

# Recherche de Films et de Séries

## Adaptation des Interfaces à l'Environnement

---

Ce rapport se propose de présenter nos 4 choix technologiques au travers d'un exemple choisi.

### I - Présentation du groupe

Colombet Aurélien - [aurel.colombet@gmail.com](mailto:aurel.colombet@gmail.com)  
Cursus IAM - Foundation

Sibut Clément - [clesib@hotmail.fr](mailto:clesib@hotmail.fr)  
Cursus IAM - Pure CSS

Laborde Quentin - [qlaborde@polytech.unice.fr](mailto:qlaborde@polytech.unice.fr)  
Cursus IAM - Xamarin

Stackler Mathieu - [mathieu.stackler@etu.unice.fr](mailto:mathieu.stackler@etu.unice.fr)  
Cursus IAM - Angular 2

### II - Description de l'application

**Objectifs:** Recherche, ajout/suppression/modification et notation de films et séries

Voici les différentes fonctionnalités mises en place dans l'application :

- 1 - Rechercher des films/séries par genre, titre, réalisateur, acteurs
- 2 - Ajouter/modifier/supprimer un film/série
- 3 - Ajouts de commentaires
- 4 - Lister tous les films
- 5 - Noter un(e) film/série

**Contextes :**

- Utilisateur standard sur PC : assis avec un clavier
- Utilisateur de smartphone assis ou se déplaçant à pieds

**Scénarios:**

Michaël est un cinéphile amateur. Aujourd'hui il a vu le film "L'IHM dans la vie de tous les jours" et de retour chez lui sur son pc, veut le commenter sur le site RFS. Cependant, après **une recherche**, il remarque que le film n'est pas présent et décide donc de **l'ajouter**. Ensuite, il décide de **lister** tous les films présents sur le site, et **ajoute un commentaire et une note** à un autre film qu'il a vu aujourd'hui "L'IHM sans fin".

Damien est quelqu'un de connecté, il se balade toujours avec son smartphone. Aujourd'hui il veut aller au cinéma mais ne sait pas quoi regarder. Il utilise donc le site RFS pour **lister tous les films** à l'affiche dans son cinéma préféré et trouver un qui puisse l'intéresser. Une fois "Les Dents de l'IHM" fini, il décide de le **rechercher** sur le site et de le **noter**.

### **Argumentation:**

Notre application RFS correspond bien à un projet d'adaptation des interfaces à l'environnement. En effet, elle est compatible avec plusieurs supports (notamment Desktop et Mobile dans notre cas) ce qui permet de démontrer des adaptations en fonction de la taille de l'écran via le responsive design.

Nos deux mises en situation permettent de créer une application ayant plusieurs usages, ce qui permet d'avoir ou non certaines fonctionnalités.

La simplicité de l'application nous a permis de nous concentrer sur le développement de l'interface en elle même sans problématiques de backend.

## **III - Tutorials**

### **1 - Foundation**


Pour cette application web j'ai choisi de mettre en place une interface responsive avec les fonctionnalités suivante : recherche de films, lister tous les films, voir la fiche d'un film, ajouter un commentaire et une note sur un film et enfin ajouter un film. J'ai donc utilisé Foundation pour ce projet. Pour l'installation il m'a juste fallu inclure dans ma page HTML template la feuille CSS et le script javascript de foundation. À partir de là le développement a commencé.

#### **1. Développement :**

J'ai commencé par développer le menu qui possède un champ de recherche et deux menus (lister tous les films et ajouter un film). Ensuite j'ai fait la page qui liste tous les films, elle affiche les films sous forme d'éléments qui sont cliquable et affiche des informations sur le film en question (affiche du film, date de sortie, etc...).


Menu Rechercher un film ... Go

### Tous nos films




**Star Wars IV : Un nouvel espoir**  
1977  
De George Lucas  
avec : Mark Hamill, Harrison Ford, Carrie Fisher  
[Voir la fiche du film](#)

---




**Orange Mécanique**  
1972  
De Stanley Kubrick  
avec : Malcolm McDowell, Patrick Magee, Michael Bates  
[Voir la fiche du film](#)

---



**Troll 2**  
1990  
De Claudio Fragasso  
avec : Michael Stephenson, George Hardy, Michael Bates  
[Voir la fiche du film](#)


---




**Les Gardiens de la Galaxie**  
13/08/2014  
De James Gunn  
avec : Chris Pratt, Zoe Saldana, Dave Bautista  
[Voir la fiche du film](#)

RFS  Go Tous les films Ajouter un film

### Tous nos films




**Star Wars IV : Un nouvel espoir**  
1977  
De George Lucas  
avec : Mark Hamill, Harrison Ford, Carrie Fisher  
[Voir la fiche du film](#)




**Orange Mécanique**  
1972  
De Stanley Kubrick  
avec : Malcolm McDowell, Patrick Magee, Michael Bates  
[Voir la fiche du film](#)

---




**Troll 2**  
1990  
De Claudio Fragasso  
avec : Michael Stephenson, George Hardy, Michael Bates  
[Voir la fiche du film](#)




**Les Gardiens de la Galaxie**  
13/08/2014  
De James Gunn  
avec : Chris Pratt, Zoe Saldana, Dave Bautista  
[Voir la fiche du film](#)

---




**Hot Fuzz**  
18/07/2007  
De Edgar Wright  
avec : Simon Pegg, Martin Freeman, Bill Nighy  
[Voir la fiche du film](#)



**H2G2 : le guide du voyageur galactique**  
17/08/2005  
De Garth Jennings  
avec : Martin Freeman, Yasin Bey, Sam Rockwell  
[Voir la fiche du film](#)

---



lol  
09/10/2016  
De zacdazdc  
avec azdczaddc  
[Voir la fiche du film](#)

Figure : Comparaison entre Desktop et Mobile pour l'affichage de tous les films

Ensuite j'ai fait la fiche de détail du film qui va afficher plus d'informations sur celui-ci. Il y a aussi en dessous une partie sur les commentaires à propos de ce film et un petit formulaire pour poster son propre commentaire avec sa note. Ensuite il a fallu faire la page d'ajout de film, elle comprend un formulaire qu'il faut remplir avec les informations du film que l'on veut ajouter. Ensuite il suffit de cliquer sur "ajouter" ce qui va rediriger vers la page de détail du film que l'on vient de créer.

