appliquerez les bonnes pratiques de nommage à vos commits (c.f : [support ICT-306](https://github.com/ETML-INF/306-Agilite/blob/main/Supports/Git.pdf))

# Évaluation

1. Votre enseignant peut effectuer une ou plusieurs évaluations impromptues à l’oral tout au long du projet.
2. Vous devez être en capacité d’expliquer à votre enseignant chaque ligne de code que vous avez écrites.
3. Votre rapport final contiendra une section intitulée « Recours à l’IA » dans lequel vous décrirez quel usage vous avez fait de l’IA dans votre projet. Si vous n’y avez pas eu recours, vous détaillerez les raisons de ce choix.   
   Le recours à l’IA n’est pas interdite, mais elle ne peut servir que d’inspiration à la réalisation ou d’aide à l’apprentissage (montée en compétence en JavaScript par exemple). En cas d’abus et/ou d’inexactitude de cette section, l’évaluation du projet en tiendra compte.