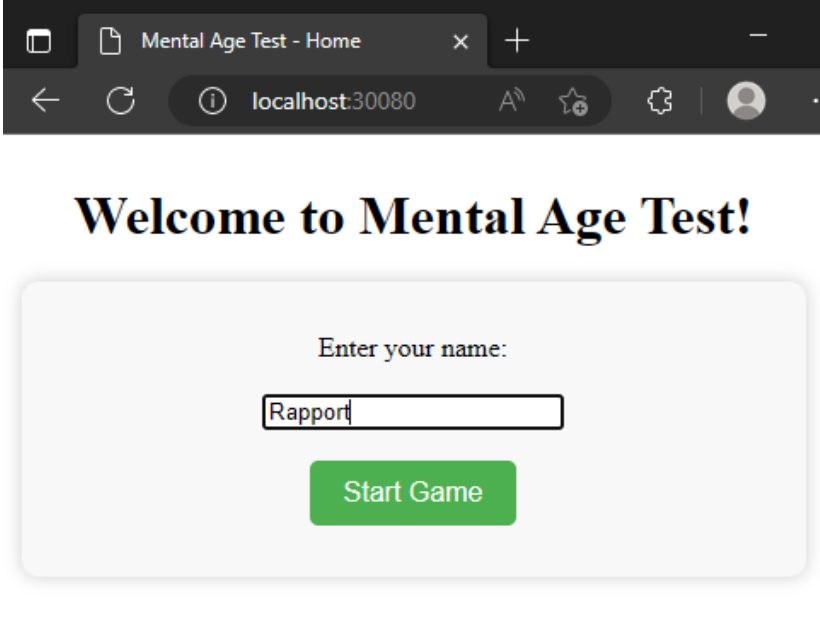


Projet Virtualisation: Mental Age Test

Réalisé par: Soufiane MOURCHID et Rui SUN

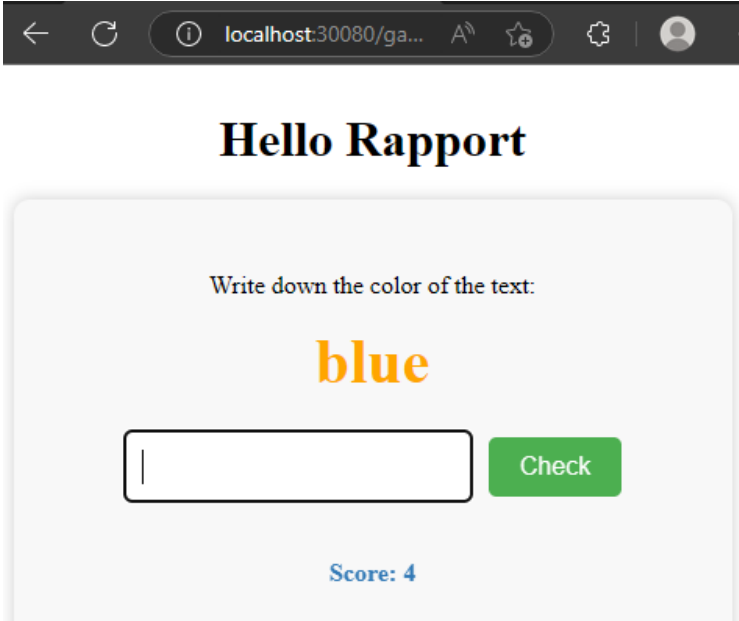
Démonstration du projet:

L'application commence par demander à l'utilisateur d'entrer un nom:



The screenshot shows a web browser window with the title "Mental Age Test - Home". The address bar displays "localhost:30080". The main content area features a large heading "Welcome to Mental Age Test!". Below this, there is a light gray rounded rectangle containing the text "Enter your name:" followed by a text input field with the value "Rapport". A green button labeled "Start Game" is positioned below the input field.

Après avoir appuyé sur le bouton "Start Game", le jeu commence. Le but du jeu est d'écrire la couleur du texte affiché, et non pas d'écrire le texte lui-même. Le jeu dure 30 secondes et le score final définit l'âge mental de l'utilisateur (le nom de ce dernier s'affiche en haut de l'écran). Une réponse correcte donne 1 point, une réponse mauvaise -1.