Toutes les pages sont responsives et s'adaptent au format de l'écran de l'utilisateur, par exemple sur la page qui affiche tous les films, si on est en format desktop cela va afficher deux films par lignes alors qu'en format mobile il n'y aura qu'un film par ligne.

## 2. Outils de développement et de tests :

Les outils de développement que j'ai utilisé sont : sublime text 3 pour écrire le code et firefox pour tester. J'ai du aussi utiliser wamp pour tester mon application sur google chrome et internet explorer car j'ai utiliser des requêtes ajax pour la navigation entre les pages et donc cela m'empêchait de tester sur ces navigateurs à cause des requêtes cross-domains. Pour les tests j'ai testé avec firebug et l'outils de développement de firefox.

## 3. Comment déployer ?

Pour déployer mon exemple il suffit d'ouvrir le fichier "index.html" avec firefox pour tester en local. Si on veut tester avec google chrome ou internet explorer, il faut installer un serveur local et ouvrir l'index.html à partir de l'url donné par le serveur en local (par exemple : localhost:8080/index.html). Si nous voulons le déployer sur un serveur distant il suffirait de déposer les fichiers sur le serveurs et d'accéder directement à l'index.html.

Pour tester les capacités d'adaptation il suffit d'utiliser l'outils de développement de firefox ou d'un autre navigateur, qui permet de choisir la taille de l'écran. On peut aussi directement mettre en ligne le site web sur un serveur et y accéder avec notre smartphone pour tester sur un vrai appareil.

## 4. Avantages et inconvénients

### Avantages

- Facilité et rapidité (de mise en place et développement)  
Uniquement du HTML, CSS et javascript, donc très facile à trouver les bugs (sauf pour javascript certaines fois) mais lorsqu'on a l'habitude la vitesse de développement est très rapide.
- Système de grid pour le responsive design  
Le système de grille de foundation est vraiment pratique à utiliser et vraiment simple.
- Classes CSS et javascript utiles  
Là il y a pas mal de javascript utiles comme des datePicker, ce qui est vraiment pratique.
- Permet une personnalisation du CSS simple  
La personnalisation du CSS est beaucoup plus simple sur foundation que sur Bootstrap par exemple. J'avais déjà essayer de personnaliser un bootstrap c'était vraiment compliqué par rapport à la personnalisation de foundation.

- Documentation précise  
La documentation est vraiment précise sur le site web de foundation.

#### Inconvénients

- Ne supporte pas IE 8  
Le fait qu'il ne supporte pas IE8 limite les utilisateurs de Windows XP SP2 à ne pas pouvoir utiliser le site web. Mais l'utilisateur peut installer un autre navigateur pour utiliser le site web, même si cela est un inconvénient pour l'utilisateur. Sur les anciens Windows phone, qui avait une version d'IE 6 et 7 ne marcherons donc pas non plus. Les Windows phone maintenant offre un IE 10.
- Peu de templates exemples  
Foundation nous propose peu de templates exemple sur leurs site web. En effet Bootstrap son concurrent propose plus de template exemple ce qui permet de commencer un développement spécifique plus rapide.
- Le support mobile est moins bien géré que celui de Bootstrap par exemple  
En effet le support mobile de Foundation est moindre que celui de Bootstrap. Les comportements sur mobile avec foundation sont moins bien géré que ceux sur Bootstrap, il faut juste bien debugger sur smartphone et tablette avec une application faite avec Foundation.

#### 5. Conclusion sur Foundation

Foundation est un Framework CSS qui dans le cadre de RFS peut être une bonne idée, en effet le fait qu'il soit vraiment responsive pour les différents format d'écran est vraiment un plus, car il permet aux différents utilisateurs d'appareils différent de consulter le site web. Malgré le support IE8 qui n'est pas respecté une grande partie des utilisateurs est impacté et donc peut utiliser notre application.

## 2 - Pure CSS

### 1) Fonctionnalités



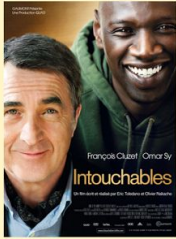

Pour la réalisation de cette application web, j'ai choisi de développer l'ensemble des fonctionnalités mis à part la modification des fiches des films et séries. La plupart des fonctionnalités ne sont pas opérationnelles. En effet, j'ai préféré ne pas utiliser JavaScript pour implémenter les fonctionnalités mais uniquement pour permettre la réalisation d'une interface adaptable.

### 2) Développement

Pour naviguer à travers le site et proposer les différentes fonctionnalités j'ai mis en place un menu. Il pourra être étoffé par la suite lorsque l'on voudra ajouter de nouvelles fonctionnalités. Lorsque l'utilisateur dispose d'un écran d'une assez large résolution, le menu est affiché horizontalement et entièrement. A partir de ce menu il peut alors naviguer à travers les différentes rubriques. Il peut aussi lancer une recherche, quand l'utilisateur en

lance une, elle est effectuée sur l'ensemble des critères, c'est à dire qu'on recherche à la fois parmi les titres des films mais aussi les réalisateurs et acteurs par exemple. La partie recherche n'est pas encore fonctionnelle car elle aurait nécessité l'utilisation de JavaScript pour afficher dynamiquement le résultat de la recherche. La page possède un point de rupture à 768px car en dessous de cette largeur, l'affichage horizontal du menu n'est plus pertinent. J'ai alors testé 2 alternatives, adapter le menu pour les basses résolution en utilisant uniquement du CSS ou alors utiliser Javascript. J'ai finalement choisi de le faire avec JQuery et le plugin SlickNav car la solution basée uniquement sur CSS possède de trop grands inconvénients. Elle semble difficile à maintenir et risque de présenter des problèmes d'affichages sur les navigateurs un peu plus anciens. Slicknav a pour avantage de ne pas polluer le code HTML avec toutes les classes CSS nécessaires pour réaliser un menu adaptable. Utiliser un plugin JQuery comme SlickNav est très simple, il suffit d'utiliser la balise script dans un fichier HTML pour importer le fichier Javascript et la balise link pour importer le style CSS correspondant. Le menu peut donc s'ouvrir verticalement lorsqu'on appuie sur les 3 barres sur les écrans basses résolutions. Pure CSS m'a alors fournis des classes CSS pour faciliter la modification de l'apparence du menu et de mes boutons. J'ai également utilisé la bibliothèque CSS Font-Awesome qui fournit de nombreuses icônes.

