Commencé le	mercredi 21 février 2024, 18:30
État	Terminé
Terminé le	mercredi 21 février 2024, 19:29
Temps mis	59 min 47 s
Note	**21,25** sur 25,00 (**85**%)

Description

LISEZ CES INSTRUCTIONS AVANT DE COMMENCER L'EXAMEN

- L'examen est noté sur un total de 25 points.
- La note partielle associée à chaque question est marquée à gauche de la question.
- Certaines questions demandent d'ajouter une justification à votre réponse. Une bonne réponse sans justification valide ne sera pas acceptée.
- **Vous avez une seule tentative pour l'examen! Ne soumettez pas votre tentative à moins d'être 100% sûr(e) d'avoir terminé l'examen ! La soumission sera automatique à la fin de la période.**
- **En cas de doute sur le sens d'une question, faites une supposition raisonnable, énoncez-là clairement dans votre réponse et poursuivez. Une "question" vide est disponible sur la première page d'examen si vous avez besoin de plus de place ou pour des questions spécifiques.**
- **Vous avez accès à une copie des notes de cours PDF sur l'instance Moodle et droit à 1 feuille recto-verso (imprimée ou manuscrite) comme documentation.**

Bon travail!

Question 1

Terminé

Non noté

Cette question est un espace dédié à vos commentaires ou questions sur l'examen. Nous ne répondons à aucune question pendant l'examen.

Priorisez de mettre vos commentaires et/ou hypothèses directement dans la question spécifique.

Okay

Question 2

Correct

Note de 1,50 sur 1,50

Quelle est l'utilité de la commande **ng generate component XYZ** de l'outil **angular-cli** ?

- a. Générer le code de départ pour un nouveau component XYZ ✓
- b. Aucune, la syntaxe est invalide et doit plutôt être **ng generate new component XYZ**
- c. Générer de la documentation automatisée pour un component existant XYZ
- d. Générer des tests unitaires pour un component existant XYZ

Votre réponse est correcte.

Question 3

Correct

Note de 1,50 sur 1,50

Considérez le code suivant :

```
<div *ngFor="let name of listname; odd as odd">
  <hello *ngIf="odd" [nameObj]="{ name: name }"></hello>
</div>
<hr />

<div *ngFor="let name of listname; odd as odd">
  <hello [hidden] ="!odd" [nameObj]="{ name: name }"></hello>
</div>
```

Sachant que le tableau **listname** contient 5 éléments (que des *string*) et que le paramètre **odd** est égal à **true** seulement pour les index impairs dans un ***ngFor**, combien de components **<hello>** seront générés dans le DOM dans chaque boucle ?

- a. Aucun component ne sera généré puisque la syntaxe de ***ngIf** est invalide
- b. 5 components dans la 1re boucle et 5 components dans la 2e
- c. 2 components dans la 1re boucle et 2 components dans la 2e
- d. Aucun component ne sera généré puisque la syntaxe des ***ngFor** est invalide
- e. 2 components dans la 1re boucle et 5 components dans la 2e ✓

Votre réponse est correcte.

Question 4

Correct

Note de 1,50 sur 1,50

Les tests du côté client utilisant les librairies Jasmine et Karma sont toujours exécutés dans un ordre aléatoire d'un lancement des tests à un autre. Pourquoi ?

a. Ceci n'a aucun impact sur les tests exécutés et leurs résultats.

b. Ceci est une limitation du fonctionnement de l'outil Karma.

c. Ceci permet de détecter la présence de dépendances entre les tests unitaires. ✓

d. Ceci permet d'exécuter les tests plus rapidement.

Votre réponse est correcte.

Question 5

Correct

Note de 1,50 sur 1,50

Voici l'implémentation de votre **ChronometerService** qui gère une minuterie du temps total dans vos parties. Ses 2 méthodes sont appelées au début et à la fin de chaque partie et l'attribut *nbElapsedSeconds* est affiché à l'écran dans votre **GamePageComponent**. La fonction *interval* est un Observable qui émet à chaque seconde (1000ms).

