Marvel - Version 1.5.0 - Evaluation

Alexandre GIRARD - Conseiller en Nouvelles Technologies - alexandre.girard@maif.fr

Objectifs

- Evaluation des compétences acquises en développement web
- Evaluation des compétences acquises en bonnes pratiques de développement

Fonctionnalité - Affichage date - Session 1 - Lundi 2 Décembre 2024

- Afficher la date de modification du personnage (donnée **modified** d'un personnage) dans un format lisible par un humain
  - la date de modification est au format ISO 8601 et ne doit pas être modifiée dans le fichier JSON (sauf éventuellement sur un personnage pour tester le cas d'une date non conforme)
- Ajouter l'affichage de cette date dans la liste des personnages et dans le détail d'un personnage
- Appliquer la bonne pratique permettant d'avoir le même comportement sur les deux pages

Fonctionnalité - Affichage date (suite)

Home - About - Contact

Marvel Characters

Order by: Name Order: Ascending

- **Beast** - Jan 13, 2014
- **Captain America** - Apr 5, 2020
- **Deadpool** - Apr 5, 2020
- **Groot** - Oct 17, 2013
- **Hulk** - Jul 21, 2020
- **Iron Man** - Sep 28, 2016
- **Rocket Raccoon** - Jul 17, 2014
- **Silver Surfer** - Nov 7, 2013
- **Thanos** - May 5, 2016
- **Thor** - Mar 11, 2020
- **Wolverine** - Invalid Date

There are 11 characters

Home - About - Contact

Beast

Jan 13, 2014

Capacities

Using D3

Fonctionnalité - Affichage date (suite)

- La librairie **date-fns** permet de manipuler des dates en JavaScript de manière simple et efficace
- Les balises `<strong>` et `<small>` peuvent être utilisées pour mettre en avant le nom du personnage et en retrait la date de modification
- Les tests unitaires devront sûrement être adaptés pour prendre en compte cette nouvelle fonctionnalité

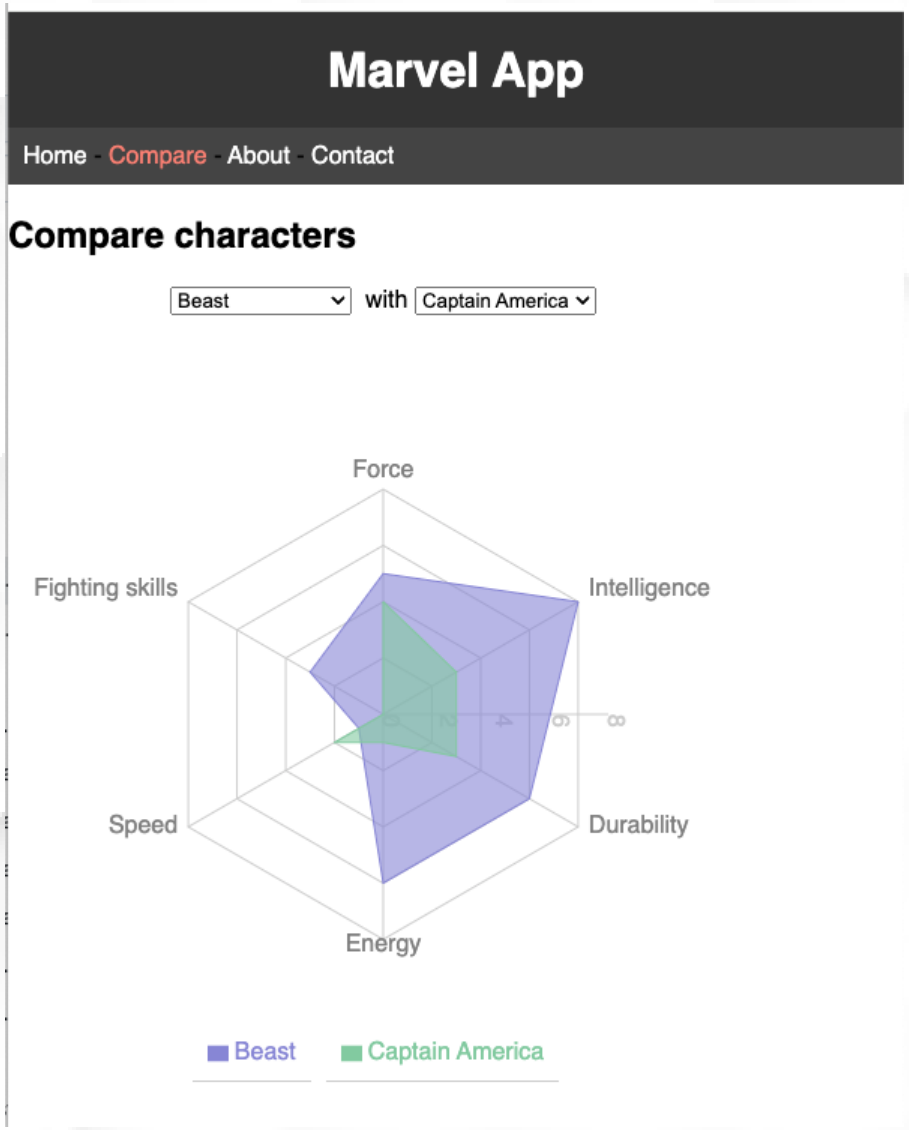
Fonctionnalité - Affichage date - Mise à disposition du code

- Les bonnes pratiques de développement doivent être respectées (commit, nommage, formatage, ...) afin de faciliter la relecture du code et seront évaluées,
- Proposer cette modification à l'équipe de développement (alex1dregirard) pour une éventuelle intégration au projet si elle est jugée pertinente, c'est cette modification qui sera évaluée,
  - Il est nécessaire d'inviter l'utilisateur `alex1dregirard` sur le dépôt GitHub pour qu'il puisse accéder au code
- Les modifications doivent être finalisées en fin de session 1, aucune modification ne sera acceptée après la fin de la session.

