# Table des matières

# Liste des tableaux

# Liste des figures

# 1. Introduction

GWS – Aux Services du Vin SA (GWS) est une entreprise spécialisée dans l'organisation d'évènements de dégustations professionnelles de vins. Monsieur X, co-gérant de la société et mandant du projet VaudVin, souhaiterait étendre leurs activités en proposant une plateforme mobile d'évaluation de vins suisses, afin de pouvoir mettre en avant les vins locaux et attirer une clientèle plus large en visant un public lambda, potentiels consommateurs de vins.

L'objectif de l'application est d'encourager la consommation de vins locaux, en aidant les utilisateurs à choisir un vin en fonction de leurs goûts. Après avoir évalué un certain nombre de vins, l'utilisateur aura la possibilité de retrouver via l'application son historique de notes triées par vins mais également les restaurants disponibles à proximité de son emplacement. De cette manière, il pourra consulter leur carte des vins et savoir quel millésime peut lui plaire avant même de se rendre au restaurant. Chaque vin sera accompagné d'un coefficient de correspondance (cf : chapitre 9) calculé à partir de ses évaluations précédentes afin de l'aider dans son choix.

Moi-même étant un consommateur occasionnel de vin, je ne me suis jamais réellement penché sur la problématique évidente de la tendance des acheteurs à vouloir consommer davantage de vins importés aux dépens des produits locaux. Par la suite, si le projet fonctionne et le succès escompté est au rendez-vous, la portée de l'application pourrait s'étendre à l'échelle nationale.

Ce projet a donc éveillé ma curiosité pour en apprendre plus sur la manière de promouvoir des produits locaux, tout en développant une culture dans un domaine d'étude qui pourrait m'être utile tout au long de ma vie, à savoir : la culture du vin, aussi connue sous le nom d'œnologie.

J'ai alors pris contact avec le mandant afin de discuter des modalités du projet ainsi que de ses attentes. Nous nous sommes donc rencontrés et j'ai été très inspiré, intéressé et enthousiaste vis-à-vis de son idée de projet. Nous avons conclu que le développement de l'application se terminerait le 31 août 2019 dans le cadre de mes études et qu'une éventuelle maintenance et mise à jour régulière de l'application seraient envisageables en contrepartie d'une négociable future rémunération.

## 2. Consommation des vins en Suisse

Les Suisses sont de grands consommateurs de vin avec une consommation annuelle d'environ 29.40 litres par personne (Doris Boehlen, OFAG 2018), soit un total de 249 millions de litres pour l'année 2017, vins suisses et étrangers confondus.

Cependant, la consommation de vins a subi une baisse sur les dernières années. Si l'on compare plus spécifiquement les années 2016 et 2017, la consommation globale a reculé de près de quatre millions de litres, soit 1,6% de moins que l'année précédente, selon l'Office fédéral de l'agriculture (OFAG). Cette diminution se traduit par le fait que les vendanges ont souffert de conditions météorologiques extrêmes, ce qui a valu des récoltes et une production de vin inférieure à la demande. On anticipe malheureusement une baisse régulière pour les années à venir due à la forte demande en hausse et une diminution de la production (LeNouvelliste 2018).

Selon l'Organisation Internationale de la Vigne et du Vin (OIV), la Suisse se place à la 19e place des plus gros pays consommateurs de vins derrière la Belgique, mais devant l'Autriche. (Thomasvino 2018)

Tableau 1 : Consommation des vins blancs en Suisse

| | 2015 | 2016 | 2017 | 2018 |
|---|---|---|---|---|
| | hl | hl | hl | hl |
| Vin blanc suisse | | | | |
| Stocks début année* | 706'071 | 613'070 | 714'730 | 688'657 |
| Récolte sans jus de raisin | 396'006 | 526'415 | 402'070 | 540'739 |
| Disponibilités | 1'102'077 | 1'139'485 | 1'116'800 | 1'229'396 |
| Stocks fin année* | 613'070 | 714'730 | 688'657 | 791'493 |
| **Consommation** | **489'007** | **424'755** | **428'143** | **437'903** |
| Vin blanc étranger | | | | |
| Stocks début année** | 108'881 | 120'570 | 116'935 | 118'359 |
| Importations | 401'815 | 394'051 | 400'372 | 393'886 |
| Disponibilités | 510'696 | 514'621 | 517'307 | 512'245 |
| Stocks fin année** | 120'570 | 116'935 | 118'359 | 125'473 |
| **Consommation** | **390'126** | **397'686** | **398'948** | **386'772** |
| Consommation totale de vin blanc | 879'133 | 822'441 | 827'091 | 824'675 |

(Swisswine, rapport viticole 2018, p. 26)

Nous constatons ci-dessus une consommation de vins blancs plus ou moins régulière d'année en année, avec une légère tendance des consommateurs à préférer le vin suisse aux importations étrangères. Cela est très certainement dû à une bonne récolte annuelle locale, ainsi que le maintien d'un large stock disponible en fin d'année 2018.

Tableau 2 : Consommation des vins rouges en Suisse

| | 2015 | 2016 | 2017 | 2018 |
|---|---|---|---|---|
| | hl | hl | hl | hl |
| **Vin rouge suisse** | | | | |
| Stocks début année* | 835'861 | 792'732 | 877'067 | 826'528 |
| Récolte sans jus de raisin | 454'443 | 550'324 | 389'720 | 570'795 |
| Disponibilités | 1'290'304 | 1'343'056 | 1'266'787 | 1'397'323 |
| Stocks fin année* | 792'732 | 877'067 | 826'528 | 942'030 |
| **Consommation** | **497'572** | **465'989** | **440'259** | **455'293** |
| **Vin rouge étranger** | | | | |
| Stocks début année** | 433'710 | 438'697 | 424'075 | 434'760 |
| Importations | 1'262'148 | 1'231'817 | 1'237'092 | 1'159'138 |
| Disponibilités | 1'695'858 | 1'670'514 | 1'661'167 | 1'593'898 |
| Stocks fin année** | 438'697 | 424'075 | 434'760 | 435'061 |
| **Consommation** | **1'257'161** | **1'246'439** | **1'226'407** | **1'158'837** |
| **Consommation totale de vin rouge** | **1'754'733** | **1'712'428** | **1'666'666** | **1'614'130** |

(Swisswine, rapport viticole 2018, p. 26)

Le vin rouge quant à lui subit une baisse constante en termes de consommation au fil des années. Il est impressionnant de constater à quel point les suisses préfèrent consommer le vin rouge étranger aux dépens du vin local. Plus de 70% de la consommation totale concerne le vin rouge importé, alors que les stocks de vins suisses quant à eux, subissent une très forte augmentation à la fin de l'année 2018.

Ceci laisse affirmer que les vins locaux ont du mal à se vendre aux consommateurs et de ce fait, les stocks ne s'écoulent pas, ce qui explique le pic record en fin d'année.