Quelle sera la valeur initiale du temps affichée lorsque vous débutez une 2e partie après avoir complété une partie ayant pris 30 secondes ?

```
@Injectable({ providedIn: 'root'})
export class ChronometerService {
   public nbElapsedSeconds: number = 0;
   private chronometer: Subscription = new Subscription();
   public startChronometer(): void {
       this.chronometer = interval(N_MS_IN_SECOND).subscribe(() => {
           this.nbElapsedSeconds++;
       });
   }
   public stopChronometer(): number {
       if (this.chronometer) this.chronometer.unsubscribe();
       return this.nbElapsedSeconds;
   }
}
```

a. 0 secondes

b. Aucune de ses réponses

c. 30 secondes + le temps écoulé entre la fin de la 1re partie et le début de la 2e partie

d. undefined

e. 30 secondes ✓

Votre réponse est correcte.

Question 6

Correct

Note de 1,50 sur 1,50

Vous recevez l'erreur **'NG0304: 'app-info' is not a known element'** lorsque vous exécutez les tests unitaires de votre component app-parent. Sachant que la balise <app-info> est utilisée dans le gabarit du component testé, quelle serait la meilleure solution pour corriger l'erreur ?

a. Un spy sur le component app-info devrait être ajouté dans le tableau declarations de la configuration du TestBed.

b. Le component app-info devrait être ajouté dans le tableau declarations de la configuration du TestBed.

c. Le component app-info possède un défaut dans son code HTML et le défaut devrait être corrigé en premier.

d. Un mock du component app-info devrait être ajouté dans le tableau declarations de la configuration du TestBed. ✓

Votre réponse est correcte.

Question 7

Correct

Note de 1,50 sur 1,50

Quelle est la différence entre Express, NodeJS et Socket.IO ?

a. NodeJS est un environnement d'exécution à partir duquel on utilise les librairies Express et Socket.IO, pour une gestion à plus haut niveau des requêtes HTTP et de la communication WebSocket, respectivement. ✓

b. La librairie Socket.IO est nécessaire pour traiter le protocole WebSocket, mais Express est optionnelle pour un serveur utilisant NodeJS.

c. NodeJS, Express et Socket.IO sont des librairies côté serveur pour la communication réseau.

d. La librairie Express est nécessaire pour traiter des requêtes HTTP, mais Socket.IO est optionnel pour un serveur utilisant NodeJS.

e. NodeJS est un environnement d'exécution à partir duquel on utilise la librairie Express pour une gestion à plus haut niveau des requêtes HTTP et SocketIO est un protocole de communication bidirectionnelle.

Votre réponse est correcte.

Question 8

Correct

Note de 1,50 sur 1,50

Vous êtes responsables de l'ajout d'une application mobile en tant que client possible à votre projet. Vous avez choisi d'utiliser l'outil Flutter basé sur le langage de programmation Dart qui ne possède pas une librairie Socket.IO, mais supporte nativement le protocole WebSocket.

Pouvez-vous faire fonctionner l'application mobile avec votre serveur sans avoir à modifier le code de votre serveur ?

a. Oui, Socket.IO utilise le protocole WebSocket pour communiquer, donc les 2 systèmes sont compatibles

b. Oui, mais vous ne pouvez pas utiliser les fonctionnalités supplémentaires que Socket.IO offre par-dessus WebSocket.

c. Non, le client et le serveur doivent utiliser Socket.IO pour une communication valide. ✓

d. Non, le client et le serveur doivent être implémentés dans le même langage de programmation pour pouvoir utiliser WebSocket ou Socket.IO.

Votre réponse est correcte.