Fonctionnalité - Comparaison des personnages - Session 2 - Mardi 3 Décembre 2024

- Ajouter un élément de menu `Compare` permettant d'accéder à une page de comparaison
- La page de comparaison doit permettre de sélectionner deux personnages parmi la liste des personnages et afficher un graphique de type radar comparant les caractéristiques des deux personnages

La page ci-dessous présente le rendu attendu de cette fonctionnalité, la suivante présente un squelette de code pour démarrer le développement de cette fonctionnalité.



Alexandre GIRARD - Conseiller en Nouvelles Technologies - alexandre.girard@maif.fr

```
import React from 'react';

const CompareCharactersPage = () => {
    // change the title of the page
    document.title = "Compare | Marvel App";

    // A supprimer, permet de rendre le composant fonctionnel dans un premier temps
    const characters = [
        {
            name: '...'
        },{
            name: '---'
        }
    ]
    // Fin de la partie à supprimer

    // transform the characters to array of label/value objects
    const options = characters.map((character, index) => ({
        value: index,
        label: character.name,
    }));

    // set the default options to the first two characters
    const [option1, setOption1] = React.useState(options[0]);
    const [option2, setOption2] = React.useState(options[1]);

    const centerStyle = {
        textAlign: 'center',
        width: 500,
    };

    return (
        <>
            <h2>Compare characters</h2>

            <p style={centerStyle}>
                <select
                    data-testid='select-character-1'
                    value={option1.value}
                    onChange={(event) => setOption1(options[event.target.value])}
                >
                    {options.map((option) => (
                        <option key={option.value} value={option.value}>
                            {option.label}
                        </option>
                    ))}
                </select>  {/* Fix the ambiguous spacing */}
                with 
                <select
                    data-testid='select-character-2'
                    value={option2.value}
                    onChange={(event) => setOption2(options[event.target.value])}
                >
                    {options.map((option) => (
                        <option key={option.value} value={option.value}>
                            {option.label}
                        </option>
                    ))}
                </select>
            </p>

            { /* A supprimer, permet le voir comment récupérer les éléments selectionnés */ }
            <p style={centerStyle}>
                {characters[option1.value].name} vs {characters[option2.value].name}
            </p>
            { /* Fin de la partie à supprimer */ }
        </>
    );
};

export default CompareCharactersPage;
```

Fonctionnalité - Comparaison des personnages (suite)

Quelques informations pour vous aider dans le développement de cette fonctionnalité :

- Un exemple de graphique `recharts` de type `RadarChart` est disponible sur la page de documentation de la librairie **recharts**
- Il existe déjà des bouts de code dans l'application pour la navigation et l'affichage des personnages, vous pouvez vous appuyer sur ces éléments pour développer cette fonctionnalité
- Pour l'écriture de la fonctionnalité et la génération des tests unitaires, il est surement plus simple de se baser sur le code existant plutôt que de demander à `github copilot` de générer le code à partir de rien

Une proposition de mise en oeuvre de cette fonctionnalité par étapes :

- Ajouter un élément de menu `Compare` pour accéder à la page de comparaison
  - Appuyer vous sur ce qui existe déjà dans l'application pour la navigation et l'affichage des personnages
- Faite en sorte d'alimenter correctement les deux zones de liste déroulante
- Mettre en oeuvre un composant `RadarChart` pour afficher les caractéristiques des deux personnages passés en paramètre
  - Bien regarder le format des données attendues par le graphique dans l'exemple de la documentation, il faudra surement adapter les données des personnages pour les afficher correctement
- Ajouter des tests unitaires pour valider le bon fonctionnement de la page de comparaison

Fonctionnalité - Comparaison des personnages (suite)

- L'application et les exemples fournis devraient vous permettre de produire une solution fonctionnelle et de qualité, `github copilot` pourra vous aider 🪄 mais il ne remplacera pas votre réflexion 🧠 sur la mise en oeuvre de la fonctionnalité et pourrait même être contre productif si vous lui en demander trop. C'est un assistant, pas un développeur à part entière .

Fonctionnalité - Comparaison des personnages - Mise à disposition du code

- Les différentes étapes de mise en oeuvre pourraient être séparées en commits distincts pour vous aider à structurer votre travail et faciliter la relecture 😉 du reviewer, mais aussi de fournir du code fonctionnel mais pas forcément complet en cas de difficultés
- Petit rappel, une `Pull Request` est un moyen de proposer des modifications à un projet, elle permet de discuter des modifications apportées et de les faire valider avant de les intégrer au projet.

Fonctionnalité - Comparaison des personnages - Mise à disposition du code

- Une `Pull Request` peut-être en mode `draft` le temps de la finaliser, elle peut-être mise à jour autant de fois que nécessaire avant de la finaliser pour relecture.
- Il s'agit d'une nouvelle fonctionnalité indépendante de la première, la première n'ayant pas été "en théorie" intégrée au projet.