RFS Accueil Ajouter un film Lister tous les films Contact  Q

<p><b>La grande vadrouille (1966)</b></p>  <p>De Gérard Oury</p> <p>Avec Bourvil, Louis de Funès</p> <p>Genres : Comédie</p>	<p><b>Titanic (1998)</b></p>  <p>De James Cameron</p> <p>Avec Leonardo DiCaprio, Kate Winslet</p> <p>Genres : Drame, Romance</p>	<p><b>Intouchables (2011)</b></p>  <p>De Eric Toledano, Olivier Nakache</p> <p>Avec François Cluzet, Omar Sy, Anne Le Ny</p> <p>Genres : Comédie</p>	<p><b>Les Aristochats (1971)</b></p>  <p>De Wolfgang Reitherman</p> <p>Avec Claude Bertrand, Michèle André, Roger Carel</p> <p>Genres : Animation, Aventure, Comédie</p>
---	---	--	---

RFS

LE FILM PRÉFÉRÉ DES FRANÇAIS FÊTE SES 50 ANS !  
 BOURVIL LA GRANDE VADROUILLE LOUIS DE FUNES  
 GÉRARD OURY  
 NOUVELLE VERSION RESTAURÉE  
 EN 4K DANS NOS CINÉMAS À PARTIR DU 29 JUILLET

**La grande vadrouille (1966)**

De Gérard Oury

Avec Bourvil, Louis de Funès

Genres : Comédie

LEONARDO DICAPRIO KATE WINSLET  
 JAMES CAMERON  
 TITANIC  
 VIVEZ L'EXPÉRIENCE COMME JAMAIS AUPRÈS  
 EN 3D  
 LE 4 AVRIL AU CINÉMA

**Titanic (1998)**

De James Cameron

Avec Leonardo DiCaprio, Kate Winslet

Genres : Drame, Romance

**Intouchables (2011)**

**Les Aristochats (1971)**

*Comparaison de l'affichage de la liste des films pour un écran haute résolution et un écran basse résolution*

Une fois la recherche effectuée, les résultats s'affichent. Les films sont présentés sous forme de fiche contenant l'affiche, le titre, le réalisateurs, les acteurs principaux et le genre. Chaque fiche correspond à un bloc et le nombre de blocs à afficher sur la largeur de la page dépend de la résolution de l'écran. Sur les écrans les plus larges 4 films sont affichés, lorsque l'on passe en dessous du point de rupture de 1024 de large , on en affiche plus que 2, les autres se décalent en dessous. Si l'écran est moins large que 768px on en affiche plus qu'un seul par ligne. Pure CSS est adapté à la gestion du positionnement des blocs sur l'écran car il possède un système de grille. Il suffit d'indiquer la largeur en nombre de blocs pour chaque élément en fonction de la largeur de l'écran.

Lorsqu'on affiche les détails d'un film, ceux-ci sont réorganisés pour les écrans ayant une largeur inférieure à 1024px. De même, pour ajouter un film, les formulaires sont ajustés en fonction de la largeur de l'écran.

Ces points de rupture de 768px et 1024px permettent la réorganisation de l'affichage et l'adaptation aux différentes résolutions d'écran notamment celles des tablettes et portables. Les points de ruptures sont soit créés automatiquement par Pure CSS (pour l'affichage des films) soit à partir de media queries CSS (pour le passage du menu large résolution au menu petite résolution).

PureCSS aide donc à construire une interface adaptable mais parfois tout réaliser en CSS n'est pas la solution la plus appropriée et Javascript avec les plugins JQuery peuvent être d'une grande aide pour adapter certains éléments.

### 3) Outils utilisés

Le développement ne nécessite pas d'outils particuliers, un simple éditeur texte comme Notepad++ peut suffire. Pour les projets de plus grande envergure où l'on utilise plusieurs technologies à la fois comme Bootstrap et JQuery, un IDE comme WebStorm peut être utilisé et Yeoman pour mettre en place la structure du projet.

### 4) Tests

Pour tester l'adaptation de l'interface j'ai utilisé le site [www.responsinator.com](http://www.responsinator.com). Ce site permet de vérifier que notre interface s'adapte bien à tous les types d'écrans en simulant le résultat sur des téléphones cibles. On peut également tester les points de ruptures en redimensionnant le navigateur, cela nous permet de vérifier que les changements sont cohérents et de savoir à quoi peut ressembler l'application pour toutes les tailles d'écrans contrairement à responsinator qui vérifie l'apparence de notre application sur les smartphones les plus courants. Enfin, j'ai testé l'application sur mon smartphone après l'avoir déployé sur un serveur web.

### 5) Déploiement

Mon projet ne comporte que des technologies exécutables directement par un navigateur, il n'y a donc pas de difficulté liée au déploiement. J'ai utilisé un client FTP, FileZilla pour placer l'application sur mon serveur web.

## 3 - Xamarin

Pour adapter notre exemple sur Xamarin, qui est un technique cross platform sur smartphone, je n'ai implémenté qu'une partie de fonctionnalités définie ci-dessus pour s'adapter à la taille du dispositif mais aussi au contexte d'utilisation souvent non propice à l'utilisation d'une application complexe (en déplacement, dans les transports, etc.). Je n'ai donc pas implémenté la fonctionnalité numéro 2 (Ajouter/modifier/supprimer un film /série) qui ne peut pas être facilement réalisé et utilisé sur une interface de smartphone ou tablet.

De plus, les données de l'application sont ajoutés en dure dans l'application donc avec aucune persistance lorsque l'on éteint l'application.





#### a) Développement et déploiement

Pour développer cette application, j'ai utilisé Visual Studio for Xamarin sur window 7 avec du code en C#. Visual studio intègre des templates Xamarin (depuis le rachat de xamarin par Microsoft) qu'on peut utiliser pour générer un helloWord à compléter.

Visual Studio inclut un gestionnaire de package (NuGet) pour importer les bibliothèques que l'on veut utiliser. Dans mon cas, j'ai importé la dernière version stable de Xamarin Forms et toutes ses dépendances (télécharger automatiquement). Il faut aussi installer la SDK Android avec la bonne version d'OS ainsi que la JDK java 1.8.