Question 9

Correct

Note de 1,50 sur 1,50

Combien de serveurs sont utilisés dans votre projet, excluant la base de données ?

a. 2 serveurs dynamiques : un qui sert le site web et un qui est responsable de la communication réseau entre les clients.

b. 1 serveur statique et 2 serveurs dynamiques : un qui sert le site web, un qui est responsable de la communication HTTP et un qui est responsable de la communication avec la base de données.

c. 1 serveur statique et 1 serveur dynamique : un qui sert le site web et qui communique avec le serveur dynamique et un responsable de la communication réseau entre les clients.

d. 1 serveur statique et 1 serveur dynamique : un qui sert le site web et un responsable de la communication réseau entre les clients. ✓

e. 2 serveurs statiques : un qui sert le site web et un qui est responsable de la communication réseau entre les clients.

Votre réponse est correcte.

Question 10

Correct

Note de 1,50 sur 1,50

Une fois qu'une demande de fusion (*Merge Request*) a été revue et approuvée par un membre de l'équipe, qui devrait être responsable de l'intégration du code dans la branche de destination ?

- a. Aucune de ses réponses.
- b. Le ou les membre(s) designé(s) par une convention quelconque dans l'équipe. ✓
- c. Seulement le membre d'équipe ayant revisé la majorité du code.
- d. Seulement un membre de l'équipe qui n'est pas l'auteur ou l'évaluateur de la demande fusion.
- e. N'importe quel membre de l'équipe.
- f. Seulement le membre d'équipe ayant implémenté la majorité du code.

Votre réponse est correcte.

Question 11

Correct

Note de 2,00 sur 2,00

Parmi les choix suivant, le ou lesquels sont des objectifs principaux d'un *stand up* dans l'approche Scrum ?

- a. Revoir et mettre à jour le calendrier du projet
- b. Assigner des tâches aux membres de l'équipe
- c. Faire une revue de code rapide des nouvelles fonctionnalités
- d. Identifier des obstacles au travail des membres de l'équipe ✓

Votre réponse est correcte.

Question 12

Terminé

Note de 3,75 sur 4,50

Vous êtes responsable de l'évaluation de la demande de fusion (*Merge Request*) d'un de vos collègues. Il a travaillé sur l'implémentation de la gestion des raccourcis pour le choix de réponses dans la vue du joueur.

Voici un extrait de son implémentation :

```
1  export class AnswerHandler {
2
3    toggleChoice(index: number) {
4      if (this.gameState === GameState.Answering) {
5        if (index < this.selectedChoices.length && this.gameService.time > 0 && !(this.gameService.notFinalAnswer)) {
6          this.selectedChoices[index] = this.selectedChoices[index] === false ? true : false;
7          this.gameService.selectChoice(indexChoice);
8        }
9      }
10   }
11
12   handleKeyboardEvent(event: any) {
13     if (this.gameState == GameState.Answering) {
14       switch (event.key) {
15         case '1':
16         case '2':
17         case '3':
18         case '4':
19           this.toggleChoice(parseInt(event.key) - 1);
20           break;
21         case 'Enter':
22           this.soumettreChoix();
23           break;
24         default:
25           break;
26       }
27     }
28   }
29
30 }
```

Identifiez et expliquez brièvement les problèmes d'assurance qualité avec ce code.

Les problèmes d'assurance qualité avec ce code sont les suivants (il y en a franchement trop selon moi):

- la condition "this.gameState == GameState.Answering" devrait être une variable booléienne ou vérifier dans une fonction appart afin d'alléger le code et améliorer la lisibilité ou à la rigueur, vu qu'il est vérifier dans les deux fonctions de cette classe, on devrait vérifier cette condition quelque part d'autre (logiquement dans l'endroit on appelle ces fonctions), aussi dans "toggleChoice", on fait un "===" pour vérifier que le type de ces deux variables est le même mais ce n'est pas le cas dans "handleKeyboardEvent", pourquoi?

