Créer une application de type Trivial Poursuit

# TP du module 6 - Développer des services avec Angular

|  |
| --- |
| Avant de démarrer ce TP, il convient d’avoir suivi les vidéos des modules 1 à 6 et d’avoir réalisé les TP proposés. |

|  |
| --- |
| **Durée estimée** |
| 4 heures |

Énoncé

* Créer un service « OpenTriviaService » qui expose
  + une méthode « getQuestions » prenant en paramètre le niveau de difficulté (inutilisé à cette étape-là).
  + Cette méthode doit être asynchrone et renvoyer un tableau de questions comme suit (pour le moment, le retour est statique, mais sera remplacé plus tard par l’interrogation d’une API) :

[

{

  category: "Entertainment: Japanese Anime & Manga",

  type: "multiple",

  difficulty: "easy",

  question: "In &quot;Fairy Tail&quot;, what is the nickname of Natsu Dragneel?",

  correct\_answer: "The Salamander",

   incorrect\_answers: ["The Dragon Slayer", "The Dragon", "The Demon"]

  },

  {

  category: "Entertainment: Video Games",

  type: "boolean",

  difficulty: "medium",

   question: "&quot;Return to Castle Wolfenstein&quot; was the only game of the Wolfenstein series where you don&#039;t play as William &quot;B.J.&quot; Blazkowicz",

   correct\_answer: "False",

   incorrect\_answers: ["True"]

  }

]

* Créer les interfaces Question et Answer pour gérer le service, en fonction des données précédentes

# Dans le contrôleur home.page.ts,

* Interroger ce nouveau service pour récupérer une liste de questions.
* Afficher la première question,
* Puis la suivante lorsque l’utilisateur a répondu et qu’il clique sur le bouton « Question suivante ».
* À chaque fois que l’utilisateur répond, il faut lui indiquer s’il a répondu correctement ou non et compter ses points.
* Lorsqu’il n’y a plus de questions à poser
  + Afficher dans un toast le score du joueur ( = nombre de bonnes réponses).
  + Et un bouton propose de rejouer (retour au formulaire et le bouton « commencer » est affiché).
* Pour afficher les réponses, il convient de construire un tableau intermédiaire qui rassemble à la fois les mauvaises réponses et la bonne réponse (le service les renvoie séparément).
* Attention, il convient de mélanger les questions dans un ordre aléatoire pour surprendre l’utilisateur.

# Résultats attendus :

|  |  |
| --- | --- |
| Question non répondue : | Affichage des mauvaises réponses et de la bonne : |
|  |  |
| Score du joueur et bouton pour rejouer : |  |
|  |