Commencé le	mercredi 29 novembre 2023, 14:44
État	Terminé
Terminé le	mercredi 29 novembre 2023, 14:45
Temps mis	58 s
Note	**1,00** sur 1,00 (**100**%)

Question 1

Correct

Note de 0,20 sur 0,20

Faites la distinction parmi les méthodes HTTP suivantes:

1. La méthode POST ✔ est utilisée pour créer une ressource.
2. La méthode PATCH ✔ est utilisée pour modifier partiellement une ressource.
3. La méthode PUT ✔ est utilisée pour remplacer une ressource spécifiée par un URI ✔ par une ressource fournie par la requête.

Question 2

Correct

Note de 0,20 sur 0,20

Quelle méthode HTTP est utilisée lorsque le navigateur veut récupérer la page *index.html* à partir d'un serveur?

a. SEND /index.html

b. POST /index.html

c. SEND /index.html HTTP/1.1

d. GET /index.html HTTP/1.1 ✔

e. GET /index.html

f. POST /index.html HTTP/1.1

Votre réponse est correcte.

La réponse correcte est :
GET /index.html HTTP/1.1

Question 3

Correct

Note de 0,20 sur 0,20

En vous basant sur le code suivant, en quel ordre sera affiché les messages de la console?

```
1  const getOne = () => Promise.resolve("One");
2
3  const two = () => {
4    getOne().then(console.log);
5    console.log("Two");
6  };
7
8  const three = async () => {
9    console.log(await getOne());
10   console.log("Three");
11 };
12
13 try {
14   two();
15   three();
16   console.log("Done");
17 } catch (error) {
18   console.log("Erreur");
19 }
```

a. Two
One
Done
One
Three

b. *Erreur* sera affichée en raison du *console.log* à la ligne 4.

c. Aucune des réponses

d. One
Two
One
Three
Done

e. Erreur
Two
One
Three
Done

f. Two ✔
Done
One
One
Three

Votre réponse est correcte.

La réponse correcte est : Two
Done
One
One
Three

Question 4

Correct

Note de 0,20 sur 0,20

Quelle méthode HTTP permet de décrire les options de communication ou les méthodes HTTP acceptées et accessibles par un serveur?

a. HEAD

b. CONNECT

c. GET

d. OPTIONS ✔

Votre réponse est correcte.

La réponse correcte est :
OPTIONS

Question 5

Correct

Note de 0,20 sur 0,20

Lorsqu'un client veut communiquer avec un serveur, les étapes suivantes sont réalisées:

1. Le client ouvre une ou plusieurs connexions TCP ✔ au serveur.

2. Le client envoie un(e) message ✔ HTTP au serveur.

3. Le client lit le/la réponse ✔ retourné(e) par le serveur.

Commencé le	mardi 28 novembre 2023, 17:53
État	Terminé
Terminé le	mardi 28 novembre 2023, 17:54
Temps mis	1 min 1 s
Note	**1,00** sur 1,00 (**100**%)

Question 1

Correct

Note de 0,20 sur 0,20

Quels sont les utilités de la librairie ExpressJS?

a. Express permet de définir des *middleware* pour répondre aux requêtes HTTP.

b. Express permet de définir des règles dynamiques pour traiter plusieurs routes similaires de la même manière.

c. Toutes les réponses. ✔

d. Express définit une table de routage qui peut gérer des requêtes HTTP en fonction de leur méthode HTTP utilisée et leur URI relative.

Votre réponse est correcte.

La réponse correcte est :
Toutes les réponses.

Question 2

Correct

Note de 0,20 sur 0,20

Plusieurs méthodes de Node.js et Express font usage de ***callback***. C'est quoi un Callback?

a. Un *Callback* est une fonction appelée après la fin de l'exécution d'une fonction primaire. ✔

b. Un *Callback* est simplement une fonction asynchrone.

c. Un *Callback* est un objet qui nous donne accès à des paramètres supplémentaires tels que *request* et *response* dans nos fonctions.

d. Aucune des réponses.

Votre réponse est correcte.

La réponse correcte est :
Un *Callback* est une fonction appelée après la fin de l'exécution d'une fonction primaire.