Pour la génération de l'application, il suffit d'indiquer la platform cible parmi les sous-projets de la solution puis faire un build et un déploiement soit sur un émulateur soit, comme dans mon cas, sur mon smartphone directement (Android 6.0).

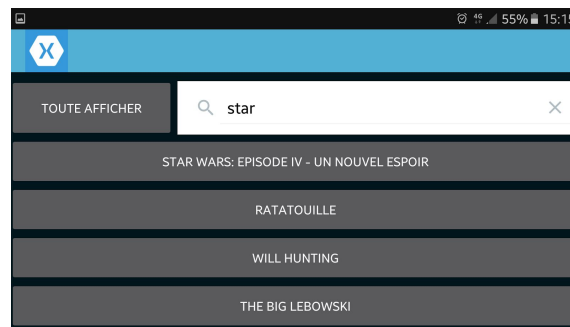
Pour déployer l'application, puisque Xamarin génère des applications natives, il est possible de passer par les stores respectifs de chaque plateforme. Le lancement de l'application ce fait comme n'importe quelle application mobile.

#### b) Description de l'application

Vidéo de démonstration : <https://youtu.be/z2VzJdzK4tQ>

Mon application s'organise sur 4 pages qui peuvent tous être utilisé en mode portrait ou paysage avec un redimensionnement des éléments et avec, dans certains cas, un contenu responsif : la page de démarrage permet d'afficher la liste des films disponibles dans l'application, un bar de recherche permet de filtrer la liste et un

bouton (“toute afficher”) permet d’afficher de nouveau la liste des films au complet si on a fait une recherche entre-temps (qui cache une partie des films). La barre de recherche utilise le clavier virtuel de smartphone avec, en plus des touches traditionnelles, une reconnaissance vocale pour dicter sa recherche.

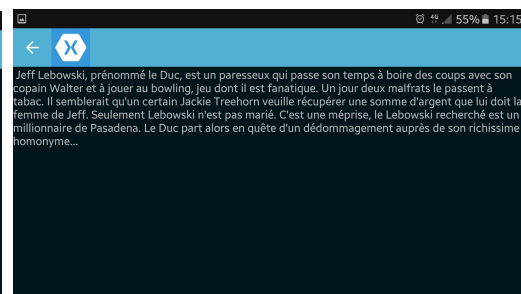
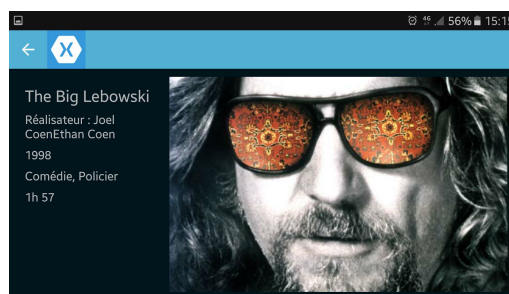
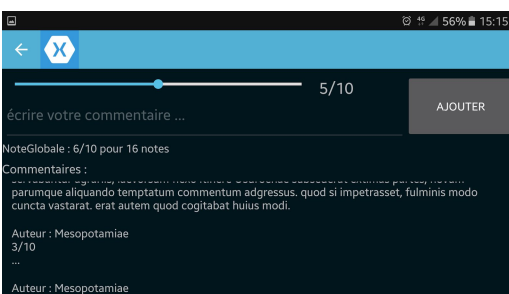
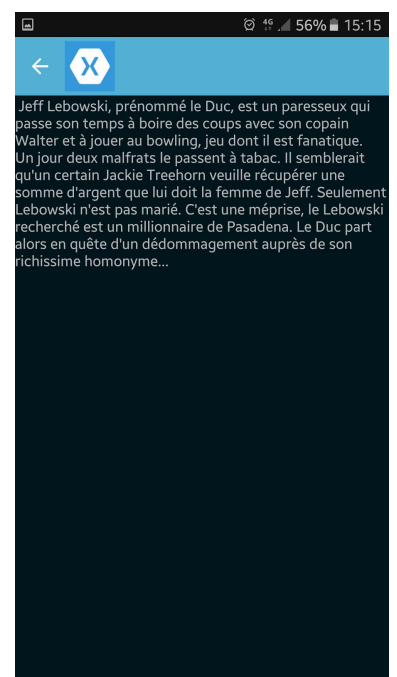
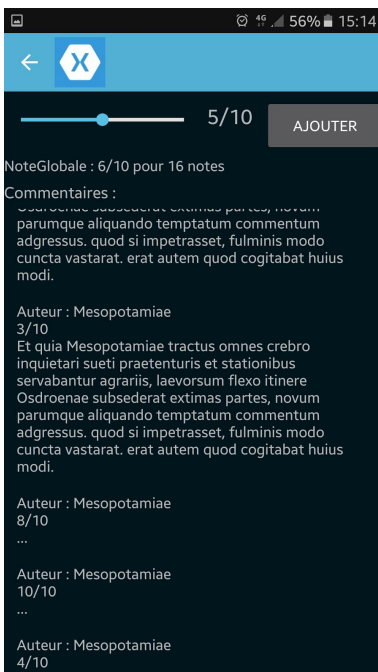


Chaque film listé est un bouton qui permet d’ouvrir la page générale sur le film en question. Cette page affiche les informations générales sur le film avec son affiche. Les informations et l’affiche sont séparées en deux layouts qui sont l’un en dessous de l’autre en mode portrait et côte à côte en mode paysage. Le reste des informations est disposé sur deux autres pages afin de ne pas surcharger les pages et de garder les informations visibles et claires. Pour alterner entre ces trois pages, il suffit de faire glisser la page à gauche ou à droite : j’ai utilisé une CarouselPage qui place les trois pages côte à côte avec de gauche à droite la page de “commentaires/notations”, la page d’informations générales (c’est la page de départ) et la page de la description. La page de la description est un simple label qui affiche la description. La page “commentaires/notations” permet à l’utilisateur, en mode portrait, d’ajouter une notation et de consulter les commentaires et la note globale du film. En mode paysage, l’utilisateur peut en plus ajouter un commentaire joint à sa note avec un champ texte disponible uniquement dans ce mode (car le clavier est plus facilement accessible et les conditions d’utilisation de la platform en mode paysage (assis, utilisation des deux mains) sont différent de celle en mode portrait). Il est également possible de dicter son commentaire comme pour la recherche.