Tableau 3 : Consommation totale des vins en Suisse

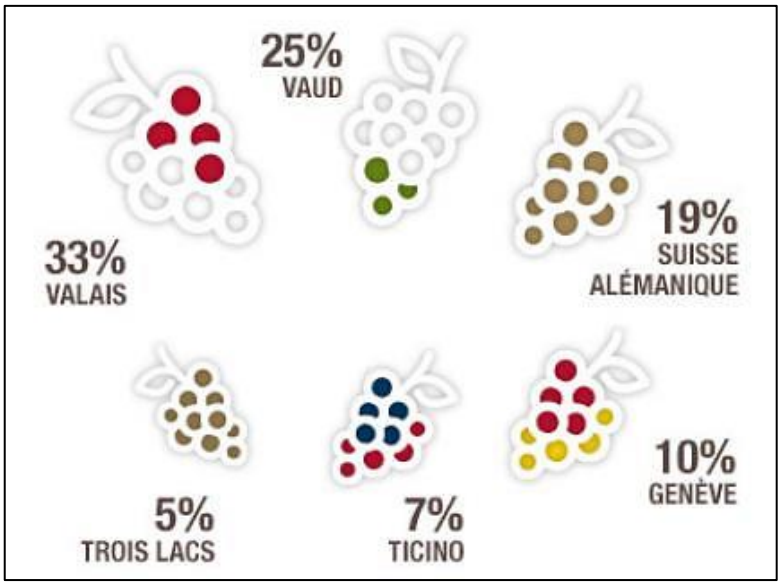
| | 2015 | 2016 | 2017 | 2018 |
|---|---|---|---|---|
| | hl | hl | hl | hl |
| **Consommation totale** | **2'633'866** | **2'534'869** | **2'493'757** | **2'438'805** |
| dont vin suisse | 986'579 | 890'744 | 868'402 | 893'196 |
| dont vin étranger | 1'647'287 | 1'644'125 | 1'625'355 | 1'545'609 |
| Exportations vin blanc*** | 5'095 | 3'916 | 4'571 | 3'527 |
| Exportations vin rouge*** | 6'913 | 7'077 | 8'187 | 7'315 |
| **Consommation totale en Suisse** | **2'621'858** | **2'523'876** | **2'480'999** | **2'427'963** |

(Swisswine, rapport viticole 2018, p. 26)

Tous vins confondus, nous pouvons constater une constante diminution de la consommation globale due notamment à une chute des importations de vins rouges en 2018 (cf : Tableau 2 ci-avant). Ce dernier étant, en Suisse, beaucoup plus consommé que le vin blanc (statistiques basées sur une population âgée à partir de 15 ans, selon les données de l'ONU), le poids d'une baisse de consommation des vins rouges affecte visiblement grandement le total de la consommation en Suisse.

## 2.1 Selon les cantons

Figure 1 : Consommation globale en pourcentage par canton




(Swisswine, chiffres clés)

Nous constatons sur la figure ci-dessus que plus de 50% de la consommation de vin en Suisse est atteinte uniquement par les cantons du Valais et Vaud. Les genevois quant à eux, ne participent qu'à 10% de la consommation totale des vins en Suisse. A l'inverse, les dix-sept cantons suisse allemands sont tous regroupés en un seul et même

pourcentage certes non négligeable de 19%, mais toutefois insignifiant si l'on devait séparer les chiffres de chacun des cantons suisse allemand.

Nous pouvons donc conclure que la grande majorité des consommateurs de vin du pays sont les cantons du Valais, Vaud et Genève, avec un total cumulé de plus de 60% de la consommation globale, suivi de loin par les Suisses allemands (19%) puis des tessinois qui ne constituent que 7% du total et enfin les cantons de Neuchâtel, Fribourg et Bienne, plus connus sous le nom de : "Pays des Trois-Lacs" (Wikipédia 2019), qui eux ne participent qu'à hauteur de 5% de la consommation globale des vins en Suisse

En termes de chiffres, nous pouvons comparer les acteurs principaux du marché suisse de la consommation des vins, à l'aide des tableaux ci-dessous :

Tableau 4 : Consommation totale des vins valaisans

| Vins valaisans | 2015 | 2016 | 2017 | 2018 |
|---|---|---|---|---|
| | hl | hl | hl | hl |
| Vin blanc | | | | |
| Stocks début année* | 228'043 | 202'983 | 219'847 | 191'797 |
| Récolte sans jus de raisin | 129'473 | 169'149 | 112'445 | 171'922 |
| Disponibilités | 357'516 | 372'132 | 332'292 | 363'719 |
| Stocks fin année* | 202'983 | 219'847 | 191'797 | 228'971 |
| **Consommation** | **154'533** | **152'285** | **140'495** | **134'748** |
| Vin rouge | | | | |
| Stocks début année* | 324'954 | 311'745 | 357'460 | 327'338 |
| Récolte sans jus de raisin | 198'363 | 244'558 | 150'358 | 247'972 |
| Disponibilités | 523'317 | 556'303 | 507'818 | 575'310 |
| Stocks fin année* | 311'745 | 357'460 | 327'338 | 385'402 |
| **Consommation** | **211'572** | **198'843** | **180'480** | **189'908** |
| **Consommation totale** | **366'105** | **351'128** | **320'975** | **324'656** |

Tableau 5 : Consommation totale des vins vaudois

| Vins vaudois | 2015 | 2016 | 2017 | 2018 |
|---|---|---|---|---|
| | hl | hl | hl | hl |
| Vin blanc | | | | |
| Stocks début année* | 299'265 | 252'486 | 292'281 | 317'300 |
| Récolte sans jus de raisin | 154'092 | 218'289 | 195'662 | 213'666 |
| Disponibilités | 453'357 | 470'775 | 487'943 | 530'966 |
| Stocks fin année* | 252'486 | 292'281 | 317'300 | 352'514 |
| **Consommation** | **200'871** | **178'494** | **170'643** | **178'452** |
| Vin rouge | | | | |
| Stocks début année* | 135'519 | 117'537 | 135'363 | 132'120 |
| Récolte sans jus de raisin | 63'934 | 85'490 | 71'727 | 84'749 |
| Disponibilités | 199'453 | 203'027 | 207'090 | 216'869 |
| Stocks fin année* | 117'537 | 135'363 | 132'120 | 148'578 |
| **Consommation** | **81'916** | **67'664** | **74'970** | **68'291** |
| **Consommation totale** | **282'787** | **246'158** | **245'613** | **246'743** |

(Swisswine, rapport viticole 2018, p. 30)

Tableau 6 : Consommation totale des vins genevois

| Vins genevois | 2015 | 2016 | 2017 | 2018 |
|---|---|---|---|---|
| | hl | hl | hl | hl |
| Vin blanc | | | | |
| Stocks début année* | 33'671 | 29'412 | 34'128 | 34'968 |
| Récolte sans jus de raisin | 35'202 | 53'117 | 31'941 | 47'471 |
| Disponibilités | 68'873 | 82'529 | 66'069 | 82'439 |
| Stocks fin année* | 29'412 | 34'128 | 34'968 | 33'918 |
| **Consommation** | **39'461** | **48'401** | **31'101** | **48'521** |
| Vin rouge | | | | |
| Stocks début année* | 56'525 | 50'550 | 60'762 | 55'463 |
| Récolte sans jus de raisin | 42'231 | 61'159 | 36'920 | 53'366 |
| Disponibilités | 98'756 | 111'709 | 97'682 | 108'829 |
| Stocks fin année* | 50'550 | 60'762 | 55'463 | 59'174 |
| **Consommation** | **48'206** | **50'947** | **42'219** | **49'655** |
| **Consommation totale** | **87'667** | **99'348** | **73'320** | **98'176** |

(Swisswine, rapport viticole 2018, p. 30)

Il est intéressant de constater une forte croissance de la consommation totale chez les genevois en 2018 contrairement aux vaudois et valaisans qui, eux, stagnent de leur côté. De plus, les vaudois préfèrent nettement le vin blanc, alors que leurs voisins tendent vers le vin rouge. Quant aux stocks de fin d'année, tous ont subi une augmentation significative sauf chez les genevois, qui ont visiblement l'air de bien vendre leurs vins.

# 3. Problématique à résoudre avec VaudVin

Au fil des années, les Suisses consomment vraisemblablement de moins en moins de quantité de vin. Jusqu'en 2016, pour ne parler que des vins suisses, pas moins de 100 millions de litres s'écoulaient habituellement chaque année chez les helvètes. En 2017, suivant les statistiques de l'OFAG, ce chiffre a chuté pour s'établir à 87 millions (43 millions de litres pour les vins blancs et 44 millions pour les vins rouges), soit une diminution conséquente de 13.1%. En ce qui concerne les vins étrangers, l'importation a elle aussi subi une faible baisse de 1.1%, soit un peu moins d'un million de litres, pour finalement s'établir à 163 millions de litres consommés en 2017 (OFAG 2018).