Question 3

Correct

Note de 0,20 sur 0,20

Choisissez le(s) déclaration(s) correcte(s) en analysant le bout de code ci-dessous:

```
let http = require("http");

const handleResponse = (response) => {
  response.writeHead(200, { "Content-Type": "text/html" });
  response.write("Request Completed");
};

const handleAPI = (request, response) => {
  if (request.method === "GET") {
    handleResponse(response);
    response.send();
  }
};

const handleWeb = (request, response) => {
  handleResponse(response);
  response.send();
};

http.createServer(handleAPI).listen(5000);
http.createServer(handleWeb).listen(5005);
```

- [x] a. ***response.send()*** est identique à ***response.json()*** sauf que ***response.json()*** convertit également des types primitifs et toutes autres données comme un objet *JSON*. ✔
- [x] b. *handleWeb* retournera "Request completed" pour les requêtes HTTP suivants: *POST*, *PUT*, *GET* et *DELETE*. ✔
- [ ] c. ***response.send()*** est obligatoire dans chaque *callback* de *createServer* puisque c'est la seule manière de retourner une réponse.
- [x] d. Le code HTTP 200 sera renvoyé en naviguant au port 5005. ✔
- [x] e. *handleAPI* retourne une page HTML comme réponse provenant d'une requête HTTP GET. ✔
- [ ] f. *handleAPI* retournera "Request completed" pour les requêtes HTTP suivants: *POST*, *PUT*, *GET* et *DELETE*.

Votre réponse est correcte.

Les réponses correctes sont :
Le code HTTP 200 sera renvoyé en naviguant au port 5005.,

handleAPI retourne une page HTML comme réponse provenant d'une requête HTTP GET.,

handleWeb retournera "Request completed" pour les requêtes HTTP suivants: *POST*, *PUT*, *GET* et *DELETE*.,

response.send() est identique à ***response.json()*** sauf que ***response.json()*** convertit également des types primitifs et toutes autres données comme un objet *JSON*.

Question 4

Correct

Note de 0,15 sur 0,15

Par défaut, Node.js est un environnement d'exécution [asynchrone] ✔ et il est utilisé au niveau du [Serveur et Client] ✔

Question 5

Correct

Note de 0,25 sur 0,25

Complétez le bout de code suivant qui récupère l'identifiant dans une requête HTTP:

```
let express = require("express");
let app = express();

app.get("/:id", ( request ✔ , response ✔ , next ✔ ) => {
    const id = request ✔ . params.id ✔ ;
    // ...
    // en assumant que les opérations ont été réalisées avec succès
    response ✔ . status(200) ✔ .send(id) ;
})

app. listen ✔ (5000);
```

Commencé le	mardi 28 novembre 2023, 19:05
État	Terminé
Terminé le	mardi 28 novembre 2023, 19:06
Temps mis	1 min 15 s
Points	1,00/1,00
Note	**1,00** sur 1,00 (**100**%)

Question 1

Correct

Note de 0,17 sur 0,17

Quelle est la différence entre la mise à l'échelle horizontale vs verticale d'une base de données?

a. L'horizontale concerne la répartition des données entre des instances secondaires, alors que la verticale concerne la répartition des données entre une instance primaire et des instances secondaires

b. L'horizontale permet d'avoir différentes colones pour chaque ligne d'un document permettant une BD plus flexible que la version vertical

c. L'horizontal est une méthode de replication où la base de données interagit avec un serveur et une nouvelle base est choisie en cas d'erreur, alors que la verticale concerne le répartitionnement des données sur différents serveurs

d. L'horizontale permet d'avoir un schéma de table dynamiques, alors que la verticale est liée à un schéma statique.

e. L'horizontale consiste à ajouter des machines dans la grappe de serveur pour diminuer la charge par machine, alors que la verticale consiste à augmenter la capacité d'une base de données en augmentant le nombre de CPUs, de RAM, etc. d'une seule machine. ✔

Votre réponse est correcte.

La réponse correcte est :
L'horizontale consiste à ajouter des machines dans la grappe de serveur pour diminuer la charge par machine, alors que la verticale consiste à augmenter la capacité d'une base de données en augmentant le nombre de CPUs, de RAM, etc. d'une seule machine.

Question 2

Correct

Note de 0,17 sur 0,17

Dans notre collection, chaque document contient une liste de "hobbies" qui contient les passions de l'utilisateur en question. Quelle commande permet de trouver tous les documents qui contient la passion "coding"?

Exemple d'un document:

```
{
  _id: ObjectId("507f191e810c19729de860ea"),
  name: "John",
  hobbies: ["swimming", "soccer", "coding"],
  // other elements
}
```

a. db.collection("users").findInArray({ hobbies: "coding" })

b. db.collection("users").find({ $array : { hobbies: "coding"} })

c. db.collection("users").find({ hobbies: ["coding"] })

d. db.collection("users").find({ hobbies: "coding" }) ✔

e. db.collection("users").findAll({ hobbies: ["coding"] })

Votre réponse est correcte.