Il est aussi possible de naviguer entre ces trois pages grâce à l’accéléromètre qui enregistre les accélérations sur l’axe X donc à droite ou à gauche en mode portrait. Il suffit d’agiter son smartphone à droite ou à gauche pour changer de page. Cette fonction n’est pas disponible en mode paysage car c’est non adapté si on utilise son smartphone avec les deux mains.

J'ai utilisé une NavigationPage donc on peut revenir à tout moment à la page de recherche grâce à un bouton retour (en haut) disponible sur les trois pages décrivant un film ou en utilisant le bouton retour du smartphone.

Comme vous avez pu le voir, j'utilise en tant qu'Input pour mon interface l'écran tactile, l'accéléromètre et le gyroscope.



### c) Tests

Pour mon application, j'ai effectué de simples tests d'utilisations avec comme plan le scénario d'utilisation version mobile (avec Damien). Le C# est un langage objet, il est donc possible de faire des tests unitaires avec xamarin sur le même modèle que les tests Java, mais dans le cas de cette application qui ne contient pas vraiment de partie backend, ce n'est pas utile.

### c) Justification du choix de la plate-forme

Je développais mon application uniquement à destination des cibles Android (SDK 5.0) pour plusieurs raisons. L'utilisation de l'accéléromètre dans mon interface m'oblige à développer une partie de mon code sur le projet natif (Xamarin.Droid) pour l'utiliser, ce qui complique la tâche en cas de multi platform. Je n'ai pas accès au matériel adéquat : il y a beaucoup de contraintes pour développer une application IOS notamment avec le matériel (obligation d'avoir un mac). Le développement pour une cible Windows Phone est compliqué lui aussi et inutile vu la part de marché.

Cependant, je pense que l'adaptation de la majorité de mon application (sans l'utilisation de l'accéléromètre) est faisable grâce à la mise en commun du code.

## 4 - Angular 2

### a. Angular 2 en quelques mots

Angular 2 est un framework bâti avec TypeScript permettant de créer des applications web et mobile. Il se caractérise par une augmentation des performances via les web components, une amélioration de la productivité via TypeScript et une adaptation au mobile via le "responsive design".

Angular 2 repose sur l'utilisation de Web Components, ces derniers permettent de créer des projets très bien structurés, ce qui apporte une meilleure vitesse de développement, une modification et une maintenabilité plus aisées.

Chaque Web Component est composé de 4 fichiers : un fichier HTML, un fichier CSS ainsi que deux fichiers TypeScript : un pour la logique et un pour les tests.



Figure : logo d'Angular 2

### b. Mise en place d'Angular 2 avec Angular-cli

Pour ce projet, j'ai choisi d'utiliser Angular Cli qui est un générateur mettant en place tous les outils nécessaires à l'utilisation d'Angular 2, dont un serveur web.

Note : tutoriel pour Windows 7+, Angular CLI ne fonctionne pas sur mon Linux.

- Mise en place de l'environnement

Installer nodejs : <https://nodejs.org/en/>

```
> npm install -g angular-cli typescripts
```

- Création du projet

```
> ng new my_application
```

```
> cd my_application
```

```
> ng serve
```

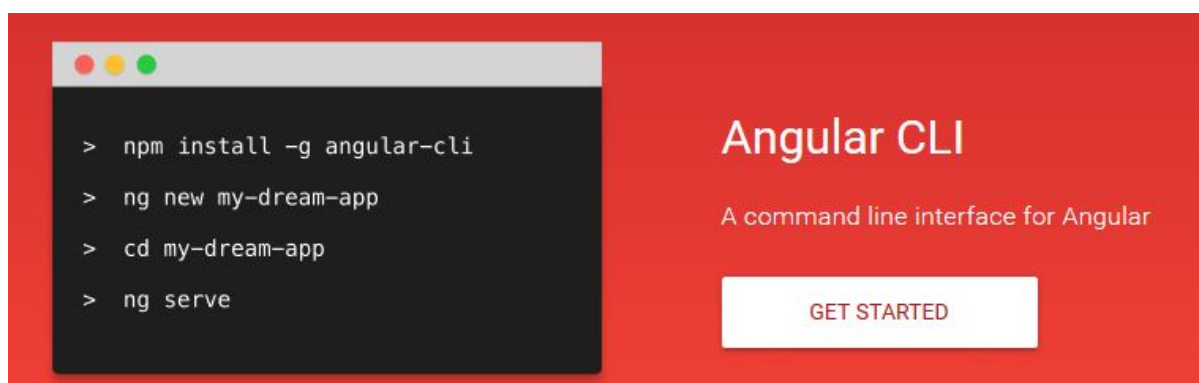


Figure : page d'accueil d'Angular Cli

Ces commandes permettent de créer le projet et de lancer le serveur, si tout s'est bien passé, en ouvrant un navigateur web à l'adresse suivante <http://localhost:4200/>, une page affichant "app works" s'affiche.