Nous pouvons donc en déduire que le marché des vins locaux est en recul par rapport à celui des produits importés. C'est pourquoi le mandant du projet VaudVin, a imaginé une application mobile qui permettrait de promouvoir dans un premier temps les vins vaudois (et par la suite envisager à plus grande échelle les vins nationaux) dans le but d'encourager les utilisateurs à pouvoir consommer davantage les vins locaux aux dépens des vins étrangers, en leur permettant de noter les cépages au fur et à mesure de leur(s) expérience(s), qu'ils se trouvent au supermarché, dans un restaurant ou encore dans les caves ouvertes, chez un vigneron.

En effet, VaudVin vient avant tout répondre à un besoin économique par le fait de pousser le consommateur à s'orienter vers des vins locaux en lui permettant de sélectionner un vin en fonction de ses goûts (cf : coefficient de correspondance). Après avoir évalué un certain nombre de vins, l'utilisateur aura la possibilité de voir à travers l'application, les vins vaudois disponibles au restaurant dans lequel il se trouve. Chaque vin sera accompagné d'un coefficient de correspondance calculé à partir de ses évaluations précédentes, afin de l'aider dans son choix.

De ce fait et pour conclure, l'application VaudVin apporte une solution sur mesure afin de pallier la forte baisse de consommation des vins suisses au fil des années, tout en proposant une interface mobile disponible à portée de main et utile dans plusieurs contextes différents, selon où se trouve le client au moment de l'utilisation.

# 4. Les concurrents directs

Il existe une multitude d'applications considérées comme des concurrents directs au projet VaudVin. C'est pourquoi, j'ai directement demandé au mandant de me lister, selon lui, les principaux qu'il considère comme étant des concurrents de taille pouvant affecter les futurs utilisateurs dans leur choix et leur orientation. Les concurrents directs pour ce projet sont les suivants :

## 4.1 Vivino

Un des concurrents principaux pour ce projet n'est autre que l'application mobile communautaire Vivino, permettant à tout amateur de vin de choisir sa bouteille, de partager ses goûts, de répertorier ses notes de dégustation et de gérer sa cave (liste de bouteilles) en ligne et/ou commander/acheter des bouteilles (François Potevin, 2017). L'application donne toutes les informations sur le cépage, le pedigree du raisin mais aussi les notes des autres utilisateurs, les commentaires des connaisseurs et le prix. Ceci permet à l'utilisateur de connaître absolument tout sur le vin qui se trouve devant lui.

Avantages : c'est la première application mondiale consacrée au vin regroupant pas moins de 25 millions d'utilisateurs. Ces derniers peuvent évaluer, partager et acheter leurs bouteilles directement aux producteurs, sans passer par la grande distribution. Chaque utilisateur possède un profil de dégustation, qui permet de répertorier ses coups de cœurs et découvrir chaque semaine les meilleures bouteilles, selon nos goûts.

Inconvénients : la cave en ligne incite l'utilisateur à consommer du vin comme s'il était sur un site de vente en ligne, ce qui engage une commercialisation trop intense pour finalement laisser de côté l'aspect "évaluation" aux dépens de l'achat de bouteilles. De plus, la fiabilité du scanner d'étiquettes laisse à désirer. Beaucoup de domaines ont une appellation qui existe en deux, voire trois couleurs de vins (Léa Delage, Lemonde, 2018).

A la différence principale avec mon projet, Vivino ne se focalise pas sur une région particulière et place tous les vins du monde sur le "même niveau", ce qui signifie que la culture viticole de l'utilisateur ne peut se limiter par région.

## 4.2 Wine Ring

Au moins une fois dans notre vie, il nous est arrivé de passer des heures à parcourir le rayon ou la liste des vins, convaincus de trouver la bouteille parfaite basée sur le cépage et le prix. Là où Wine Ring vient se démarquer de ses concurrents, c'est par le fait d'avoir un "sommelier personnel" qui nous propose des suggestions basées sur nos goûts à l'aide de l'intelligence artificielle (IA).

Wine Ring utilise un algorithme avancé qui apprend quel type de vin est susceptible de nous plaire en fonction de nos goûts. C'est la première application à utiliser la technologie de l'IA et du machine-learning pour créer des suggestions inspirées de nos préférences personnelles (Jeniece Pettitt, CNBC, 2016)

A la différence de VaudVin, Wine Ring présente une interface complexe et très orientée sur les préférences de l'utilisateur. Il s'agit là d'une application regroupant des recommandations personnelles et ne permet pas d'orienter l'utilisateur vers des restaurants proposant tel ou tel cépage.

## 4.3 Wine Picker

Très similaire à Wine Ring, mais dépourvu de technologies comme l'intelligence artificielle, l'e-sommelier britannique permet d'aider le public à s'orienter à travers la sélection d'un établissement. En fonction du plat que l'utilisateur choisit, mais aussi de son budget ainsi que de la cotation des vins de la cave, l'application lui recommande un nombre de choix (Sophie Marenne, Agefi, 2018).

Il est également possible d'ajouter une carte des vins de manière automatisée. Il suffit de la charger dans l'application sous forme de document ou de photo. Seul bémol : si la carte a une police d'écriture trop alambiquée, alors il est fort probable qu'elle ne soit pas reconnue par l'application.

L'impact de Wine Picker est très faible : seulement 15'000 utilisateurs actifs, soit un nombre insignifiant à côté de Vivino par exemple.

En comparaison avec VaudVin, Wine Picker se focalise principalement sur les restaurants et non sur un historique de préférences personnelles, ce qui revient à limiter les cas d'utilisation.

## 4.4 Wine Advisor

L'application 100% française permet, grâce à la reconnaissance d'image, de scanner l'étiquette de la bouteille de vin afin d'accéder aux informations clés telles que : les caractéristiques, les commentaires des autres utilisateurs mais également les notes attribuées par la communauté.

Wine Advisor affiche également les accords mets & vins, les avis et le potentiel de garde ainsi que toutes les informations sur le domaine, l'appellation et la région viticole. Elle stocke un historique de toutes les dégustations de vins et les coups de cœur de l'utilisateur. C'est également un outil de partage, c'est-à-dire que l'application donne la possibilité de pouvoir échanger des messages avec la communauté ainsi qu'avec les professionnels du vin (Macave, E.Leclerc, 2014).

Il y a également une fonction « Gestion de Cave » qui permet de gérer facilement une cave à vin virtuelle. L'utilisateur peut ainsi connaître facilement les types de vins dont il dispose et les quantités stockées.

Par rapport à VaudVin, Wine Advisor est très orienté sur les partages et les échanges entre les différents utilisateurs de l'application, ce qui en fait un concurrent direct puisque la fonctionnalité d'historique des notes se rapproche de celle de mon projet. En revanche, pas de possibilité de pouvoir retrouver des restaurants contenant les vins préférés.

## 4.5 Delectable Wine

Prédécesseur des applications viticoles, Delectable Wine est une application mobile américaine développée en 2011, dont le concept principal est de scanner une bouteille de vin afin de reconnaître n'importe quel vin du monde à partir de son étiquette et se procurer toutes les informations nécessaires sur son origine, sa composition, son producteur, son prix, etc. (Soonsoonsoon, 2015).

Les notes et avis des autres utilisateurs permettent de se faire une idée de la qualité du vin et des liens vers des partenaires commerciaux sont proposés par l'application afin de commander directement une bouteille.

La vinothèque virtuelle permet d'enregistrer tous les vins scannés et ainsi, plus besoin de mémoriser quoi que ce soit.

