# Analyse

https://www.believemusic.com/

# Le site en quelques mots

Préambule :

Believe est une entreprise avec  un ensemble de services qui va permettre de distribuer / promouvoir de la musique (tout type de formats) sur toutes les plateformes du monde.
Avant distribution, en respectant les normes des différents stores.
Après distribution en fournissant aux clients des outils d'analyses performant
Mais l'accompagnement de l'artiste est l'un des cœurs du métier que ce soit de part un accompagnement marketing ou stratégique.

Le Website de Believe est un site vitrine (très visuel au niveau des images) qui veut mettre en avant les éléments suivants :

-entreprise internationale
-entreprise innovante
-entreprise avec des valeurs
-entreprise qui offre de l'emploi
-entreprise dans laquelle on aime travailler

# A tester en priorité

**-Tester que le changement de langue est bien effectif**
➔ le site est international

**-Tester que chaque lien fonctionne bien**
➔ on souhaite montrer un site fonctionnel et fiable tout comme nos services

**-Tester le temps de réponse**
➔une page longue à afficher est un utilisateur en moins

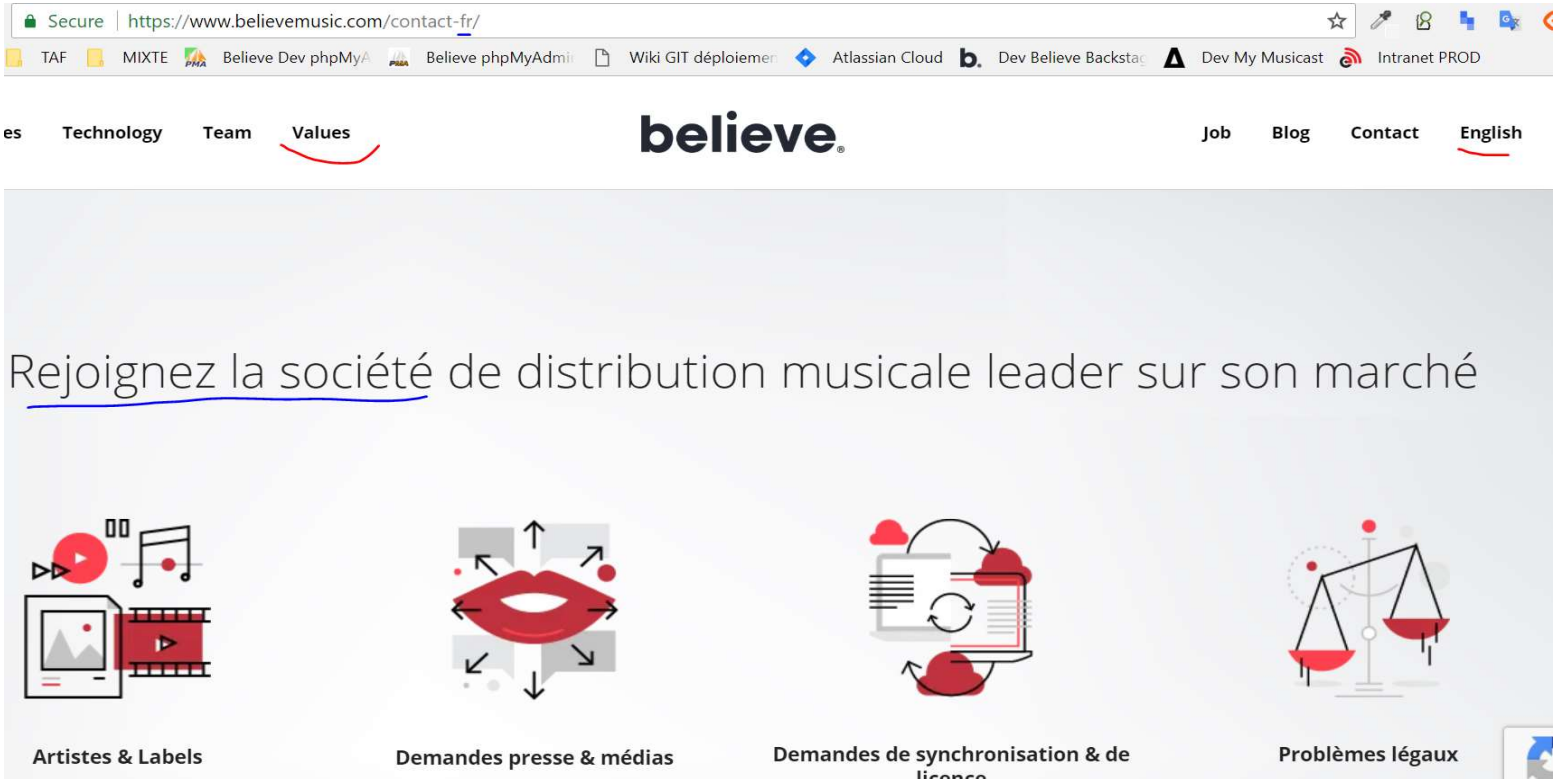
**-Tester que le titre de la page est en adéquation avec l'url**
➔on souhaite respecter les normes html et la lecture robotisée pour les non-voyants