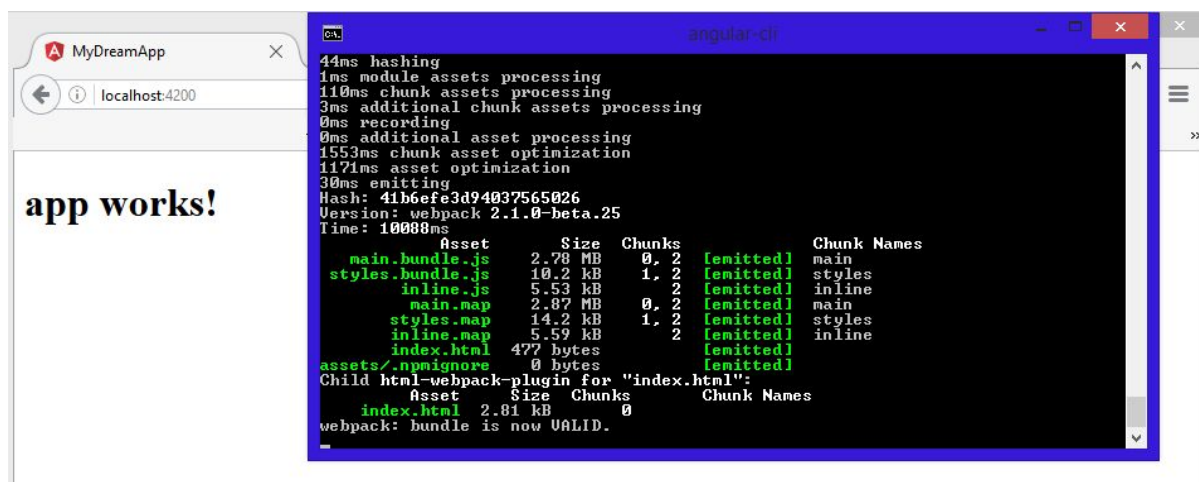


Figure : lancement de l'application MyDreamApp

- Création des différents éléments

```
> ng g component my_new_component
```

Angular cli crée automatiquement toutes les dépendances lors de la création d'un nouveau composant. La manipulation est la même pour la création de services, directives, classes, ...

```
C:\Users\Stackler\Desktop\Projet\my-dream-app>ng g new component my_new_component
Invalid blueprint: new
C:\Users\Stackler\Desktop\Projet\my-dream-app>ng g component my_new_component
installing component
create src\app\my-new-component\my-new-component.component.css
create src\app\my-new-component\my-new-component.component.html
create src\app\my-new-component\my-new-component.component.spec.ts
create src\app\my-new-component\my-new-component.component.ts
```



Figure : création d'un composant

➤ Déploiement de l'application

```
> ng build
```

Cette commande permet de générer toutes les ressources nécessaires au déploiement sur un serveur web dans le dossier dist/ de l'application.

c. *Application à RFS*

➤ Réalisation de l'exemple

Comme expliqué précédemment, RFS permet à un utilisateur de faire un ensemble d'actions : lister les films, ajouter un film, mettre un commentaire, mettre une note et enfin faire une recherche. Comme les actions ne sont pas les mêmes en fonction du support, j'ai choisi de faire correspondre chaque action à un web component.

Prenons un exemple : l'utilisateur de smartphone ne peut pas ajouter un film alors que celui sur desktop le peut. Le bouton ajouter un film qui correspond au web component "ajout-film" n'est alors présent que sur l'application desktop. L'utilisation de média queries via HTML5 et CSS présents sous Angular 2 permettent de rendre utilisable ou non un composant en fonction (dans cet exemple) de la taille de l'écran.



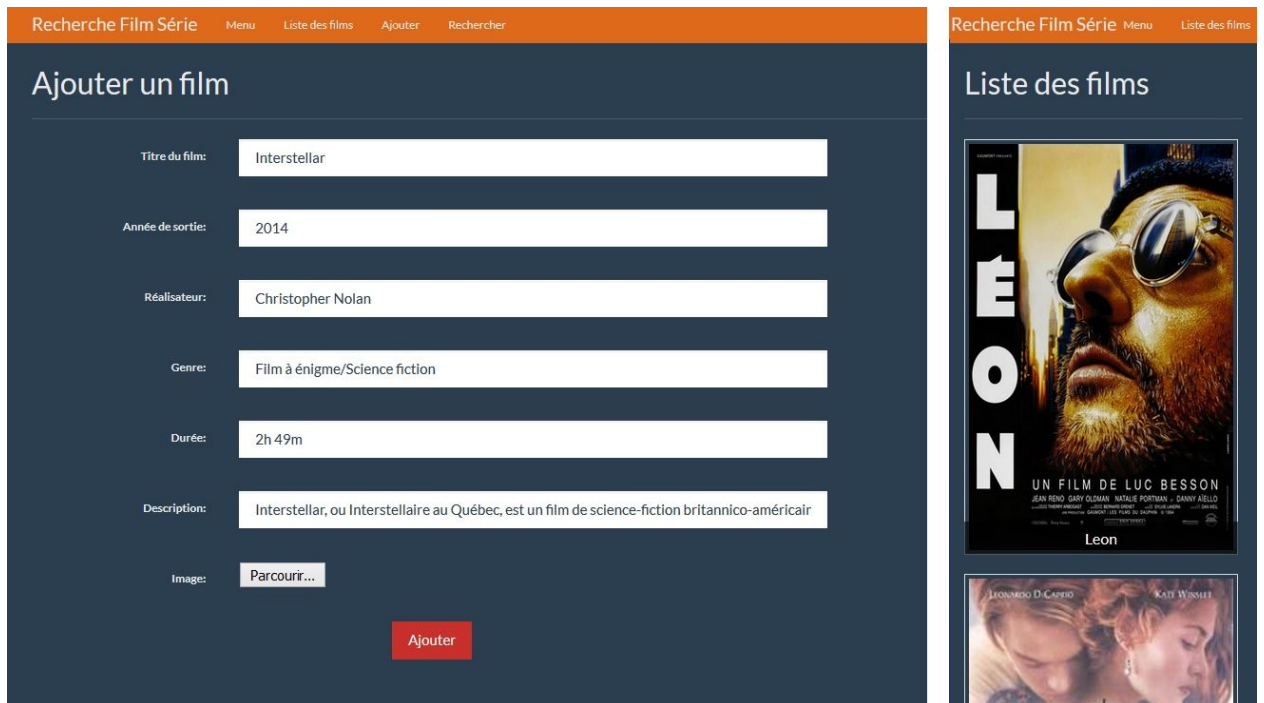


Figure : Comparaison entre Desktop et Mobile pour une fonctionnalité non partagée

Pour une action partagée par tous les supports, la force d'Angular 2 est qu'il suffit d'écrire le web component pour un support et on peut le réutiliser pour les autres supports. Par exemple l'action de lister tous les films est la même pour le desktop et pour le mobile (avec utilisation de responsive design).

