M1IF 03 – Conception d'applications Web – Examen

Durée : 1 heure 30 – Documents autorisés (4 pages max) – Ordinateurs, calculatrices, tablettes, téléphones portables… interdits

Toutes les questions ont au moins une bonne réponse.
Il faut cocher toutes les bonnes propositions d'une question pour avoir un point.
Si vous cochez une mauvaise proposition, vous annulez la question.
Remplissez au stylo noir ou bleu la case de la ou des bonne(s) réponse(s) ; une croix ne suffit pas.
Ne barrez pas une mauvaise réponse, mettez du blanc.
Ne redessinez pas une case que vous avez effacée, laissez blanc.
La première question n'est pas prise en compte dans la notation, mais elle est indispensable pour la notation des autres questions.

1. Identification du sujet :
   Cochez les cases A et C

2. Une réponse HTTP avec un code de statut 301 :
   a) doit être vide
   b) doit contenir un message d'erreur
   c) doit contenir un header `Location`
   d) doit contenir un header `Content-Type`
   e) doit contenir la représentation d'une collection de ressources

3. En REST, une réponse HTTP avec un code de statut 200 fait suite à une requête envoyée :
   a) avec la méthode `GET`
   b) avec la méthode `POST`
   c) avec la méthode `OPTIONS`
   d) avec la méthode `DELETE`
   e) avec la méthode `HEAD`

4. La fonction `response => response.json()` peut être utilisée dans :
   a) un script jQuery
   b) un filtre
   c) une promesse après une requête en fetch
   d) une JSP
   e) une servlet

5. Un token JWT :
   a) est spécifique à chaque requête
   b) est généré par le client
   c) contient un identifiant opaque
   d) permet de s'authentifier sur plusieurs serveurs
   e) peut être transmis dans l'URL d'une requête

6. Sur votre VM, l'infrastructure qui expose vos TPs serveur en HTTPS s'appuie sur :
   a) un certificat SSL/TLS sur nginx
   b) un certificat SSL/TLS sur Tomcat
   c) le mécanisme de Server-Side Includes
   d) un reverse proxy sur nginx
   e) un load balancer sur nginx

7. Dans une servlet, indiquez les différences entre `requestDispatcher.forward()` et `response.sendRedirect()` :
   a) `requestDispatcher.forward()` permet de renvoyer un contenu au client

b) `requestDispatcher.forward()` permet d'appliquer le pattern MVC côté serveur
c) `response.sendRedirect()` permet d'appliquer le pattern MVC côté serveur
d) `response.sendRedirect()` permet de renvoyer un contenu au client
e) `response.sendRedirect()` spécifie un code de retour HTTP particulier

8. Pour implémenter le pattern MVC côté serveur en Java, d'après les bonnes pratiques vues en cours et en TP :
a) une JSP est une vue
b) un AJAX est un modèle
c) une servlet est un modèle
d) un filtre est un contrôleur délégué
e) un JavaBean est une vue

9. La SPA qu'il était demandé de réaliser en TP utilisait :
a) Mustache pour mettre à jour la vue
b) des JSP pour effectuer le routage
c) JWT pour contrôler l'affichage des vues
d) deux serveurs différents pour l'authentification et pour les ressources
e) une page HTML commune pour toute l'application

10. Un fichier war :
a) contient un fichier de configuration Maven pom.xml
b) contient un descripteur de déploiement web.xml
c) permet de spécifier une Web API
d) contient une application Web
e) doit être déployé sur un serveur Tomcat ou similaire

11. Le code suivant : `<% out.println("Bonjour"); %>`
a) provient d'une servlet
b) provient d'un outil de templating
c) est compilé à la première exécution
d) permet de simplifier l'écriture dans le buffer de sortie
e) ne sert à rien

12. En REST, la notion d'interface uniforme :
a) standardise les URLs de base des applications Web
b) s'appuie sur la notion de ressource
c) s'appuie sur le protocole HTTP
d) s'appuie sur la notion d'hypermédia
e) favorise la scalabilité

13. L'instruction `response.setContentType("application/json");` peut être utilisée dans :
a) un script jQuery
b) un filtre
c) une promesse après une requête en fetch
d) une JSP
e) une servlet