La réponse correcte est :
db.collection("users").find({ hobbies: "coding" })

Question 3

Correct

Note de 0,17 sur 0,17

Le **_id** d'un des documents courants est basé sur une ancienne version du schéma de la collection qui n'est plus valide. On voudrait mettre à jour ce **_id**. Vous écrivez le code pour récupérer le document afin de remplacer son **_id**, mais celui-ci ne change pas. Quoi faire?

a. Un **_id** ne peut pas être modifié ✔

b. Aucune des réponses

c. Utiliser replaceOne()

d. Mettre l'option *replace* à vraie dans la commande updateOne()

e. Utiliser updateOne()

Votre réponse est correcte.

La réponse correcte est :
Un **_id** ne peut pas être modifié

Question 4

Correct

Note de 0,17 sur 0,17

MongoDB est structuré autour de 3 éléments.

Un(e) base de données ✔ contient un(e) ou plusieur(e)s collections ✔ .

Un(e) collection ✔ regroupe un ensemble de documents ✔

Question 5

Correct

Note de 0,17 sur 0,17

C'est quoi une projection dans MongoDB?

- a. Une projection permet d'effectuer des calculs avec les données
- b. Une projection permet de gérer l'affichage des résultats
- c. Une projection permet de sélectionner les champs qui doivent être retournés ✔
- d. Une projection permet de faire des requêtes sur le Serveur

Votre réponse est correcte.

La réponse correcte est :
Une projection permet de sélectionner les champs qui doivent être retournés

Question 6

Correct

Note de 0,17 sur 0,17

À quelle famille de NoSQL appartient MongoDB?

- a. BDD Orientée colonnes
- b. BDD Clé-Valeur
- c. BDD Orientée graphe
- d. BDD Orientée document ✔

Votre réponse est correcte.

La réponse correcte est :
BDD Orientée document

Commencé le	mercredi 29 novembre 2023, 14:59
État	Terminé
Terminé le	mercredi 29 novembre 2023, 14:59
Temps mis	31 s
Note	**1,00** sur 1,00 (**100**%)

Question 1

Correct

Note de 0,20 sur 0,20

Quel est le nom de la representation de l'interface utilisateur qui est gardée en mémoire et est synchronisée avec le vrai DOM?

- a. DOM Incrémental
- b. Shadow DOM
- c. DOM
- d. DOM Virtuel ✔

Votre réponse est correcte.

La réponse correcte est :
DOM Virtuel

Question 2

Correct

Note de 0,20 sur 0,20

En analysant le code ci-dessous, comment faut-il faire pour afficher le texte "hello" si et seulement si la valeur du *counter* change?

```
const App = () => {
  let [counter, setCounter] = useState(0);
  let [isCounterZero, setIsCounterZero] = useState(true);

  useEffect(() => {
    // à executer seulement quand counter change
    console.log('Hello');
  });

  const incrementCounter = () => {
    counter++;
    setIsCounterZero(false);
  };

  return (
    <div>
      <h1>{counter}</h1>
      <button onClick={incrementCounter}>Click Me</button>
    </div>
  );
};
```

a. Ajouter *counter* dans la liste des dépendances du useEffect ✔

b. Appeler le *hook* dans la fonction *incrementCounter*. (ex: useEffect())

c. Le code présent est valide et la chaîne de caractères sera affichée que lorsque la valeur du *counter* change

d. Ajouter un tableau vide dans la liste des dépendances du useEffect

e. Ajouter une condition dans le *hook* pour vérifier que c'est bel et bien la valeur du *counter* qui a changé

Votre réponse est correcte.

La réponse correcte est :
Ajouter *counter* dans la liste des dépendances du useEffect

Question 3

Correct

Note de 0,20 sur 0,20

Identifiez les énoncés erronnés données sur le code suivant:

```
const App = () => {
  const paragraphes = [
    'Mon premier paragraphe',
    '-',
    'Mon dernier paragraphe',
    '',
  ];

  return (
    <div>
      {paragraphes.map((p) => (
        {p && <p>Valid</p>}

        {p.startsWith('-') ? <p>Invalid</p> : <p>{p}</p>}
      ))}
    </div>
  );
};
```

- [ ] a. Toutes ces réponses.
- [x] b. React serait en mesure d'automatiquement détecter quels éléments ont été ajoutés, modifiés ou enlevés de notre tableau. ✔ Pour que cela soit vrai les clés (key) sont essentielles dans le contexte de la réconciliation.
- [x] c. Il est possible de retourner plusieurs éléments HTML dans la fonction *map()* ✔
- [x] d. Une boucle *For* serait plus efficace pour itérer à travers les éléments du tableau. ✔
- [ ] e. Il n'est pas possible d'appliquer de la logique conditionnelle en JSX.