A titre de comparaison avec VaudVin, Delectable Wine ne propose pas une recherche de restaurants à proximité afin de retrouver ses vins préférés. L'application américaine se contente d'enregistrer les préférences du compte utilisateur connecté et de pouvoir retrouver ses vins préférés ou scanner de nouveaux vins.

## 4.6 Hello Vino

L'application mobile Hello Vino n'est autre qu'une plateforme de recommandations de vins dépendant du périmètre de recherche et des filtres sélectionnés. Il y a, entre autres, la possibilité de retrouver un vin en fonction d'une occasion spéciale (accompagnement d'un plat, mariage, soirée estivale, etc.). L'application donne à l'utilisateur une liste de vins recommandés en fonction de ses critères en termes de goûts, mais aussi de prix (Robyn, Macsources, 2013).

L'application fournit des informations très précises sur les accords mets-vins. En revanche, le scanner d'étiquette est une fonctionnalité payante.

Encore une fois, les vins étrangers ne sont pas différenciés des vins suisses et c'est là que VaudVin trouve sa force en comparaison avec Hello Vino. L'application américaine est certes exhaustive en termes de recommandations, mais l'interface se montre trop chargée et pas très intuitive à l'utilisation.

## 4.7 MyOeno (smart Object)

Original et innovant, MyOeno n'est pas seulement une application mobile mais bel et bien un objet intelligent à part entière. Connecté via la technologie Bluetooth au téléphone de son possesseur, minimaliste et efficace, ce petit scanner de vin permet de reconnaître en quelques secondes les informations du contenu se trouvant dans un verre de vin, en plongeant l'extrémité de l'appareil à l'intérieur du liquide, tout simplement.

Plus précisément, trois informations principales ou trois paramètres œnologiques caractéristiques, sont relevés par l'objet connecté :

- Puissance
- Vicacité
- Tanins

MyOeno n'est pas un outil de mesure à proprement dit, mais permet de distinguer des similitudes de styles grâce aux trois marqueurs œnologiques cités ci-avant. Une fois ces trois informations affichées, l'application mobile propose à l'utilisateur de renseigner les informations détaillées du vin (domaine, millésime, pays, région et appellation). L'utilisateur a alors la possibilité de prendre une photo, partager la mesure avec la communauté et assigner une note sur une échelle de cinq étoiles.

L'intérêt principal de ce scanner connecté, n'est autre que de proposer des vins similaires à celui que l'on vient de scanner. Ceci limite grandement les cas d'utilisation, malgré le concept de l'objet physique relié à une application mobile minimaliste. En comparaison avec VaudVin, ce concurrent est encore en plein développement et ne pose donc pas de réelles contraintes ou risques pour l'instant.

# 5. Analyse des risques

L'envergure d'un tel projet informatique m'a fait prendre conscience de l'utilité de devoir établir une liste de risques potentiels pouvant ralentir voire mener à l'échec le développement du projet tout au long de sa progression, mais également lors de l'utilisation de l'application.

Ainsi, j'ai décidé de séparer les risques en trois parties distinctes :

- Les risques liés au développement du projet
- Les risques liés à l'application
- Les risques liés à l'équipe de projet.

## 5.1 Risques liés au développement du projet

### 1. Mauvaise gestion du processeur vis-à-vis du code

Probabilité courante / Impact faible

Solution : faire en sorte de toujours adapter le code pour optimiser la performance du processeur tout en évitant la redondance, prioriser les appels de fonctions, composants réutilisables, utilisation de librairies, etc.

### 2. Problèmes d'incompatibilité des dépendances entre API

Probabilité moyenne / Impact modéré

Solution : lors de l'utilisation de librairies ou API externes, faire en sorte de mettre à jour les dépendances du projet et s'assurer qu'il puisse continuellement run sur plusieurs machines différentes sans problèmes. Dans le cas où l'installation d'une API pose problème avec la version des Frameworks, revenir en arrière et trouver une alternative pour obtenir le même résultat.

### 3. Les technologies des Frameworks mal maîtrisées

Probabilité courante / Impact modéré

Solution : consulter régulièrement la documentation officielle des Frameworks utilisés et dans la bonne version (dans notre contexte Ionic 4 et Laravel PHP 5.5), afin de répondre correctement aux fonctionnalités attendues. Si besoin, mettre à jour les librairies afin d'éviter la dépréciation de certaines fonctions.

### 4. Trop d'effets de bords après une modification

Probabilité moyenne / Impact élevé

Solution : faire en sorte de structurer l'application, afin de comprendre quels composants interagissent entre eux et faire régulièrement des tests de l'application après chaque modification minime.

### 5. Mauvaise documentation du code

Probabilité très courante / Impact modéré

Solution : A chaque appel de fonction à l'intérieur d'une autre fonction, commenter le code de manière à ce qu'une personne qui ne connaît pas l'application ne perde pas de temps à comprendre qu'est-ce qui fait quoi.

Tableau 7 : Risques liés au développement (avant mitigation)

| Probabilité/ Impact | Faible | Moyenne | Courante | Très courante |
|---|---|---|---|---|
| Faible | | | **1** | |
| Modéré | | **2** | **3** | **5** |
| Elevé | | **4** | | |
| Très élevé | | | | |

(Edouard Diep 2019)

Tableau 8 : Risques liés au développement (après mitigation)

| Probabilité/ Impact | Faible | Moyenne | Courante | Très courante |
|---|---|---|---|---|
| Faible | **1, 5** | | | |
| Modéré | **2, 4** | **3** | | |
| Elevé | | | | |
| Très élevé | | | | |

(Edouard Diep 2019)

Comme nous le constatons ci-dessus (cf : Tableau 8), les solutions apportées aux risques liés au développement de l'application, permettent de ramener dans le "vert" la quasi-totalité des risques à anticiper pour cet aspect du projet. En effet, beaucoup de risques à probabilité moyenne et courante deviennent alors faibles, voire inexistants en termes de survenance si l'on applique correctement et régulièrement les solutions avant mitigation.

## 5.2 Risques liés à l'application

### 1. L'API de géolocalisation subit une interruption côté serveur

Probabilité faible / Impact modéré

Solution : Si l'interruption dure moins de 10 secondes : afficher un message demandant à l'utilisateur de patienter quelques instants. Si l'interruption dure plus de 10 secondes : afficher les informations des restaurants sous forme de cartes dans une liste déroulante sur laquelle l'utilisateur peut toucher l'écran pour accéder aux cartes des vins.

### 2. Les serveurs où sont stockées nos données ne sont plus accessibles

Probabilité moyenne / Impact très élevé

Solution : Mettre en place des sauvegardes dans des lieux fiables, qui pourront être utilisées en cas de d'indisponibilité ou d'inaccessibilité des données au niveau des serveurs externes.

### 3. Le site est victime d'attaques visant à interrompre le service

Probabilité moyenne / Impact très élevé

Solution : Nous éviterons ce type de problème en disposant d'une copie de l'application sur un serveur de secours, qui peut être activé à tout moment afin de pallier une potentielle interruption de service.

### 4. Vol de données

Probabilité moyenne / Impact très élevé

Solution : Pour éviter ce risque à impact élevé, il faut identifier et classer les données sensibles, afin de faire en sorte que ces dernières soient stockées, selon les besoins, de manière locale grâce au stockage ionic ou global sur la base de données. Nous ferons également en sorte d'avoir des copies des données sur des disques partagés ou un cloud.