14. Parmi ces headers HTTP, cochez le(s) header(s) de cache :
a) `Accept`
b) `If-Modified-Since`
c) `ETag`
d) `If-Match`
e) `Same-Version`

15. En REST, lorsqu'une requête conduit à la création d'une ressource, il est recommandé de :
    a) renvoyer un header `Location`
    b) renvoyer un header `Created`
    c) renvoyer un code de statut indiquant le succès de la requête
    d) sauvegarder la ressource
    e) renvoyer une représentation de la ressource

16. Une requête **et** une réponse HTTP 1.1 doivent contenir :
    a) des headers
    b) un contenu
    c) une méthode
    d) des paramètres
    e) un code de statut

17. Dans votre TP4, les ressources qu'il était demandé d'exposer sur votre serveur étaient :
    a) la collection de résultats
    b) les instances de résultats
    c) la collection de ballots
    d) les instances de ballots
    e) la collection de bulletins

18. Cochez les propositions qui correspondent à des requêtes valides en termes de négociation de contenus (les Content-Length sont bons, inutile de recompter) :
    a)
    ```
    GET /users/toto HTTP/1.1
    Host: localhost
    Content-type: application/json
    Content-length: 0
    ```

    b)
    ```
    GET /users/toto HTTP/1.1
    Host: localhost
    Accept: application/json
    ```

    c)
    ```
    POST /users/toto HTTP/1.1
    Host: localhost
    Accept: application/json
    Content-type: application/x-www-form-urlencoded
    Content-length: 28

    email=toto@site.fr&pass=toto
    ```

    d)
    ```
    PUT /users/toto HTTP/1.1
    Host: localhost
    Accept: */*
    Content-type: application/json
    Content-length: 38

    {"email":"toto@site.fr","pass":"toto"}
    ```

    e)
    ```
    DELETE /users/toto HTTP/1.1
    Host: localhost
    Content-Type: */*
    ```

19. Côté client, pour optimiser le temps de chargement de l'app shell d'une SPA, on peut :
    a) placer les fichiers statiques sur un serveur différent de celui de l'API
    b) placer tous les templates dans le même script

c) placer les CSS au début de la page HTML
d) charger certains scripts en defer
e) séparer le métier de l'application dans plusieurs scripts

20. Un framework Web :
    a) permet de gagner du temps de développement
    b) fournit une implémentation de certains patterns
    c) doit être configuré avant d'être compilé
    d) s'appuie sur la notion de restifieur
    e) s'appuie sur la notion de conteneur

21. Un mécanisme de requêtage asynchrone :
    a) s'appuie sur le langage AJAX
    b) permet de réaliser des mashups de données
    c) s'appuie sur un mécanisme de callback
    d) permet d'améliorer la fluidité des applications Web
    e) permet d'améliorer la mise en cache des ressources d'une API

22. Les scripts d'intégration continue qu'il vous a été demandé de réaliser pour déployer votre code serveur :
    a) se connectent sur votre VM
    b) exécutent des goals Maven
    c) minifient le code métier de l'application
    d) téléchargent les dépendances
    e) font de la copie de fichier

23. Côté client, pour optimiser le temps de chargement du CRP d'une SPA, on peut :
    a) placer les fichiers statiques sur un serveur différent de celui de l'API
    b) placer tous les templates dans le même script
    c) placer les CSS au début de la page HTML
    d) charger certains scripts en defer
    e) séparer le métier de l'application dans plusieurs scripts

24. Le code suivant :

```
((a, b) => {
  const c = {d: a};
  const e = Mustache.render("{{d}}", c);
  $("#" + b).html(e);
  document.getElementById(b).innerHTML = e;
})("toto", "test");
```

    a) nécessite un moteur de templates pour fonctionner
    b) nécessite un moteur de servlets pour fonctionner
    c) utilise l'API DOM
    d) utilise jQuery
    e) utilise AJAX

25. Si tous les outils nécessaires sont chargés en mémoire, le code de la question précédente :
    a) ne fait rien
    b) fait deux fois la même chose
    c) modifie un élément de la page HTML
    d) modifie tous les éléments de même nom de la page HTML
    e) génère une erreur