Module 2	Développer pages web en lien avec base de données			
Programmation client side	Séance	S02	Activité	A004

Activité de synthèse des compétences consacrées à la partie front de notre application Web.

Je vous propose dans cet atelier de réaliser la synthèse des apprentissages de la programmation web côté front et de poursuivre la découverte du développement et de l'usage d'Api Web.

Pour réaliser cette activité, il vous faut :

- Créer la base de données ContactsDB,
- Exécuter le script téléchargeable pour créer les objets de la base et les données
- Installer le projet d'Api Web ApiContacts

Après avoir pris connaissance de la demande, procédez à l'installation des outils de tests des Wep Api et des formulaires Web qui consomment ces services (dernier point du document).

Vous devez réaliser une page permettant de saisir les coordonnées d'un utilisateur et stocker sa photo via un formulaire en respectant les contraintes suivantes :

- Le formulaire doit être responsiv. Recourez à Boostrap pour assurer un bon rendu visuel sur smartphone comme sur desktop. Vous trouverez sur le web de nombreux sites proposant une aide à la génération de formulaires.
- Votre formulaire doit pouvoir être chargé par des navigateurs très divers. Votre code devra donc être cross platform. Coder avec jQuery les événements, les modifications de style et les mécanismes de manipulation du DOM. Evitez les techniques très récentes qui peuvent ne pas être supportées par les navigateurs de la génération précédente ou alors implémentez une « prothèse » voir la définition de polyfill https://fr.wikipedia.org/wiki/Polyfill .
- Pour l'implémentation des mécanismes de validation, vous avez le choix. Privilégier la validation HTML 5 quand celle-ci est envisageable, recourir au plugin de validation jQueryValidate ou implémenter vos propres méthodes. Dans tous les cas, vous devez fournir à l'utilisateur un feedback lui signalant ces erreurs en privilégiant des approches standards.
- Les échanges avec les serveurs sollicités seront réalisés via des appels hors bande d'API Web

L'application doit permettre le stockage des informations des utilisateurs d'un site de gestion de blogs.

Les données d'un utilisateur, représentées ci-dessous au format json :


```
{
    "userId": 1,
    "name": "Bost",
    "userName": "Vincent Bost",
    "email": "vincent.bost@afpa.fr",
    "address": {
        "street": "2 route de la Maurie Haut",
        "suite": "La Maurie",
        "city": "Sainte Féréole",
        "zipCode": "19270",
        "geo": {
            "latitude": "45.2281",
            "longitude": "1.5828"
        }
    },
    "phone": "06 40 75 27 78",
    "birthDay": "1962-01-13T00:00:00",
    "photoUrl": "UsersPhotos/1.png",
    "posts": []
}
```


Ces données sont stockées dans une base SQL Server via des web api
Pour le moment, ces API seront disponibles localement ….

Les données de la commune de localisation de l'utilisateur sont obtenues auprès du site https://zippopotam.us/. Exemple de restitution avec le code postal de Brive.

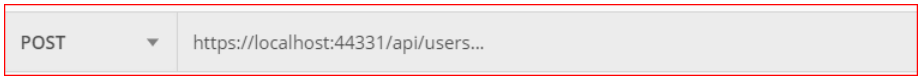
```
{
    "post code": "19100",
    "country": "France",
    "country abbreviation": "FR",
    "places": [
        {
            "place name": "Brive-la-Gaillarde",
            "longitude": "1.5333",
            "state": "Limousin",
            "state abbreviation": "B1",
            "latitude": "45.15"
        }
    ]
}
```


L'appel retourne un tableau de lieux places[], les codes postaux représentant des bureaux distributeurs.

Les données de l'utilisateur sont stockées en deux temps :

Une première requête pour créer l'utilisateur.

Les données sont isolées sur un **premier formulaire** et transmises en json dans le corps de la requête.




```
{

    "name": "Morillon",
    "userName": "Jean Morillon",
    "email": "vincent.bost@afpa.fr",
    "address": {
        "street": "2 route de la Maurie Haut",
        "suite": "La Maurie",
        "city": "Sainte Féréole",
        "zipCode": "19270",
        "geo": {
            "latitude": "45.2281",
            "longitude": "1.5828"
        }
    },
    "phone": "06 40 75 27 78",
    "birthDay": "1962-01-13T00:00:00"


}
```


Le serveur retourne l'objet créé avec son identifiant :


```
{
    "userId": 3,
    "name": "Morillon",
    "userName": "Jean Morillon",
    "email": "vincent.bost@afpa.fr",
    "address": {
        "street": "2 route de la Maurie Haut",
        "suite": "La Maurie",
        "city": "Sainte Féréole",
        "zipCode": "19270",
        "geo": {
            "latitude": "45.2281",
            "longitude": "1.5828"
        }
    },
    "phone": "06 40 75 27 78",
    "birthDay": "1962-01-13T00:00:00",
    "photoUrl": null,
    "posts": []
}
```

Deuxième requête

Si la transaction de création s'est bien déroulée, alors une deuxième requête permet d'uploader la photo sélectionnée au préalable via le contrôle input file (bien entendu si une photo a été sélectionnée…)
Afin de faciliter le traitement de la requête, isolez votre champ input file sur un deuxième formulaire.