**-Tester que les images s'affichent correctement**
➔le site est rempli d'images mises en avant. Une image cassée saute aux yeux et donne une idée de travail baclé

**-Tester que les envoies de formulaire fonctionnent (pas de failles xss non plus)**
➔c'est la page qui va permettre aux clients de nous contacter pour la première fois. Il faut que ça marche.

**-Tester que les mailto html correspondent bien au texte affiché**
➔il ne faut pas se tromper sur les liens des contacts car en interne cela pourrait être confusant.

Exemple d’erreur (page en fr qui affiche de l’anglais) :
https://wwwbelievemusic.com/contact-fr/



# Three amigos

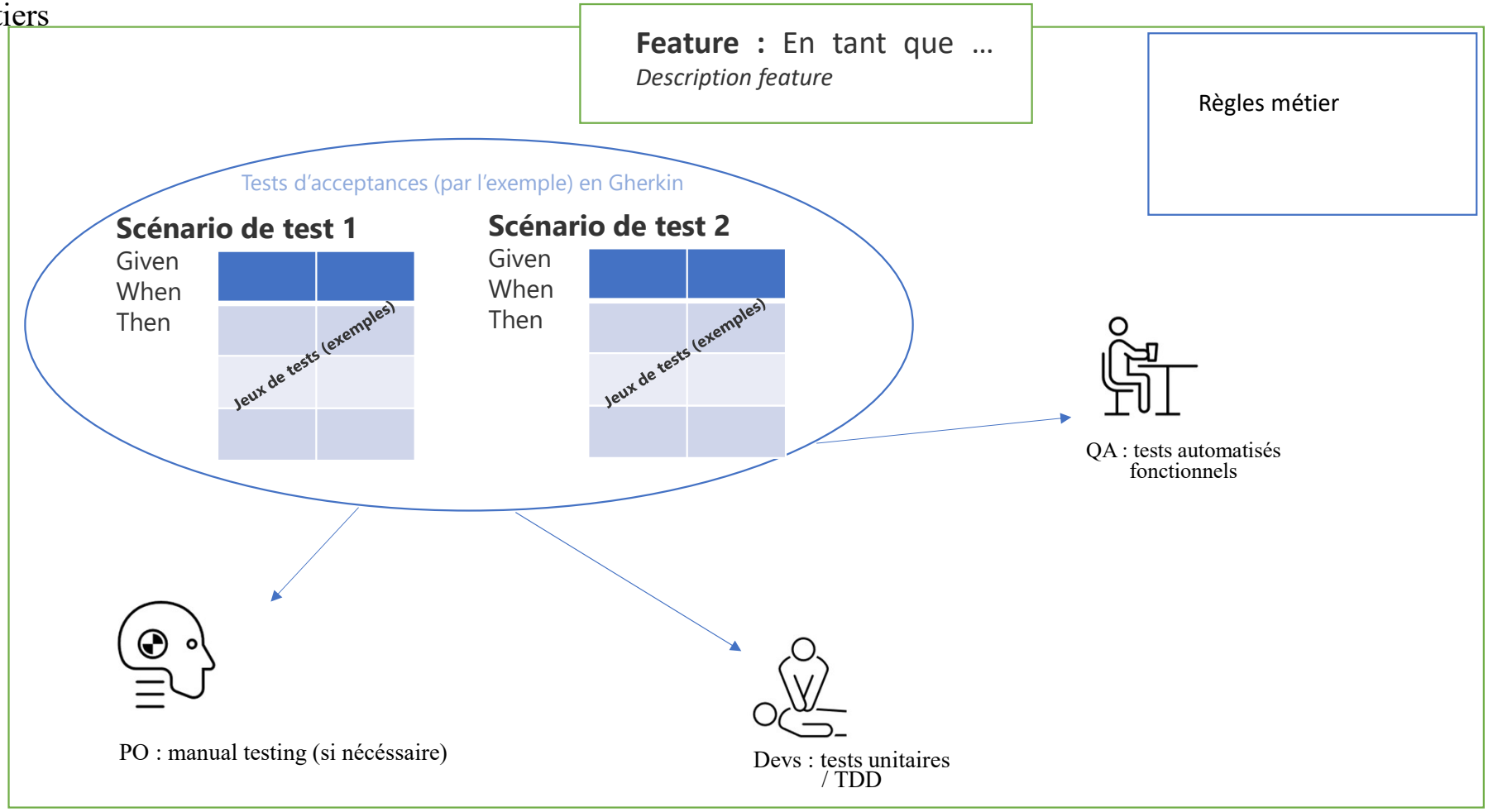
Imagine you are with your three amigos, what how do you explain to them what we have to test

