

M1IF 03 – Conception d'applications Web – Examen

Durée : 1 heure 30 – Documents autorisés – Ordinateurs, calculatrices et téléphones portables interdits

Questions de cours (barème : 14 points)

1. Que contient le header `Host` et pourquoi doit-on obligatoirement l'indiquer dans une requête HTTP 1.1 ?
2. Quel est le point commun entre les headers `Last-Modified` et `ETag` ? Citez deux exemples d'applications, l'un où il est plus logique d'utiliser le premier, et l'autre où c'est le second qui est préférable.
3. Comment pourrait-on implémenter l'équivalent d'un ou de plusieurs filtres (`javax.servlet.Filter`) à l'aide de servlets ?
4. Quel est l'intérêt d'utiliser plusieurs types de `ViewResolver` en Spring Web MVC ?
5. Donnez deux headers HTTP à positionner lorsqu'un serveur renvoie une réponse au sujet d'une ressource RESTful, et expliquez pour chacun d'eux à quelle propriété de REST il correspond.
6. En quoi peut-on dire que la fonction `jQuery.json()` n'effectue pas réellement une requête AJAX ? Donnez un exemple de « vraie » requête AJAX en jQuery.
7. Citez deux exemples de données que l'on peut obtenir avec l'objet `window.performance.timing`.

Étude de cas (barème : 8 points)

Suite au dernier pic de pollution, vous avez été engagé-e par la ville de Lyon pour réaliser une application Web de covoiturage local. Cette application devra permettre à ses utilisateurs de :

- Se créer un compte (l'identifiant sera l'adresse email) et s'authentifier
- Proposer des trajets locaux en indiquant un itinéraire, une date, une heure (par exemple, Ambérieu-La Doua, le mercredi 4 janvier 2017, à 14h), et la parité de leur plaque d'immatriculation
- Visualiser les trajets proposés, et pour chaque trajet, un ensemble de prévisions : météo, qualité de l'air et restrictions de circulation
- Mettre en relation gratuitement les utilisateurs (pas de paiement en ligne à gérer)

Cette application s'appuiera sur Spring Web MVC. Elle sera optimisée pour répondre aussi rapidement que possible aux requêtes des utilisateurs. Toutefois, les vues « principales » (pages HTML) seront générées côté serveur en JSP. Les ressources non critiques seront requêtées en AJAX et traitées côté client. Les prévisions sont fournies par différents services, interrogés par un bean Spring, qui met à disposition des méthodes `String getMeteo(LocalDateTime heure, String ville)`, `String getAir(LocalDateTime heure, String ville)` et `String getTraffic(LocalDateTime heure, String ville)`.

Les questions ci-dessous ne concernent que les deux cas d'utilisation liés aux trajets (proposition et visualisation). Ne traitez pas ceux liés à la création de compte, à l'authentification ni à la mise en relation des utilisateurs.

Conception (barème : 5 points)

8. Listez les différents éléments dont vous aurez besoin pour réaliser la proposition et la visualisation de trajets. Indiquez leurs types (composants Spring (précisez le type), JSP, classes Java, pages statiques, scripts...) et à quoi ils servent.
9. Décrivez, à l'aide de diagrammes UML appropriés, leurs communications lorsqu'un utilisateur clique sur un lien lui permettant de visualiser un trajet. Les éléments du SI externes seront modélisés comme des boîtes noires.
Remarque : pour cette question, vous devez optimiser le chemin critique de rendu (CRP) de l'application et montrer qu'elle réagit en conséquence.

Programmation (barème : 3 points)

10. Écrivez la JSP permettant de renvoyer au client la vue principale correspondant à la visualisation d'un trajet.