SAE Atelier de développement d'application web
Semestre 4
Sujet Architecture – développement serveur

L'objectif de la SAE est de développer une application web et mobile se composant

- d'un backend gérant des données et proposant des services accessibles pour certains via une interface HTML et pour d'autres via une api retournant des données au format JSON. Le backend est dockerisé.
- d'une application web construite en javascript, consommant les données JSON proposées par l'api, et permettant de consulter et naviguer dans les données,
- d'une application mobile construite en flutter, consommant les données JSON proposées par l'api, et permettant de consulter et naviguer dans les données.

Cette SAE permet de mettre en application des connaissances et des techniques issues de la virtualisation, de l'architecture, de la gestion de données, de la programmation web, de la programmation mobile.

Sujet de travail Architecture : LaChaudièreAgenda.core

L'objectif est de développer LaChaudièreAgenda.core, une application PHP de backend qui permet de créer et stocker des événements culturels proposés par La Chaudière et de les rendre disponibles en ligne au travers d'une api.
LaChaudièreAgenda.core comprend deux composants principaux :

- Une api mettant à disposition les données au format JSON. L'api est librement utilisable et permet uniquement de rechercher et consulter l'ensemble de l'agenda culturel de La Chaudière,
- Une partie Admin permettant de créer et modifier des événements en saisissant les données dans des formulaires HTML. Cette partie admin est réservée aux utilisateurs enregistrés.

Les données gérées par LaChaudièreAgenda sont composées d'un titre, d'une description, d'un tarif, d'une ou deux dates (date ou début/fin), d'un horaire éventuel, et d'une liste d'images éventuellement sous la forme d'une URL. Les événements sont classées par catégories correspondant à un type d'événement : concert, spectacle, expo, conférence etc…

L'ensemble de LaChaudièreAgenda.core doit être déployé au sein d'une composition de services docker installés sur la machine docketu.iutnc.univ.lorraine.fr. Les deux composants indiqués ci-dessus doivent être exécutés dans deux services séparés.

Fonctionnalités détaillées :

Gestion des événements	**1, 2, 3, 4**
Gestion des utilisateurs	**5, 6**
API	**7, 8, 9, 10, 11**
Fonctions étendues	**12, 13, 14**

Les fonctions étendues sont à réaliser lorsque toutes les autres sont finalisées.

1) **Créer un événement :** affichage et traitement d'un formulaire de saisie d'un événement comprenant l'ensemble des données descriptives, y compris la catégorie. La description d'un événement est saisie et stockée en markdown et transformée en HTML au moment de l'affichage.
2) **Gestion des catégories :** affichage et traitement d'un formulaire de saisie et création d'une catégorie. Doit comprendre notamment un libellé de catégorie et une description saisie et stockée en markdown.
3) **Afficher la liste des événements :** affichage de la liste des événements de l'agenda, triés par date. On affiche uniquement le titre, la catégorie et la ou les dates.
4) **Afficher la liste des événements en filtrant par catégorie :** affichage de la liste d'événements dans le même format que la fonctionnalité 3, en ajoutant le filtrage par catégorie.

5) **Formulaire d'authentification :** un utilisateur doit s'authentifier pour pouvoir créer des entrées, en fournissant un identifiant (@mail) et un mot de passe.
6) **Contrôle d'accès :** seuls les utilisateurs inscrits et authentifiés peuvent créer des entrées.

7) **Api :** liste des catégories – la liste des catégories existantes est disponible au format JSON sur l'url /`api/categories`
8) **Api :** liste des événements - la liste des événements est disponible au format JSON sur l'url /`api/evenements`. Pour une entrée dans cette liste, seuls le titre, la date et la catégorie, sont retournés, accompagnés de l'url permettant d'obtenir le détail complet de l'événement. La liste est triée par date.
9) **Api :** événements d'une catégorie – la liste des événement d'une catégorie est disponible au format JSON sur l'url /`api/categories/{id}/evenements`. Mêmes données retournées que pour la fonctionnalité 8.
10) **Api :** détail d'un événement – un événement décrit de façon complète est disponible au format JSON à l'url /`api/evenements/{id}`
11) **Api :** Liste d'événements datés : pour les fonctionnalités 8 et 9, possibilité de préciser si on souhaite obtenir les événements des mois passés, du mois courant, ou des mois futurs. On doit pouvoir combiner. Sous la forme :
`/api/evenements?periode=passee,courante,futur`

