API LABO Orchestra

* **Musicien** : PUBLISHER (Service)
* **Auditor** : SUBSCRIBER (Client)

---------------------------------------------------------------------------------------------------------------------------

**Task 1: design the application architecture and protocols**

| **#** | **Topic** |
| --- | --- |
| Question | How can we represent the system in an **architecture diagram**, which gives information both about the Docker containers, the communication protocols and the commands? |
|  | Insert your diagram here... |
| Question | Who is going to **send UDP datagrams** and **when**? |
|  | Les datagrammes UDP sont envoyés par chaque musicien toutes secondes. |
| Question | Who is going to **listen for UDP datagrams** and what should happen when a datagram is received? |
|  | C’est l’auditeur qui va écouter sur le port 2205. Une fois les datagrammes reçus, il va envoyer les informations sur les musiciens actifs en format JSON. |
| Question | What **payload** should we put in the UDP datagrams? |
|  | |  |  | | --- | --- | |  |  | |
| Question | What **data structures** do we need in the UDP sender and receiver? When will we update these data structures? When will we query these data structures? |
|  | Le musicien(Emetteur) est représenté par son instrument et le son de celui ci  L’auditeur (recepteur) recupère le tableau des musiciens actifs avec les propriétés: uuid, instrument etactiveSince. |

**Task 2: implement a "musician" Node.js application**

| **#** | **Topic** |
| --- | --- |
| Question | In a JavaScript program, if we have an object, how can we **serialize it in JSON**? |
|  | On utilise JSON.stringify(object) pour serialiser un objet en JSON. |
| Question | What is **npm**? |
|  | **npm** est le gestionnaire de paquet officiel de NODE.js . il automatise toute la gestion des dépendances et des paquets des projets JavaScript. |
| Question | What is the npm install command and what is the purpose of the --save flag? |
|  | npm install [--save] module  L’option –save sert à inclure cette dependance dans le fichier package.json |
| Question | How can we use the https://www.npmjs.com/ web site? |
|  | On peut y rechercher les packages js importants pour notre projet. |
| Question | In JavaScript, how can we **generate a UUID** compliant with RFC4122? |
|  | Il faut au préalable installer le package UUID |
| Question | In Node.js, how can we execute a function on a **periodic** basis? |
|  | En utilisant la fonction setInterval(function, period) ; period en ms |
| Question | In Node.js, how can we **emit UDP datagrams**? |
|  | On peut émettre des datagrammes UDP en utilisant la méthode send() sur un objet de type dgram.createSocket |
| Question | In Node.js, how can we **access the command line arguments**? |
|  | On peut le faire à travers les indices du tableau argv qui contient tous les arguments de la ligne de commande. |

**Task 3: package the "musician" app in a Docker image**

| **#** | **Topic** |
| --- | --- |
| Question | How do we **define and build our own Docker image**? |
|  | Oncommence par créer un fichier appelé Dockerfile dans lequel on défini toutes les applications les fichiers de configuration et les commandes à exécuter lors de la construction de l'image. Ensuite on crée l’image grâce à la commande docker build -t nom\_de\_limage chemin\_Dockerfile  Généralement si on est situé dans le même dossier que le Dockerfile on exécute juste : docker build -t nom\_de\_limage . |
| Question | How can we use the ENTRYPOINT statement in our Dockerfile? |
|  | ENTRYPOINT[“executable”, “param1”, “param2”]  (Forme Shell) ENTRYPOINT command param1 param2 |
| Question | After building our Docker image, how do we use it to **run containers**? |
|  | Enter your response here…  docker run [-option] nom\_de\_limage  -d if we need to run our container on background  -it pour le mode interactif avce nom\_de\_limage suivit de /b*in/*bash |
| Question | How do we get the list of all **running containers**? |
|  | docker ps |
| Question | How do we **stop/kill** one running container? |
|  | Docker kill/stop nom\_du\_container |
| Question | How can we check that our running containers are effectively sending UDP datagrams? |
|  | En utilisant un client postman avec filtre UDP |

**Task 4: implement an "auditor" Node.js application**

| **#** | **Topic** |
| --- | --- |
| Question | With Node.js, how can we listen for UDP datagrams in a multicast group? |
|  | On peut le faire en utilisant la fonction bind() sur un socket UDP |
| Question | How can we use the Map built-in object introduced in ECMAScript 6 to implement a **dictionary**? |
|  | const map = new Map()  map.set(key, value); |
| Question | How can we use the Moment.js npm module to help us with **date manipulations** and formatting? |
|  | npm install moment  var moment = require('moment'); // require  moment().format(); // pour le formatage  moment().subtract() ; //pour obtenir des intervalles de temps |
| Question | When and how do we **get rid of inactive players**? |
|  | Après 5s d’inactivité on supprime la clé() de la Map |
| Question | How do I implement a **simple TCP server** in Node.js? |
|  | Pour ce faire il faut au préalable inclure le module net avec :  const Net = require('net');  appeler la fonction createServer() sur cette variable. |

**Task 5: package the "auditor" app in a Docker image**

| **#** | **Topic** |
| --- | --- |
| Question | How do we validate that the whole system works, once we have built our Docker image? |
|  | On peut vérifier la validité de notre système en exécutant le script validate.sh |