-Il faut connaitre le contexte
-Visibilité des tickets Jira par tous pour avoir une vision globale
-Participation de tout le monde (Devs / PO / QA)  sur les critères d'acceptance
-Bonne découpe des tickets
-S'assurer que chacun sache ce qu'il doit tester
-Identifier le résultat attendu

➔Montrer des exemples de bugs/soucis à tous les participants pour faire comprendre ce qui doit être testé (afin d'éviter les régressions / avoir un code sécurisé / voir les erreurs de conceptions)
➔Expliquer le vision QA mise en place chez Believe  pour tester une feature : Le BDD (behavior-driven development) afin d'encourager la **collaboration entre devs / po / qa**
➔Utiliser **un langage (une syntaxe) commun et compréhensible par tous pour réaliser les tests** : Gherkin
➔**Rédaction collective des scénarios**

# Test strategy au niveau feature (template)

Pour une feature : un scénario commun **en BDD** (et un jeu de tests) en point d'entrée qui servira pour les tests de chaque tiers



# Test strategy pour le site https://www.believemusic.com/

Voir le Github avec les fichiers .features
Chaque fichier .feature sera un scénario de test pour le site web

# Framework Choice for Testing UI
# (pros)

Le choix va se porter sur l'association **Cucumber / Selenium / Javascript**

Cucumber car :
-il embarque la syntaxe BDD de Gherkin.
-il est compatible avec beaucoup de langages
-compatibilité Jira
-Reutilisation du Gherkin en code de test fonctionnel / Assertions

Selenium car :
-c'est la référence pour le test de browser
-gère le multi browser / simule avec un langage simplifié les actions comme un vrai utilisateur
-compatible avec Cucumber
-plusieurs outils pour gérer au choix l'automatisation / ou le contrôle manuel
-sous forme d'ide ou de plugin et compatible window / mac / linux
-différents langages supportés

Javascript car :
-la plupart des développeurs chez Believe ont des connaissances js ou php
-Le javascript servira dans d'autres situations que le test UI et ce la évite de multiplier les langages de testing
-Pas besoin de logiciel spécifique. Un simple éditeur de texte fonctionne

Les trois outils présentés sont réputés et ont faits leurs preuves

# QA report Template

| Feature name | DEV | RC |
|---|---|---|
| check_language | passed | error |
| check_titles | passed | passed |
| check_response_time | passed | passed |
| check_link_effective | passed | passed |
| check_image | passed | undefined |
| check_form | Out of scope | Out of scope |
| check_mailto | passed | passed |

**Details errors :**

-check_language error :
When ..
Then …

Click here to have details

-check_image undefined :
When ..
Then …