The screenshot shows the application during a game round. The browser address bar now shows "localhost:30080/ga...". The main content area displays the heading "Hello Rapport". Below it, the instruction "Write down the color of the text:" is shown, followed by the word "blue" in orange text. A text input field is provided for the user's answer, and a green button labeled "Check" is to its right. At the bottom of the screen, the text "Score: 4" is displayed.

Write down the color of the text:

red

Check

Score: 9

Time remaining: 00:09

Après 30 secondes soit écoulé, un classement avec le nom et le score des utilisateurs s'affiche:

Mental Age Test - Scoreboard

localhost:30080/sco...

Scoreboard

RANK	NAME	SCORE
1	test	20
2	Rapport	11
3	aeze	-2

Play Again

L'utilisateur pourra appuyer sur le bouton "Play Again" pour rejouer.








Les 3 différents services mis en place:

On a divisé l'application en 3 différents services pour assurer le bon fonctionnement.

Front-end "Game":

Le service "Game" est un front-end qui offre une interface utilisateur pour interagir avec le jeu "Mental Age Test". Ce service implémente les règles et les fonctionnalités du jeu. Le service lit également les entrées de l'utilisateur, tel le nom, et les envoie au back-end pour traitement.





Il permet également d'afficher des informations provenant du back-end à l'utilisateur, principalement le score. Il agit comme une couche d'abstraction entre le back-end et l'utilisateur, fournissant une expérience utilisateur transparente et conviviale. Les fichiers nécessaires pour le fonctionnement de ce service sont les suivants:

 Dockerfile	3/26/2023 8:26 PM	File	1 KB
 game.html	3/26/2023 8:26 PM	Microsoft Edge H...	1 KB
 index.html	3/26/2023 8:26 PM	Microsoft Edge H...	1 KB
 scoreboard.html	3/26/2023 8:26 PM	Microsoft Edge H...	1 KB
 scoreboard.js	3/26/2023 9:49 PM	JavaScript File	2 KB
 script.js	3/26/2023 9:41 PM	JavaScript File	4 KB
 style.css	3/26/2023 8:26 PM	Cascading Style S...	3 KB

Back-end "Api":

Le service "api" est le back-end/serveur qui fonctionne comme un lien entre le front-end et la base de données. Ce service reçoit les informations envoyées par le service "Game", telles que les entrées de l'utilisateur, et les utilise pour interroger la base de données.

Il joue un rôle essentiel dans la gestion et la manipulation des données stockées dans la base de données. Il permet de stocker, mettre à jour, supprimer et récupérer des données en fonction des requêtes du front-end à travers des algorithmes de traitement de données. De cette manière, le service "api" facilite l'interaction entre l'utilisateur et la base de données, en offrant une couche intermédiaire pour le traitement et la récupération des données. Les fichiers nécessaires pour le fonctionnement de ce service sont les suivants:

 Dockerfile	3/26/2023 8:26 PM	File	1 KB
 package.json	3/26/2023 8:26 PM	JSON Source File	1 KB
 package-lock.json	3/26/2023 8:26 PM	JSON Source File	58 KB
 server.js	3/26/2023 8:26 PM	JavaScript File	4 KB

Base de données "Db":



Le service "db" représente la base de données où sont stockées les données, principalement le nom des utilisateurs et leurs scores.