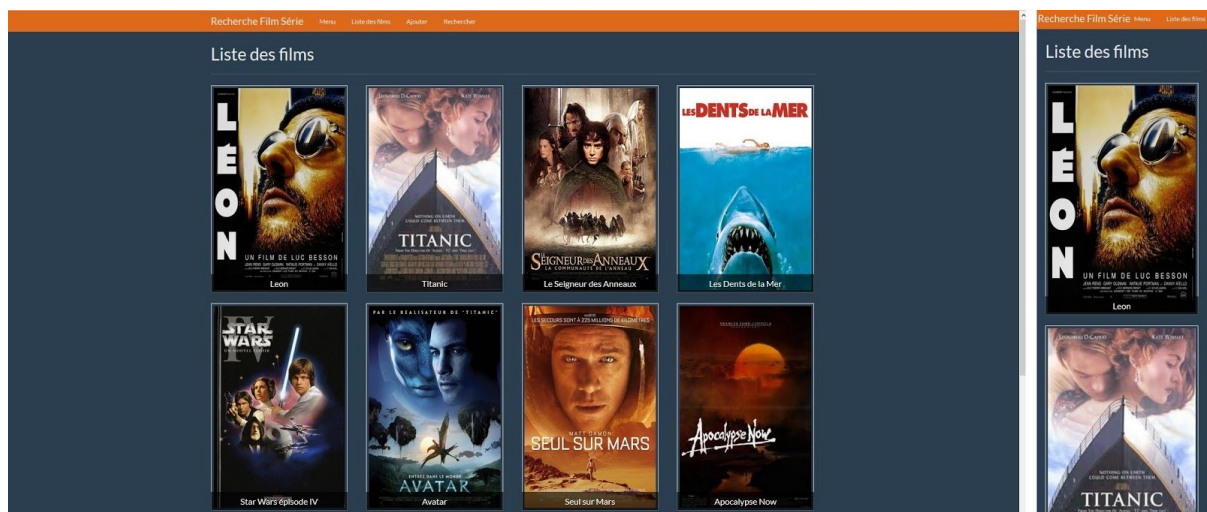


Figure : Comparaison entre Desktop et Mobile pour une fonctionnalité partagée

Pour RFS j'ai ainsi créé 6 composants (5 pour les actions et 1 pour la navigation au sein de l'application) qui seront appelés ou non en fonction de la taille de l'écran.

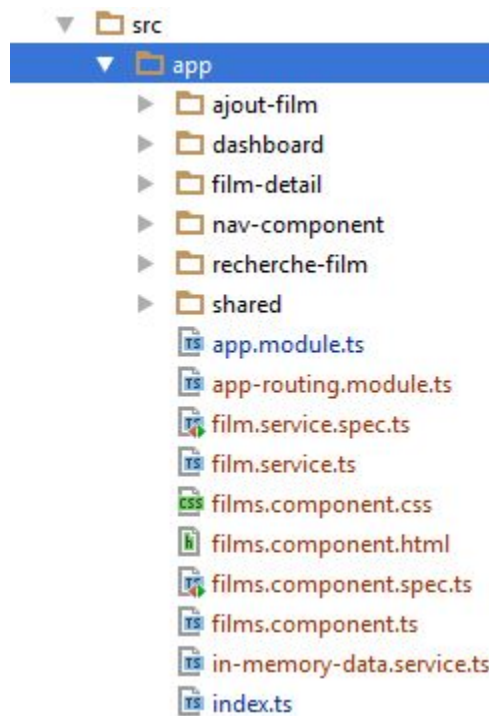


Figure : architecture de l'application

#### ➤ Outils de test

Angular cli permet de gérer les tests pour chaque composant en utilisant Karma. En effet, à la création d'un composant, il va créer un fichier mon-composant.spec.ts qui est utilisé pour les tests.



```
app.component.spec.ts x
1  /* tslint:disable:no-unused-variable */
2
3  import ...
4
5
6  describe('App: MyDreamApp', () => {
7    beforeEach(() => {
8      TestBed.configureTestingModule({
9        declarations: [
10         AppComponent
11       ],
12     });
13   });
14
15   it('should create the app', async(() => {
16     let fixture = TestBed.createComponent(AppComponent);
17     let app = fixture.debugElement.componentInstance;
18     expect(app).toBeTruthy();
19   }));
20
21   it(`should have as title 'app works!'`, async(() => {
22     let fixture = TestBed.createComponent(AppComponent);
23     let app = fixture.debugElement.componentInstance;
24     expect(app.title).toEqual('app works!');
25   }));
26
27   it('should render title in a h1 tag', async(() => {
28     let fixture = TestBed.createComponent(AppComponent);
29     fixture.detectChanges();
30     let compiled = fixture.debugElement.nativeElement;
31     expect(compiled.querySelector('h1').textContent).toContain('app works!');
32   }));
33 });
34
```

Figure : tests créé par défaut par Angular cli

**> ng test**

Il suffit ensuite d'utiliser cette commande pour lancer nos tests.

- > Déploiement de l'application

Voir partie précédente.

- > Tests des capacités d'adaptation

Pour tester l'adaptation, on peut soit lancer l'application dans un navigateur Web et réduire la page pour faire varier la taille de l'écran ou bien tout simplement ouvrir l'application avec un navigateur sur mobile.

#### d. Avantages et Inconvénients

Angular 2 ayant été créé en remettant à plat de nombreux concepts présents dans Angular 1, il présente de nombreux avantages :

- Une augmentation des performances en se reposant sur les briques natives des navigateurs, notamment les Web Components (10 fois plus performant qu'Angular 1, voir graphe).
- Une amélioration de la productivité en utilisant typescript plutôt que le javascript.
- Une adaptation au mobile selon deux points : une empreinte mémoire réduite ainsi que le responsive design
- Une très forte communauté
- Un projet bien structuré
- Modification et maintenabilité plus aisées grâce aux web components
- Tout reste à apprendre et à créer
- Un respect des nouveaux standards du Web

Par rapport à l'adaptation :

La réutilisation des Web Components permet d'adapter en fonction de la plateforme ciblée en réduisant les coûts de développement.

Angular 2 permettant d'utiliser des technologies de Responsive Design tel que BootStrap, il permet une adaptation simple en fonction de la taille de l'écran.