Tableau 9 : Risques liés à l'application (avant mitigation)

| Probabilité/ Impact | Faible | Moyenne | Courante | Très courante |
|---|---|---|---|---|
| Faible | | | | |
| Modéré | **1** | | | |
| Elevé | | | | |
| Très élevé | | **2, 3, 4** | | |

(Edouard Diep 2019)

Tableau 10 : Risques liés à l'application (après mitigation)

| Probabilité/ Impact | Faible | Moyenne | Courante | Très courante |
|---|---|---|---|---|
| Faible | **1** | **4** | | |
| Modéré | | **2, 3** | | |
| Elevé | | | | |
| Très élevé | | | | |

(Edouard Diep 2019)

En ce qui concerne l'aspect déploiement de l'application, les risques ont été mitigés, mais impactent toujours modérément le projet en cas de survenance. Ceci est dû principalement au fait que ces derniers ne dépendent pas des développeurs de l'application, mais sont liés à des facteurs externes dont on ne maîtrise pas forcément la probabilité de survenance. C'est pourquoi il s'agit d'être particulièrement attentif et vigilant au cas où un de ces risques venait à survenir.

## 5.3 Risques liés à l'équipe de projet

### 1. Un des membres ne peut pas travailler (court terme)

Probabilité moyenne / Impact faible

Solution : Dans le cas d'un empêchement à hauteur de quelques heures (travail ou autres), le membre concerné doit prévenir le reste de l'équipe de projet et rattraper le retard dès son retour, de telle sorte que l'itération puisse être entièrement accomplie dans le délai imparti.

### 2. Confusion entre les attentes du mandant et la compréhension de celles-ci par l'équipe

Probabilité moyenne / Impact modéré

Solution : Faire en sorte d'obtenir les informations nécessaires de la part du mandant et ne pas hésiter à répéter les détails pour la bonne exécution de l'itération suivante afin d'éviter les potentiels conflits après coup.

### 3. Retard dû à un problème de communication globale dans l'équipe

Probabilité moyenne / Impact élevé

Solution : Planifier une mise au point hebdomadaire avec le mandant, afin de mettre à jour les nouvelles données/informations pour pouvoir continuer l'itération sans surprises.

### 4. Un des membres ne peut pas travailler (moyen terme)

Probabilité moyenne / Impact élevé

Solution : Dans le cas de vacances ou incapacité de travail pour cause maladie, le membre concerné doit absolument prévenir, si possible à l'avance, les autres membres, afin qu'ils puissent réorganiser la charge de travail de telle sorte que cette dernière puisse être entièrement accomplie dans le délai imparti de l'itération en cours.

### 5. Indisponibilité prolongée du mandant

Probabilité moyenne / Impact très élevé

Solution : Demander suffisamment de précisions au mandant pour ne pas avoir à trop souvent le solliciter et passer aux tâches suivantes pour ne pas freiner le processus de développement.

Tableau 11 : Risques liés à l'équipe de projet (avant mitigation)

| Probabilité/ Impact | Faible | Moyenne | Courante | Très courante |
|---|---|---|---|---|
| Faible | | **1** | | |
| Modéré | | **2** | | |
| Elevé | | **3, 4** | | |
| Très élevé | | **5** | | |

(Edouard Diep 2019)

Tableau 12 : Risques liés à l'équipe de projet (après mitigation)

| Probabilité/ Impact | Faible | Moyenne | Courante | Très courante |
|---|---|---|---|---|
| Faible | | **1** | | |
| Modéré | **2** | **3, 5** | | |
| Elevé | **3** | | | |
| Très élevé | | | | |

(Edouard Diep 2019)

Enfin, en ce qui concerne les risques liés à l'équipe de projet, nous avons pu mitiger presque l'entièreté, en assurant une faible probabilité de survenance ou un faible taux d'impact sur le projet, tout en garantissant une meilleure communication globale afin d'assurer une certaine autonomie pour chacun des membres au sein de l'équipe. Toutefois, si un retard sur le projet venait à se produire à cause de l'équipe, l'impact serait tout-de-même modéré et par conséquent les risques 3 et 5 requièrent une attention toute particulière afin de mener à bien le projet.

Pour conclure, j'ai établi cette analyse de risques dans le but d'anticiper d'éventuelles menaces qui pourraient survenir durant le projet et engendrer de grandes conséquences comme une perte d'argent ou des retards sur les délais voire complétement mener le projet à l'échec. En effet, il s'agit d'être constamment vigilant et de faire en sorte d'avoir une solution efficace afin de mitiger un maximum de risques pour éviter les mauvaises surprises lors du développement mais également à l'avenir, lorsque l'application sera déployée.

# 6. Analyse des Frameworks Front End

## 6.1 Le Framework

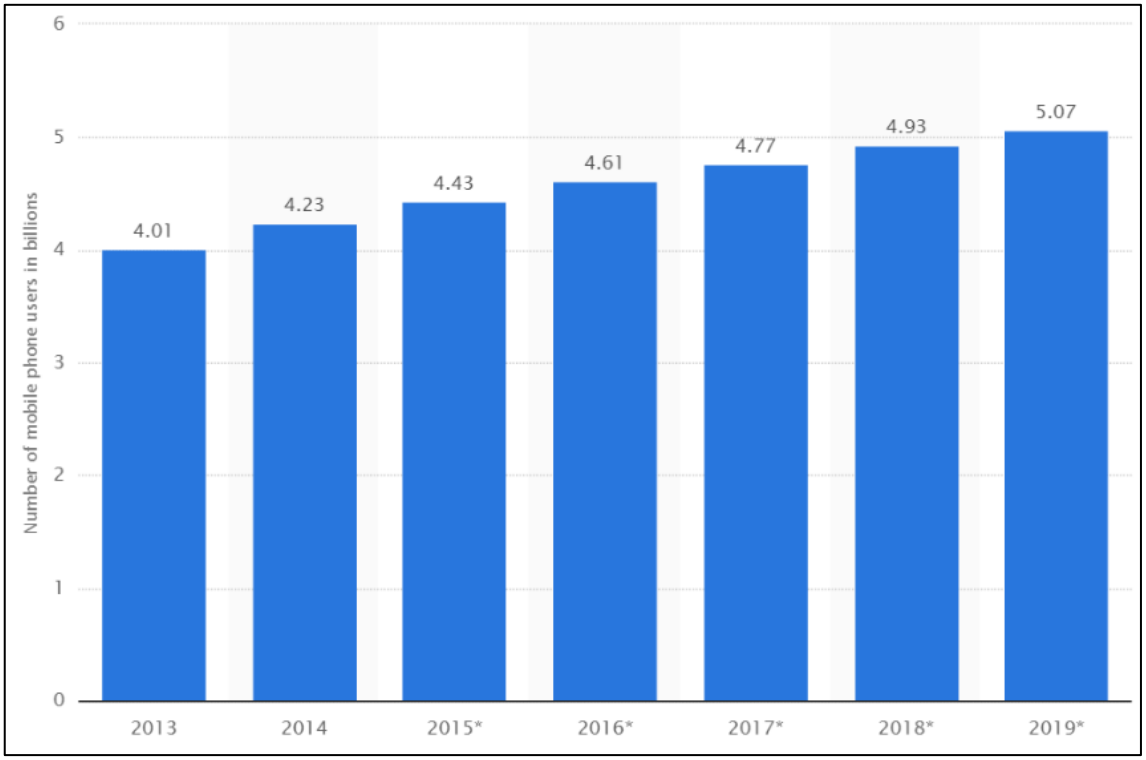
En ce qui concerne la partie principale de ce projet, à savoir le développement de l'application, la première grande décision a été de sélectionner le Framework à utiliser pour créer, coder, développer ladite application. Par définition, un Framework désigne un ensemble cohérent de composants logiciels, qui sert à créer les fondations ainsi que les grandes lignes de tout ou d'une partie d'un logiciel informatique (Wikipédia 2019).