Votre réponse est correcte.

Les réponses correctes sont :
React serait en mesure d'automatiquement détecter quels éléments ont été ajoutés, modifiés ou enlevés de notre tableau.,

Une boucle *For* serait plus efficace pour itérer à travers les éléments du tableau.,

Il est possible de retourner plusieurs éléments HTML dans la fonction *map()*

Question 4

Correct

Note de 0,20 sur 0,20

Que sera affiché par l'élément *h1* après avoir cliqué deux fois sur le bouton *Click Me*?

```
const App = () => {
  let [counter, setCounter] = useState(0);

  return (
    <div>
      <h1>{counter}</h1>
      <button onClick={() => (counter = counter + 1)}>Click Me</button>
    </div>
  );
};
```

a. Une erreur sera lancée.

b. 1

c. 0 ✔

d. 2

Votre réponse est correcte.

La réponse correcte est :
0

Question 5

Correct

Note de 0,20 sur 0,20

Quel est la différence entre un *state* (état) et un *prop* (propriété)?

a. Les *props* (propriétés) sont créées et gérées dans la composante tandis que les *states* (états) sont passés en paramètres à une composante.

b. Les *states* (états) sont créés et gérés dans la composante tandis que les *props* (propriétés) sont passées en paramètres à une composante. ✔

c. Les *states* (états) sont en fait des *props* (propriétés)

d. Aucune des réponses.

Votre réponse est correcte.

La réponse correcte est :
Les *states* (états) sont créés et gérés dans la composante tandis que les *props* (propriétés) sont passées en paramètres à une composante.

Commencé le	mercredi 29 novembre 2023, 17:19
État	Terminé
Terminé le	mercredi 29 novembre 2023, 17:20
Temps mis	1 min 29 s
Points	1,00/1,00
Note	**1,00** sur 1,00 (**100**%)

Question 1

Correct

Note de 0,17 sur 0,17

En React, souvent une composante enfante au n-ième niveau a besoin d'une propriété de la part du parent. Donc l'information doit être passée entre plusieurs composantes imbriquées à travers des propriétés (*props).* Comment peut-on esquiver cette situation?

- a. En utilisant le *Hook* useEffect
- b. À travers le *ContextAPI* ✔
- c. En utilisant le *Hook* useState
- d. À travers un Reducer ✔

Votre réponse est correcte.

Les réponses correctes sont :
À travers le *ContextAPI*,

À travers un Reducer

Question 2

Correct

Note de 0,17 sur 0,17

Dispatch ✔ permet de mettre à jour un état dans notre *Context* alors que Action ✔ permet de décrire l'événement ou le type de changement qu'on apporte.

Question 3

Correct

Note de 0,17 sur 0,17

À quoi sert un *Provider* dans le Context API en React?

- a. Un *Provider* permet de transmettre des informations d'une composante parente à ses enfants de n'importe quel niveau.
- b. Un *Provider* est simplement une composante React qui permet d'attribuer le *Context* aux composantes enfants. ✔
- c. Aucune des réponses

Votre réponse est correcte.

La réponse correcte est :
Un *Provider* est simplement une composante React qui permet d'attribuer le *Context* aux composantes enfants.

Question 4

Correct

Note de 0,17 sur 0,17

Un Reducer basé le patron *SAM,* applique le principe *Single source of truth.* Choisissez une réponse en lien avec ce principe.

- a. Un *Reducer* applique le principe *Single source of truth* ce qui signifie que chaque composante sauvegarde l'état global d'une application dans une place commune.
- b. Un *Reducer* applique le principe *Single source of truth* ce qui signifie que l'état d'une application est sauvegardé dans une place commune, accessible par tous ceux qui ont accès à cette place. ✔
- c. La composante parente, notamment le point d'entrée de notre application, contiennent tous les états utilisés à travers l'application. Ils sont alors passés en propriété (*props*) aux enfants.
- d. Le patron *SAM* n'applique pas le principe *Single source of truth.*

Votre réponse est correcte.

La réponse correcte est :
Un *Reducer* applique le principe *Single source of truth* ce qui signifie que l'état d'une application est sauvegardé dans une place commune, accessible par tous ceux qui ont accès à cette place.

Question 5

Correct

Note de 0,17 sur 0,17

Laquelle/lesquelles parmi les options suivantes n'est/ne sont pas un *Hook* en React?

- a. useState
- b. useEffect
- c. useCallback
- d. useProp ✔

Votre réponse est correcte.