Il fournit une interface pour stocker, récupérer, mettre à jour et supprimer les données de la base de données. Il est également responsable de la gestion de la sécurité et de l'intégrité des données, en veillant à ce que les données soient stockées de manière fiable et à ce que les données sensibles soient protégées contre l'accès non autorisé. Ci-dessous on peut voir la mise à jour de le bases de données après avoir fini la démonstration réalisé avec le joueur "Rapport":

```
mysql> select * from scoreboard;
+-----+-----+
| playername | score |
+-----+-----+
| aeze       |    -2 |
| Rapport    |    11 |
| test       |    20 |
+-----+-----+
3 rows in set (0.00 sec)

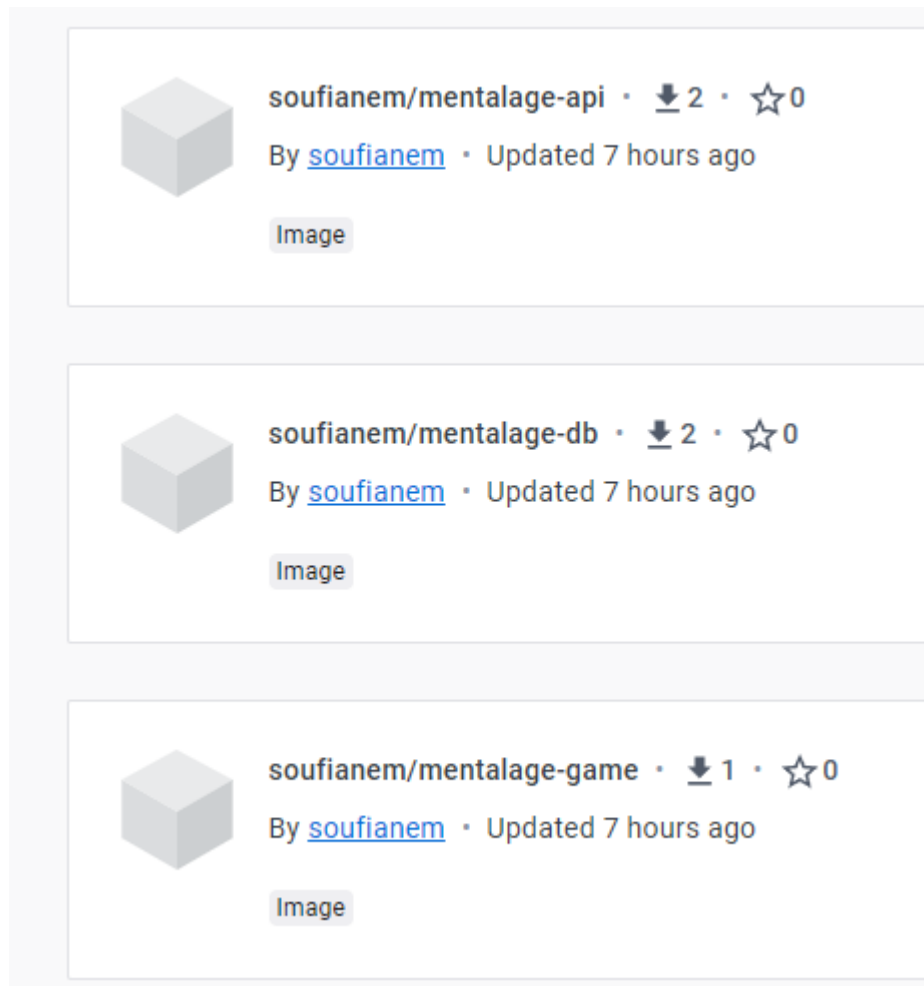
mysql>
```

Les fichiers nécessaires pour le fonctionnement de ce service sont les suivants:

 Dockerfile	3/26/2023 8:26 PM	File	1 KB
 scoreboard.sql	3/26/2023 8:26 PM	SQL Source File	1 KB

Organisation des fichiers du projet:

Comme vu précédemment, chaque service a son propre fichier DockerFile. Après création de l'image Docker, on les push sur le hub:



Au lieu de déployer chaque image dans le cluster tout seul, nous avons décidé de créer un dossier /deployment. De la même logique, nous avons aussi créé un dossier /service. Le dossier de projet est sous la forme suivante:

Name	Date modified	Type	Size
api	3/26/2023 8:26 PM	File folder	
db	3/26/2023 8:26 PM	File folder	
deployment	3/26/2023 9:19 PM	File folder	
game	3/26/2023 8:26 PM	File folder	
service	3/26/2023 9:19 PM	File folder	

Dossier Deployment:

Le dossier deployment contient les fichiers .yaml qui permettent le déploiement des images dans le cluster. On peut ensuite utiliser la commande:

```
kubectl apply -f .\deployment\
```

Le dossier deployment contient les fichiers suivants:

```
api-deployment.yaml
db-deployment.yaml
game-deployment.yaml
```

Dossier Service:

Pour créer les services on utilise la commande:

```
kubectl apply -f .\service\
```

Le dossier service contient les fichiers suivants:

```
api-service.yaml
db-service.yaml
game-service.yaml
```

En exécutant la commande:

```
kubectl get pods
```

On obtient:

NAME	READY	STATUS	RESTARTS	AGE
api-574774676b-tbzzb	1/1	Running	0	23s
db-5d4c8d766c-gxvcg	1/1	Running	0	23s
game-6644985f8f-1j4s1	1/1	Running	0	87s

Ce qui permet ainsi d'avoir le jeu disponible sur navigateur, comme il a été démontré dans la partie démonstration.

Communications entre services:

Game >> Api: (script.js)

```
fetch(apiUrl, {
  method: 'POST',
  headers: {
    'Content-Type': 'application/json'
  },
  body: JSON.stringify(data)
})
.then(response => {
  if (response.ok) {
    window.location.href = "scoreboard.html";
    // Score submitted successfully, do something here
  } else {
    alert("Failed to submit score");
    throw new Error('Failed to submit score');
  }
})
```

Game << Api:(scoreboard.js)

```
let apiUrl = "http://localhost:30000/scoreboard"
fetch(apiUrl)
.then(response => response.json())
```

Api<>db:(server.js)

```
// create MySQL connection pool
const pool = mysql.createPool({
  connectionLimit: 10,
  host: 'db',
  user: 'root',
  password: 'root',
  database: 'scoreboard_db'
});
```

et plusieurs requêtes dont:

```
// get a score by playername
app.get('/scoreboard/:playername', (req, res) => {
  const playername = req.params.playername;
  pool.query('SELECT * FROM scoreboard WHERE playername = ?', playername, (error, results) => {
    if (error) throw error;
    res.send(results[0]);
  });
});
```

Conclusion:

Dans ce projet, nous avons:

- ☒ Créé 1 service.
 - ☒ DockerFile.
 - ☒ Image Docker.
 - ☒ Push à Docker Hub.
 - ☒ Déploiement de l'image Docker dans le cluster Kubernetes.
 - ☒ Créer un service Kubernetes.
 - ☒ Accessible depuis le navigateur.
- ☒ Créé 2 services.
 - ☒ DockerFile.
 - ☒ Image Docker.
 - ☒ Push à Docker Hub.
 - ☒ Déploiement de l'image Docker dans le cluster Kubernetes.
 - ☒ Créer un service Kubernetes.
 - ☒ Accessible depuis le navigateur.
 - ☒ Communication avec le premier service.
- ☒ Créé une base de données.
 - ☒ Communiquant avec les deux services.

Page suivante pour les Google labs

Google Lab réalisés:

Soufiane MOURCHID:



Soufiane MOURCHID

Member since 2023

Your badge profile is not public and accessible.