## 6.2 De la Responsive web application à l'application mobile

Au fil des années, la forte croissance du marché des smartphones, a parallèlement fait augmenter la demande de développement d'applications mobiles ou de web applications adaptées mobiles, plus communément appelées : "Responsive web applications". Par définition, le responsive web design est une approche visuelle dont la conception vise, grâce à différents principes et techniques, à offrir une consultation confortable sur des écrans de tailles très différentes (Wikipédia 2019). Un site internet dit "responsive" a donc un design particulier, basé sur des paramètres spécifiques du langage de style CSS, appelés "*media queries*". Les media queries constituent un module CSS permettant d'adapter le contenu d'une page web aux caractéristiques de l'appareil de l'utilisateur, afin qu'il soit consultable aussi bien sur des écrans d'ordinateurs de bureau, que sur des écrans de smartphones (Wikipédia 2019). Ces paramètres de style ne sont pas sans failles : ils requièrent davantage de travail et d'attention de la part du développeur et le site devient relativement lent en termes de performances du au fait que le code est parfois redondant, afin de gérer les différents affichages.

D'années en années, la quantité d'utilisateurs de téléphones mobiles augmentent drastiquement dans le monde. Voici ci-dessous un graphique démontrant des chiffres en billions :

Figure 2 : Nombre d'utilisateurs mobiles entre 2013 et 2019 (en billions)




(Blog Thinkmobiles 2019, p. 1)

Mais alors la question suivante vient à se poser : Comment pallier ce temps conséquent et ce travail supplémentaire consacré à une simple adaptation d'affichage sur différents appareils pour un seul et même site web ? Réponse : en s'orientant vers les applications mobiles. Ces dernières sont des produits applicatifs à part entière, pouvant être installés sur un ou plusieurs appareils (smartphones ou tablettes). Les applications sont développées pour fonctionner avec différents systèmes d'exploitation, dont les plus populaires sont iOS, Android et Windows Phone. Les applications mobiles fonctionnent différemment des responsive web applications. Elles ne s'exécutent pas forcément sur tous les appareils et nécessitent d'être installées séparément d'un appareil à l'autre. En revanche, elles fournissent à l'utilisateur une expérience unique et sur mesure (Thinkmobiles, 2019).

Nous pouvons donc comparer en détail certains critères, auxquels répondent les responsive web applications et les applications mobiles sur le tableau suivant (cf : Tableau 13 page suivante) :

Tableau 13 : Comparatif – Responsive web app VS Mobile app

| | **Responsive web app** | **Mobile app** |
|---|---|---|
| **Compatibilité** | La version mobile du site est affichée de la même manière, dans tous les navigateurs. | Nécessite le développement de plusieurs applications pour diverses plateformes, dépendant du système d'exploitation. |
| **Audience** | Tous les appareils qui ont accès à internet, peuvent consulter l'application. | Seuls les smartphones et tablettes peuvent consulter l'application. |
| **Coût d'entrée** | Paiement pour le domaine et l'hébergement. | Paiement des licences pour les développeurs dans les plateformes d'applications (app stores). |
| **Facilité d'utilisation** | Ne nécessite pas de téléchargement ou d'installation. | Nécessite un téléchargement et une installation locale. |
| **Fonctionne hors-ligne** | Pas sur tous les appareils. | Oui. |
| **Maintenance, mises à jour et bug fixing** | Facile à mettre à jour, maintenir et fixer les bugs. | Difficile à mettre à jour et gérer l'application après qu'elle ait été téléchargée. Les bugs seront fixés dans la prochaine version. |
| **Utilisation régulière** | Moyenne. | Très bon pour une utilisation régulière. |
| **Personnalisation** | Moyenne. Le site mobile est plus orienté sur le service, plutôt que sur l'aspect visuel. | Très optimisé pour la personnalisation. L'application vise une expérience unique pour l'utilisateur. |

(Blog Thinkmobiles 2019)

En conclusion, les deux approches présentent des avantages comme des inconvénients. L'application mobile est bien mieux adaptée à une utilisation fréquente et permet à l'utilisateur d'avoir une interface visuelle unique et très performante en termes de navigation contrairement à la responsive web app, qui elle, se contente d'afficher un site internet adapté sur différents écrans. En ce qui concerne le projet VaudVin, le mandant et moi-même avons conclu que la meilleure solution était l'application mobile, afin de satisfaire ses besoins les plus pointilleux en termes d'affichage visuel. De plus, le mandant ayant déjà un site internet pour organiser les concours de vin annuels, il désirait étendre ses services en développant un projet accessible sur smartphone et ne voyait pas d'intérêt à créer un second site web, ce qui l'a poussé à choisir l'application mobile comme solution idéale.

## 6.3 Frameworks natifs vs hybrides

Il existe deux types de Frameworks reconnus pour le développement d'applications mobiles :

- Les Frameworks dits "natifs"
- Les Frameworks dits "hybrides"

Par définition, les applications natives sont écrites dans des langages de programmation spécifiques aux plateformes pour lesquelles elles ont été développées. Les langages utilisés les plus courants sont : Objective-C ou Swift pour les systèmes iOS, Java ou Kotlin pour Android et Flutter qui s'adapte aux deux systèmes concurrents cités ci-avant. Les applications natives fournissent généralement de meilleures performances que les applications hybrides et permettent un accès direct à toutes les fonctionnalités natives que peuvent offrir un smartphone (GPS, appareil photo, réalité virtuelle, notification, SMS, etc.). (Apparence.io 2019).

Les applications hybrides quant à elles, sont des programmes utilisant le navigateur web intégré du support sur lequel elles sont installées (Smartphone ou tablette) et les technologies Web liées (HTML, CSS et Javascript) pour fonctionner sur différents systèmes d'exploitation avec un seul et même code (iOS, Android, Windows Phone, etc.). Une telle application utilise également les fonctionnalités natives des Smartphones et peut, tout comme les applications natives, être distribuée sur les plateformes d'applications telles que l'AppStore ou Google Play (Wikipédia 2019).

Pourquoi m'être orienté vers un Framework dit "hybride" aux dépens des Frameworks de développements natifs ou standards ? À titre de comparaison, voici un tableau :

Tableau 14 : Comparatif – Applications hybrides VS Applications natives

| Applications hybrides | Applications Natives |
|---|---|
| Développées en HTML, CSS et Javascript. | Développées dans des langages spécifiques aux plateformes : Objective-C ou Swift pour iOS, Java pour Android, etc. |
| Un seul code, exécutable partout. | Un code différent pour chaque plateforme. |
| Performance moyenne comparé aux applications natives. | La plus rapide et performante expérience pour les utilisateurs. |
| Peu coûteuse en termes de temps et d'argent. | Nécessite plus d'investissement en termes de temps, talent et ressources. |
| Cycle de développement rapide. | Coûts supérieurs et plus long à développer. |

(Mike Butusov, 2019)

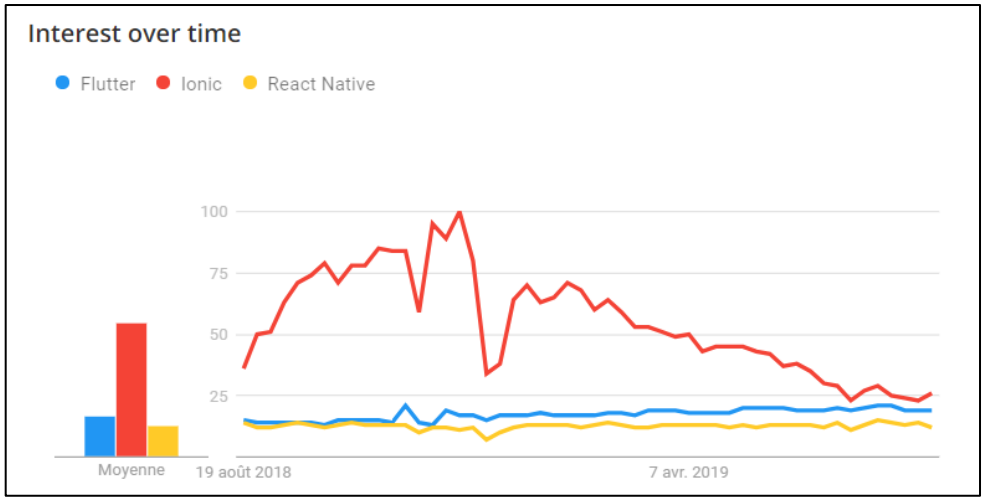
## 6.4 Framework de développement hybride