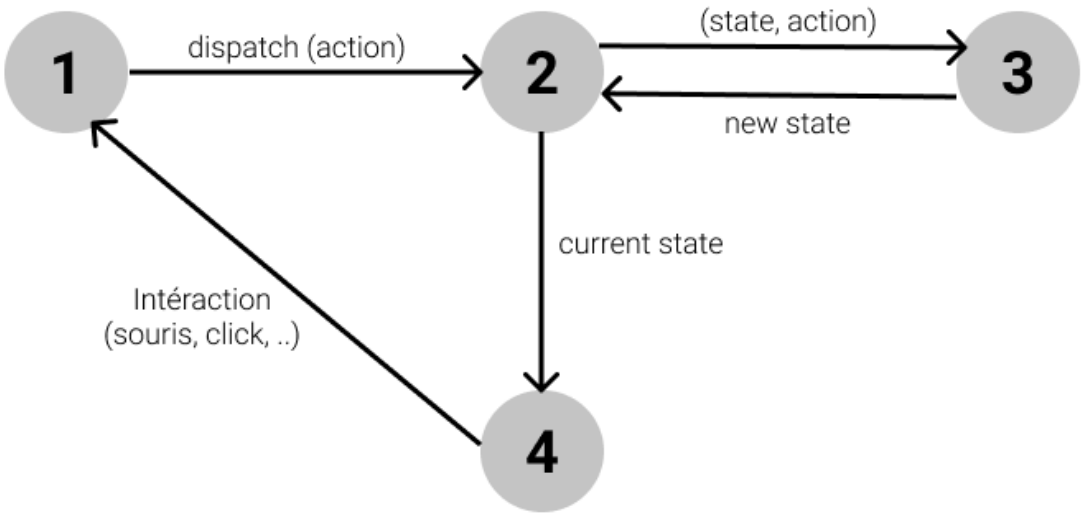
La réponse correcte est :
useProp

Question 6

Correct

Note de 0,17 sur 0,17

Sélectionnez les composantes *React* et *SAM* appropriées selon l'image suivante.



1: Action ✔

2: Context ✔

3: Reducer ✔

4: Composante React ✔

Commencé le	mercredi 29 novembre 2023, 23:01
État	Terminé
Terminé le	mercredi 29 novembre 2023, 23:02
Temps mis	19 s
Note	**1,00** sur 1,00 (**100**%)

Question 1

Correct

Note de 0,20 sur 0,20

Quel serait un avantage d'un architecture en microservices?

- a. Les microservices sont très petits, ce qui signifie que les développeurs peuvent produire des microservices puissants en peu de lignes de code
- b. Les microservices sont très faciles à gérer
- c. Toutes les réponses
- d. Un composant microservice peut être modifié ou changé indépendamment des autres composants. ✔
- e. Les microservices ne nécessitent pas beaucoup d'expertises pour les concevoir

Votre réponse est correcte.

La réponse correcte est :
Un composant microservice peut être modifié ou changé indépendamment des autres composants.

Question 2

Correct

Note de 0,20 sur 0,20

Quel est le type d'architecture Client-Serveur dans lequel le Serveur agit comme un intermédiaire dans le but de distribuer les transactions entre un Client et un ou plusieurs autres Serveurs?

- a. Aucune des réponses
- b. Gateway ✔
- c. Client à plusieurs Serveurs
- d. Serveur à plusieurs Clients
- e. Client et Serveur

Votre réponse est correcte.

La réponse correcte est :
Gateway

Question 3

Correct

Note de 0,20 sur 0,20

Quel architecture consiste d'un réseau décentralisé où chaque poste (*workstation)* ou noeud individuel possède les même capacités et responsabilités?

- a. Pair-à-pair ✔
- b. Pipe-and-filter
- c. Orientée événements
- d. Microservices
- e. Client-Serveur

Votre réponse est correcte.

La réponse correcte est :
Pair-à-pair

Question 4

Correct

Note de 0,20 sur 0,20

Il n'est pas recommandé d'utiliser le même URL ou route pour deux méthodes HTTP différents, car il y aura alors une ambiguité au niveau des ressources. Exemple:

DELETE /user pour supprimer un utilisateur dans notre base de données

POST /user pour ajouter un nouvel utilisateur dans nos données

Veuillez choisir une réponse.

- Vrai
- Faux ✔

La réponse correcte est « Faux ».

Question 5

Correct

Note de 0,20 sur 0,20

Lequel des énoncés suivants décrit le mieux REST?

- a. REST est un protocole basé sur les microservices
- b. REST est un service web standard
- c. REST est un style d'architecture ✔
- d. REST est une API d'un système distribué

Votre réponse est correcte.

La réponse correcte est :
REST est un style d'architecture