12) **Publication d'événements :** les événements créés doivent être explicitement publiées pour être disponibles dans l'api. Ils peuvent être dépubliés. La publication/dépublication des événements se fait au travers de l'affichage des listes d'événements dans l'application d'administration.
13) **Création d'utilisateurs :** un utilisateur particulier (super-admin) peut créer de nouveaux utilisateurs administrateurs de l'agenda ; il saisit l'identifiant et le mot de passe.
14) **Tri des listes d'événements dans l'api :** l'api permet de trier les listes selon différents critères. Le tri doit être optionnel, et exprimé de la façon suivante :
    a. `/api/evenements?sort=date-asc :` tri selon la date ascendante,
    b. `/api/evenements?sort=date-desc :` tri selon la date descendante
    c. `/api/evenements?sort=titre :` tri selon le titre
    d. `/api/evenements?sort=categorie :` tri selon la catégorie

Rendus et évaluation

Le projet doit être rendu au plus tard le samedi 14 juin 2025 à 12h. Il est attendu, dans l'espace arche prévu à cet effet, un dépôt d'un document pdf comprenant :

- Noms et prénoms des membres du groupe,
- url du dépôt git public contenant le code,
- url de l'application d'administration installée sur docketu.iutnc.univ-lorraine.fr
- url de l'api installée sur docketu.iutnc.univ-lorraine.fr

- identifiant-mot de passe des utilisateurs créés – au moins deux utilisateurs sont exigés,
- la base de données doit comprendre au moins 8 événements créés dans 3 catégories différentes.

Démonstration : la démonstration du projet devra se baser sur un jeu de données préparé à l'avance permettant de montrer les différentes fonctionnalités, notamment de recherche. Il est fortement recommandé d'utiliser la librairie FakerPHP/faker pour générer ces données. La démonstration devra également illustrer la création d'une nouvelle entrée.

Critères d'évaluation du projet

Architecture	Respect des principes de découplage des couches	30%	Séparation actions/services/ORM, namespaces et autoload, répertoires, single action, dispatcher
code	Utilisation de slim/twig/eloquent, qualité du code	30%	Utilisation des fonctionnalités Slim, twig et twig-view, modèles, associations et exception Eloquent,
sécurité	csrf, injection, protection des mots de passe	20%	Protection contre l'injection sql et xss, filtrage et validation de données, utilisation d'un token CSRF, protection des mots passes : stockage crypté, contrôle de la qualité
déploiement	Docker compose, docketu	10%	Docker-compose.yml portable et configurable, installation sur docketu
collaboration	Usage de git, branches, tests	10%	Git, branches

Recommandations concernant le déploiement sur docketu.iutnc.univ-lorraine.fr

1. Pour éviter des problèmes de droits liés à la contrainte de même origine, vous devez ajouter dans toutes vos réponses à des requêtes sur l'API le header 'Access-Control-Allow-Origin' avec comme valeur '*'. Ainsi, pour toutes vos réponses :
   `$rs = $rs->withHeader('Access-Control-Allow-Origin', '*') ;`
2. Pour les mêmes raisons, il est recommandé de regrouper dans le même conteneur l'api les fichiers statiques de l'application web monopage (page html, css, code js). Ainsi, l'application fera des requêtes fetch sur le même domaine (url) que son origine.
3. Dans la description des services docker, utilisez la construction `restart` de manière à ce que les services soient redémarrés automatiquement et restent ainsi opérationnels. Par exemple :
   `restart: unless-stopped`
4. Il est fortement recommandé de ne pas attendre la fin des développements pour mettre en œuvre et tester le déploiement sur la machine docketu.iutnc.univ-lorraine.fr