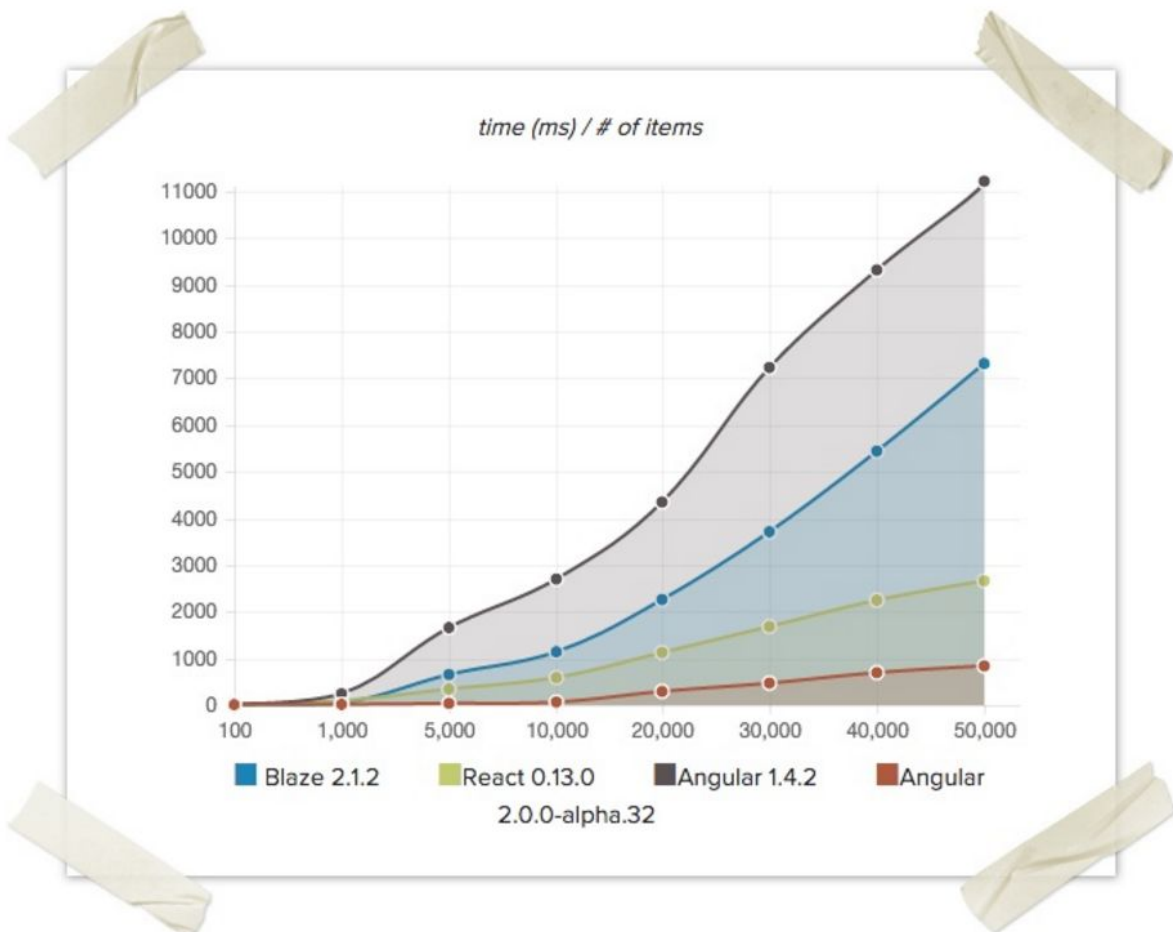


Figure : performance de différents frameworks  
Source : <http://blog.xebia.fr/2015/12/14/angular-2-presentation/>

D'un autre côté, Angular 2 étant une technologie étant encore jeune (la release datant d'à peine quelques mois), elle rencontre encore quelques faiblesses :

- Un manque de stabilité
- Un manque de visibilité pour le futur
- Les composants créés pour Angular 2 ne fonctionnent que sur Angular 2
- Une migration compliquée d'Angular vers Angular 2
- Les anciens navigateurs ne sont pas supportés

Par rapport à l'adaptation :

Angular 2 peut être lourd à être mis en place. L'installation d'un environnement de développement (ici Angular Cli) pour créer une simple application peut s'avérer lourde. De plus, le chargement d'une page reste plus long qu'avec du pure CSS ou du Bootstrap seul.

#### *e. Conclusion sur Angular 2*

Dans le cadre de RFS, Angular 2 est un bon choix. Cette technologie permet de s'adapter aussi bien à une version Desktop qu'à une version mobile via les medias queries et le responsive design. En ajoutant à cela la réutilisation des Web Components permettant de pouvoir créer une application plus rapidement, les avantages sont nombreux. On pourra simplement souligner le fait que l'utilisation des capteurs sur mobile n'est pas supportée. Angular 2 semble donc avoir un bel avenir devant lui.

## IV - Conclusion

- Comparaison Web app - application native / cross plateformes.

Pour l'application RFS, les Web App semblent être un meilleur choix. En effet le surplus de données, une interface peu pratique pour faire toutes les fonctionnalités (comme par exemple le fait d'être obligé d'enlever des fonctionnalités car non adaptables au Mobile) ou encore le fait que l'utilisation de capteurs n'est pas absolument nécessaire sont autant d'arguments contre le cross plateforme.

- Comparaison Foundation - Pure CSS

La technologie choisie par Aurélien, Foundation, est un framework qui intègre de base Javascript et permet d'utiliser des plugins JQuery. On retrouve un système de grille et des styles prédéfinis comme pour Pure CSS. Certains modules de Foundation utilisent Javascript alors que ce n'est pas le cas de Pure CSS. Cependant, même si Pure CSS n'utilise pas JavaScript il s'intègre très bien avec les autres technologies comme JQuery et Bootstrap.

- Comparaison entre Foundation, Pure CSS et Angular 2

Angular 2 est une technologie permettant de s'adapter aussi bien à une version Desktop qu'à une version Mobile. Elle reprend le meilleur des deux autres technologies (utilisation de Bootstrap et de HTML5/CSS) en y ajoutant les Web Components, ce qui en fait à l'heure actuelle une des meilleures technologies pour développer des web app.

Foundation quant à lui permet de faire du responsive design tout comme Pure CSS, il sont très légers mais ne permettent pas de faire tout ce que Angular 2 permet, avec les Web Components. Foundation reste cependant un outils très performant pour développer des web app, Pure CSS également si on l'associe à du JavaScript pour remédier à la difficulté de développer et maintenir une interface écrite uniquement en CSS.