Pourquoi s'orienter vers un Framework de développement hybride plutôt qu'un développement natif ? Dans le contexte d'un projet comme VaudVin, le temps et l'argent ont été des facteurs décisifs dès le lancement du projet. En effet, le mandant désirait obtenir un prototype fonctionnel, compatible multi-plateformes et sur une durée de développement de deux mois, ce qui est relativement court pour un projet d'une telle envergure. De plus, le fait de pouvoir écrire un seul code pour déployer le même projet

sur iOS et Android tout en utilisant les technologies web (HTML, CSS et Javascript) désormais maîtrisées, m'a convaincu de l'efficacité d'une telle méthodologie de développement. En fin de compte, cela m'a permis de gagner un temps considérable sur la partie construction, mais aussi pour le mandant qui, de son côté, n'a pas eu besoin d'embaucher une équipe de développement par plateforme, ce qui représente un coût financier moindre.

Par conséquent, mon choix s'est très vite départagé entre Ionic, React Native et Flutter : les trois Frameworks hybrides les plus populaires en 2019 (Rushi Trivedi 2019). À titre de popularité, les tendances d'intérêt pour chacun des Frameworks peuvent être constatées sur la figure ci-dessous :

Figure 3 : Courbe des tendances d'intérêt pour les Frameworks Front end




(Stackshare 2019)

Malgré une forte régression de la tendance d'intérêt entre 2018 et 2019, Ionic reste très faiblement en tête des autres Frameworks hybrides, bien que chacun s'équivaille aujourd'hui en termes de popularité et de communauté. Dans le sous-chapitre suivant, je détaillerai chaque Framework avec leurs avantages et inconvénients.

### 6.4.1 Ionic

La première version d'Ionic a été lancée en 2013 par Drifty.co. C'est un Framework standard pour le développement d'applications mobiles hybrides. Il permet aux développeurs web de construire des applications pour la majorité des plateformes à l'aide d'un seul et même code. Ionic utilise les technologies web : HTML, CSS et Javascript ainsi que les plateformes PhoneGap et Cordova afin de fournir à l'utilisateur une expérience proche de celles délivrées par les applications natives (Spec India 2018). Les applications comme MarketWatch, Pacifica ou encore Sworkit sont des applications populaires créées à l'aide du Framework Ionic (Rushi Trivedi 2019) .

#### **6.4.1.1 Avantages** :

- Open source & gratuit
- Facile à apprendre
- Flexible et évolutif
- Facilement maintenable
- Englobe le Framework Angular
- Supporte Angular Material Design
- Communauté très forte et nombreuse car Ionic 4 est basé sur Angular 7, ce qui résulte d'un support très fiable depuis des années et fournit des solutions rapides en cas d'erreurs ou de mises à jour
- Expert dans les technologies avancées comme SASS 4, HTML 5 et Typescript pour le développement de composants

#### **6.4.1.2 Inconvénients :**

- Nécessite l'utilisation des plugins Cordova pour avoir accès aux fonctionnalités natives de l'appareil mobile
- Limité en termes de performance et de réactivité par rapport aux concurrents

### 6.4.2 React Native

Développé courant 2015, React Native est devenu une des meilleures alternatives pour construire des applications proches du natif. C'est initialement Facebook qui a fait connaître cette plateforme pour construire des applications. Le Framework permet aux développeurs de coder plus facilement en leur permettant de construire, à l'aide d'un seul code JavaScript, des applications mobiles interactives qui fonctionnent aussi bien sous iOS que sous Android. Les applications développées avec React Native permettent la mise en place d'une interface utilisateur attractive avec des modules natifs. Les applications comme Uber Eats, Instagram ou même Bloomberg sont les exemples phares de ce Framework (Rushi Trivedi 2019).

#### 6.4.2.1 Avantages :

- Couvre iOS et Android
- Rendement et expérience utilisateur très similaire aux applications natives grâce aux composants indépendants et réutilisables
- Possibilité de réutiliser des composants UI à l'intérieur d'autres applications, sans devoir réécrire le code
- Compatibilité avec de nombreux plugins tiers
- Sollicite moins la mémoire du processeur
- Excellent temps de réponse en termes de performance

#### 6.4.2.2 Inconvénients :

- Nécessite davantage de temps de développement, le code devant être adapté pour les différents systèmes d'exploitation
- Nécessite une certaine expérience avec le développement d'applications natives

### 6.4.3 Flutter

Nouveau pilier chez les Frameworks de développement hybride, Google's Flutter est utilisé pour développer des interfaces natives de haute qualité sur Android et iOS. Ce Framework a été lancé en mai 2017. C'est une plateforme de développement d'application mobile open-source. Flutter est rapide et assiste le développeur dans la construction d'applications multi-plateformes (Rushi Trivedi 2019).

#### 6.4.3.1 Avantages :

- Plateforme open-source
- Utilise le langage de programmation Dart, orienté objet et fortement typé
- Bénéficie d'une large gamme de composants d'interfaces graphiques qui peuvent être personnalisés selon des besoins spécifiques
- Concurrence React Native en termes de performance et de réactivité
- Interfaces simples d'utilisation

#### 6.4.3.2 Inconvénients :

- Nécessite de se familiariser le langage de programmation Dart
- Le Framework est relativement récent et toujours en développement et améliorations, ce qui peut relever des failles
- La communauté jeune et peu nombreuse à l'heure actuelle comparé à ses concurrents malgré le fait que Flutter est soutenu par la communauté Google, on envisage donc une rapide évolution

### 6.4.4 Comparatif détaillé

Afin d'avoir un meilleur aperçu des différences entre les trois Framework cités ci-avant, j'ai créé un tableau comparatif avec des caractéristiques communes :

Tableau 15 : Comparatif – React Native VS Ionic VS Flutter

| | **React Native** | **Ionic** | **Flutter** |
|---|---|---|---|
| **Langage** | JavaScript & React | HTML, CSS, Javascript (on peut également l'utiliser avec React, VueJS ou Angular) | Dart |
| **Types d'applications** | Multi-plateformes | Hybrides multi-plateformes | Multi-plateformes |
| **Fondé en** | Mars 2015 | 2013 | Mai 2017 |
| **Développé par** | Facebook & la communauté | Drifty Co. | Google & la communauté |
| **Support communautaire** | Fort | Fort | Manque de communauté car récent et jeune |
| **Support Front-end** | Composants natifs & interface utilisateur déclarative | HTML, CSS et une large gamme de designs utilisateurs | Interfaces utilisateurs attractives avec des composants intégrés |
| **Réutilisabilité du code** | « Learn once, write everywhere | « Write once, use anywhere » | Composants d'interfaces graphiques réutilisables |
| **Performance** | Rapide et similaire aux Framework natifs | Interactif et applications plus rapides | Haute performance et améliorations graphiques possibles |

(Spec India, Juillet 2019)