Quelques précisions : Dans l'url, le dernier segment de route, ici 1, désigne l'identifiant de l'utilisateur. Il est récupéré dans l'objet json transmis dans la réponse du serveur.
Pour pouvoir transmettre des données volumineuses, comme des images, le formulaire HTML doit spécifier que le type d'encodage (enctype) est « multipart/form-data ». Vous pouvez utiliser Formdata pour charger les données.

Assurez-vous via l'attribut de la balise html input file que seules des images puissent être proposées à l'utilisateur en téléchargement.

Les besoins de l'interface

Pour la date de naissance : je vous suggère d'utiliser le composant DatePicker Bootstrap. Vous pouvez aussi utiliser un autre composant (jQuery par exemple) mais vous risquez alors d'avoir plus de difficultés à mettre au point le rendu visuel. Voir la documentation https://bootstrap-datepicker.readthedocs.io/en/latest/.
Vous avez la possibilité de travailler sur la configuration de votre composant en mode sandbox depuis le site.
Pour l'initialisation des paramètres linguistiques, chargez le fichier de la langue française et suivez les instructions au paragraphe I18N. Vous pouvez prendre connaissance du principe de l'internationalisation des logiciels ici https://fr.wikipedia.org/wiki/Internationalisation_(informatique)

Pour le choix de la commune : mettre en œuvre l'auto-complétion, en implémentant un composant compatible avec Bootstrap
Testez éventuellement un composant réalisé pour bootstrap 5 (mais vous pouvez la aussi retenir un autre composant). Pour identifier ces composants, rechercher avec les termes Autocompletion ou Typeahead J'ai testé celui-ci avec Bootstrap 3 et 4. Dont la documentation se trouve ici : https://bootstrap-autocomplete.readthedocs.io/en/latest/
Vous avez aussi des composants modifiés pour bootstrap 5. A tester :
https://www.cssscript.com/autocomplete-typeahead-bootstrap-5/
https://webcodeflow.com/bootstrap-5-autocomplete-typeahead-js/
https://github.com/gch1p/bootstrap-5-autocomplete

Pour la photo, il serait sympa de pouvoir la visualiser avant de la soumettre au téléchargement. A faire une fois que le reste fonctionne…

Pour les thèmes, vous pouvez aussi utiliser les éléments fournis par mdbootstrap pour des interfaces avec une surcouche Material Design (google) pour bootstrap. https://mdbootstrap.com/

Au niveau de l’appel des services Web via Ajax.

Explorer comment produire un objet javascript Json à partir des données d’un formulaire

Explorer comment sélectionner et uploader des fichiers.

Regarder les techniques de traitement asynchrone et comment synchroniser les deux appels asynchrones respectivement invoqués pour Créer l’utilisateur puis stocker et référencer son image.

Pour effectuer vos tests :

Pour le test des API Web côté client :
Vous avez de nombreux outils comme Postman, extension chrome Avanced rest client.
Vous pouvez installer et tester ces outils,

Vous trouverez de l'aide ici pour prendre en main l'outil : https://www.c-sharpcorner.com/article/how-to-use-postman-with-asp-net-core-web-api-testing/ ou https://openclassrooms.com/fr/courses/4668056-construisez-des-microservices/5123020-testez-votre-api-grace-a-postman

Vous pouvez installer ARC à partir du magasin Google Webstore.
https://chrome.google.com/webstore

Pour tester et documenter les services Web côté serveur, un outil fantastique (à installer absolument).
Swagger qui s'appuie sur un standard OAS Open Api Spécification dans sa version 3.

https://swagger.io/tools/swagger-ui/

Pour installer cet outil, appuyez-vous sur le tutoriel fournit par microsoft. Il propose une version incorporée de Swagger et des outils de génération auto très utiles.

https://docs.microsoft.com/fr-fr/aspnet/core/tutorials/getting-started-with-swashbuckle?view=aspnetcore-5.0&tabs=visual-studio

Pour vos tests css, jquery,html : Le mode développeur de votre navigateur mais aussi jsfiddle https://jsfiddle.net/ ou tout autre outil que vous jugerez utile. Une règle de base : isoler le code ou le composant à tester.