- la ligne 5 (`index < this.selectedChoices.length && this.gameService.time > 0 && !(this.gameService.notFinalAnswer))` est beaucoup trop longue et illisible, cette condition devrait aussi être dans sa propre fonction ou variable avec un nom descriptif de ce qu'elle vérifie, ou ca devrait être séparés en plusieurs ifs au minimum;

- on a un switch cas inutile dans `"handleKeyboardEvent"`, déjà, plusieurs cases sont présent alors qu'ils ne servent à rien (rien ne se passe quand on appui sur 1 par exemple), aussi le case default est inutile, mais de plus, vu qu'on a seulement 2 cases, il n'y a pas besoin de faire un switch;

- la ligne 6 (`this.selectedChoices[index] = this.selectedChoices[index] === false ? true : false;`) est absurde, elle est complètement illisible et on peut la réduire énormément, comme ceci, voir plus :`this.selectedChoices[index] = !this.selectedChoices[index];`

- la ligne 7 utilise une variable avec un nom pas du tout approprier et je ne sais même pas d'où cette variable vient elle est déclaré nul part;

- la ligne 22 appelle une fonction (soumettreChoix()) et cette fonction est nommé en français, elle devrait être consistent avec le reste du projet et être en anglais (quelque chose comme submitchoice());

- index n'est pas un nom de variable assez descriptif, on devrait le changer à quelque chose comme "choice";

en conclusion il y a de nombreux problèmes de qualité de code et ce code serait beaucoup trop complexe à maintenir, tester ou même à lire (une personne qui ne travaille pas sur ce projet n'aurait aucune idée ce que ce code fait).

Commentaire :
- vous relevez qu'il y a duplication de code pour la vérification de l'état du jeu, mais votre explication est plus centrée sur l'implémentation que sur le fait que c'est un enjeu de qualité (0.25pt)

- if trop long à la ligne 5 : bien (0.75pt)

- les lignes 15 à 17 ne sont pas inutiles, elles font le même code que le case '4'. default case inutile : okay, mais manque explication (0.5pt)

- ternaire inutile ligne 6 : bien (0.75pt)

- indexChoice : okay (0.75pt)

- utilisation anglais/français : bien (0.75pt)

Question 13

Terminé

Note de 2,00 sur 5,00

a) Voici un extrait de code de la configuration des tests unitaire de votre classe DatabaseService responsable du lien avec la base de données MongoDB. Expliquez l'utilité de la librairie *MongoMemoryServer*. (2 points)

```
1  import { MongoMemoryServer } from "mongodb-memory-server";
2  import { DatabaseService } from "./database.service";
3
4  beforeEach(async () => {
5      databaseService = new DatabaseService();
6      mongoServer = await MongoMemoryServer.create();
7      const mongoUri = mongoServer.getUri();
8      await databaseService.start(mongoUri);
9      // Autres configurations
10 });
```

b) Votre collègue travaille sur la validation du mot de passe pour accéder à la page d'administration. Il a décidé de faire l'évaluation du côté du serveur dynamique en envoyant le mot de passe à travers un événement WebSocket nommé "**login**". Si le mot de passe est bon, le serveur vous répond à travers l'événement "**loginSuccess**", sinon, à travers l'événement "**loginFail**" et dans les deux cas il n'y pas de données qui accompagnent l'événement. Est-ce que l'approche proposée est bonne ? Justifiez votre réponse. (3 points)

a) Je pense que l'utilité de la librairie *MongoMemoryServer* est de simuler une instance de MongoDB afin de faire des tests sur le comportement de DataBaseService, sans avoir à vraiment manipuler notre vrai base de données, on doit avoir de nombreuses fonction dans ce service qui ajoute, supprime ou modifie des données, c'est donc très utile d'avoir cette libraire pour simuler ces actions.

b) L'approche proposée pour la validation du mot de passe n'est pas optimale, bien que l'événement WS "login" permet de vérifier le mot de passe, l'absence de données accompagnant les événements "loginSuccess" et "loginFail" peut être très mélangeant pour l'utilisateur, par exemple, en cas d'échec de la connexion, l'utilisateur ne saura pas à quoi cela est dû, un mot de passe incorrect ou une erreur de serveur, pour améliorer le UX, il faudrait donner plus de contexte.

Commentaire :
a) Bien

b) Dans ce contexte, il fallait plutôt utiliser HTTP. L'utilisateur ne devrait pas savoir quel événement WS est reçu par le site web. (-3pt).