Make badge profile public

Paths		Activities		Badges	
Course	Lab	Quest	Quiz	Game	
In progress		✓ Finished			
Activity	Type	Date started	Date finished	Score	Passed
Cloud Logging on Kubernetes Engine	Lab	4 hours ago	4 hours ago	100.0/100.0	✓
Managing Terraform State	Lab	4 hours ago	4 hours ago	100.0/100.0	✓
Managing Terraform State	Lab	Mar 8, 2023	Mar 8, 2023	50.0/100.0	
Interact with Terraform Modules	Lab	Mar 8, 2023	Mar 8, 2023	100.0/100.0	✓
Infrastructure as Code with Terraform	Lab	Mar 8, 2023	Mar 8, 2023	100.0/100.0	✓
Terraform Fundamentals	Lab	Mar 8, 2023	Mar 8, 2023	100.0/100.0	✓
Creating Infrastructure on Google Cloud with Terraform	Quest	Mar 8, 2023	0 minutes ago		✓
Introduction to Docker	Lab	Mar 6, 2023	Mar 6, 2023	100.0/100.0	✓
A Tour of Google Cloud Hands-on Labs	Lab	Mar 6, 2023	Mar 6, 2023	0.0/100.0	
Creating Infrastructure on Google Cloud with Terraform	Quest	Mar 8, 2023	0 minutes ago		✓
Introduction to Docker	Lab	Mar 6, 2023	Mar 6, 2023	100.0/100.0	✓
A Tour of Google Cloud Hands-on Labs	Lab	Mar 6, 2023	Mar 6, 2023	0.0/100.0	
Orchestrating the Cloud with Kubernetes	Lab	Feb 15, 2023	Feb 16, 2023	100.0/100.0	✓
Configuring Networks via gcloud	Lab	Feb 15, 2023	Feb 15, 2023	100.0/100.0	✓
Introduction to Docker	Lab	Feb 13, 2023	Feb 13, 2023	100.0/100.0	✓
Configuring Networks via gcloud	Lab	Feb 13, 2023	Feb 13, 2023	0.0/100.0	
Kubernetes Engine: Qwik Start	Lab	Feb 7, 2023	Feb 7, 2023	100.0/100.0	✓
Creating a Virtual Machine	Lab	Jan 9, 2023	Jan 9, 2023	100.0/100.0	✓
A Tour of Google Cloud Hands-on Labs	Lab	Jan 9, 2023	Jan 9, 2023	0.0/100.0	
A Tour of Google Cloud Hands-on Labs	Lab	Jan 9, 2023	Jan 9, 2023	100.0/100.0	✓

Rui SUN:



Rui SUN
Member since 2023

Your badge profile is not public and accessible.

Make badge profile public

<div><div>≡</div><div>Paths</div></div> <div><div>≡</div><div>Activities</div></div> <div><div>⚙</div><div>Badges</div></div>					
<div><div>Course</div><div>Lab</div><div>Quest</div><div>Quiz</div><div>Game</div><div>In progress</div><div>Finished</div></div>					
Activity	Type	Date started	Date finished	Score	Passed
Cloud Logging on Kubernetes Engine	Lab	5 days ago	5 days ago	100.0/100.0	✓
Managing Terraform State	Lab	Mar 9, 2023	Mar 9, 2023	100.0/100.0	✓
Interact with Terraform Modules	Lab	Mar 8, 2023	Mar 8, 2023	100.0/100.0	✓
Infrastructure as Code with Terraform	Lab	Mar 7, 2023	Mar 7, 2023	100.0/100.0	✓
Terraform Fundamentals	Lab	Mar 7, 2023	Mar 7, 2023	100.0/100.0	✓
Creating Infrastructure on Google Cloud with Terraform	Quest	Mar 7, 2023	5 days ago		✓
Configuring Networks via gcloud	Lab	Mar 5, 2023	Mar 5, 2023	100.0/100.0	✓
Orchestrating the Cloud with Kubernetes	Lab	Mar 5, 2023	Mar 5, 2023	100.0/100.0	✓
Kubernetes Engine: Qwik Start	Lab	Mar 5, 2023	Mar 5, 2023	100.0/100.0	✓
Kubernetes Engine: Qwik Start	Lab	Mar 5, 2023	Mar 5, 2023	75.0/100.0	
Introduction to Docker	Lab	Mar 4, 2023	Mar 5, 2023	100.0/100.0	✓
Getting Started with Cloud Shell and gcloud	Lab	Feb 15, 2023	Feb 15, 2023	100.0/100.0	✓
Getting Started with Cloud Shell and gcloud	Lab	Feb 15, 2023	Feb 15, 2023	0.0/100.0	
Creating a Virtual Machine	Lab	Jan 9, 2023	Jan 9, 2023	100.0/100.0	✓
A Tour of Google Cloud Hands-on Labs	Lab	Jan 9, 2023	Jan 9, 2023	100.0/100.0	✓