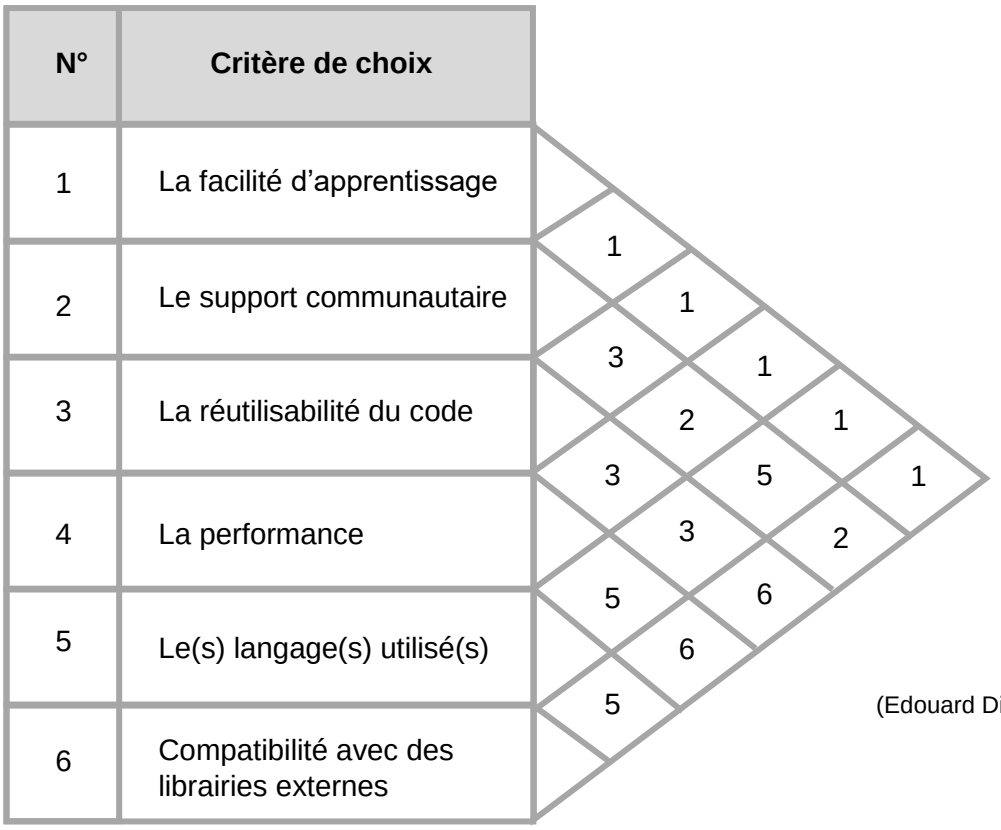
## 6.5 Choix du Framework

Afin de confirmer que Ionic corresponde bien à mes besoins, j'ai établi une matrice préférentielle avec les critères que j'ai jugé les plus importants dans mon choix de Framework hybride :

Voici la liste des critères :

1. La facilité d'apprentissage
2. Le support communautaire
3. La réutilisabilité du code
4. La performance
5. Le(s) langage(s) utilisé(s)
6. Compatibilité avec des librairies externes

Figure 4 : Matrice préférentielle des critères de choix pour un Framework




(Edouard Diep 2019)

Tableau 16 : Pondération de la matrice préférentielle basée sur les critères

| N° | Critère | Nombre (15) | Ordre | Pondération % | Pond. Modérée % |
|---|---|---|---|---|---|
| 1 | Facilité d'apprentissage | 5 | 1 | 33.33 | 34 |
| 2 | Support communautaire | 2 | 3 | 13.33 | 14 |
| 3 | Réutilisabilité du code | 3 | 2 | 20 | 20 |
| 4 | Performance | 0 | --- | --- | --- |
| 5 | Langage(s) utilisé(s) | 3 | 2 | 20 | 20 |
| 6 | Compatibilité librairies ext. | 2 | 4 | 13.33 | 12 |

(Edouard Diep 2019)

Dans le cadre de l'utilisation d'un Framework de développement hybride, j'estime que la priorité reste l'aptitude à apprendre le Framework en question. C'est ainsi que l'on maximise le rendement du code, tout en minimisant le temps d'apprentissage sur la durée. De plus, le code se doit d'être réutilisable afin d'éviter de dupliquer des lignes là où nous pouvons simplement créer des références vers des composants ou des appels vers des fonctions réutilisables par exemple.

Le langage utilisé est un critère de choix que j'estime complémentaire à la facilité d'apprentissage, car c'est en connaissant un langage au préalable, qu'on assure une certaine aisance dans l'apprentissage basé sur nos connaissances. Dans le cas d'Ionic 4, le fait d'avoir déjà travaillé avec Angular 7 facilite énormément l'utilisation de Typescript et des librairies externes propres à Ionic. Par ailleurs, le fait de maîtriser les langages HTML et CSS permet d'exploiter le véritable potentiel des composants graphiques d'Ionic.

Le support communautaire se doit, quant à lui, d'être toujours à jour et disponible afin d'éviter de perdre du temps à chercher une solution en cas de problèmes ou bugs imprévus.

La performance n'a finalement pas fait partie des critères mis en avant pour le projet VaudVin. J'estime que cette dernière est à réévaluer une fois l'application déployée et la montée en charge assurée, mais qu'il est difficile d'anticiper un tel aspect lors de la phase construction de l'application.

Tableau 17 : Analyse multicritères d'aide à la décision

| **Critères obligatoires** | | **Ionic** | | **React Native** | | **Flutter** | |
|---|---|---|---|---|---|---|---|
| Multi-plateformes | | **OUI** | | **OUI** | | **OUI** | |
| Compatible avec le GPS du Smartphone | | **OUI** | | **OUI** | | **OUI** | |
| **Critères facultatifs** | **Pondération** | **Ionic** | | **React Native** | | **Flutter** | |
| | | Coef. | Points | Coef. | Points | Coef. | Points |
| Facilité d'apprentissage | 34 | 8 | 272 | 8 | 272 | 7 | 238 |
| Support communautaire | 14 | 9 | 126 | 7 | 98 | 3 | 42 |
| Réutilisabilité du code | 20 | 7 | 140 | 6 | 120 | 6 | 120 |
| Performance | 0 | --- | --- | --- | --- | --- | --- |
| Langage(s) utilisé(s) | 20 | 9 | 180 | 7 | 140 | 4 | 80 |
| Compatibilité librairies ext. | 12 | 8 | 96 | 5 | 60 | 9 | 108 |
| **Total** | | | **814** | | **690** | | **588** |

(Edouard Diep 2019)

Mon choix s'est finalement porté sur le Framework Ionic qui semble être le plus à même de satisfaire tous les critères les plus importants, selon moi. En effet, étant donné mon expérience avec les technologies web HTML, CSS et le Framework Angular 7 sur lequel Ionic se base, il me semblait pertinent de partir sur un tel choix pour ce projet.

# 7. Analyse des Frameworks Back End

## 7.1 Web Service

Par définition, un web service est une technologie permettant à des applications de communiquer à distance via Internet et ceci indépendamment des plates-formes et des langages sur lesquelles elles reposent. Cette communication est basée sur le principe de demandes et réponses, effectuées avec des messages formatés à l'aide de protocoles Internet très répandus comme XML ou HTTP (UNIV 2019). Il existe deux types de web services répandus aujourd'hui :

- SOAP (Simple Object Access Protocol)
- REST (Representational State Tranfer)

C'est à l'architecture REST que nous allons nous intéresser dans les chapitres suivants. Tout d'abord, qu'est-ce que REST ? C'est un ensemble de principes architecturaux par lesquels les données sont transmises sur une interface standardisée (telle que HTTP).

## 7.2 API REST

REST ne s'intéresse qu'aux règles de conception de services dits "stateless" ou sans état. Un client accède à la ressource via un URI (élément permettant d'identifier une ressource) unique et une représentation de cette dernière est ensuite retournée. Avec chaque nouvelle représentation de la ressource, on dit que le serveur ne stocke pas d'état client de son côté (Thomas ROSSIER 2019). Quatre opérations principales sont gérées par REST :

- GET : demande une représentation de la ressource existante
- POST : crée et envoie au serveur une nouvelle ressource
- PUT : modifie une ressource existante sur le serveur
- DELETE : supprime une ressource existante du serveur

Le grand avantage de REST par rapport à SOAP, est qu'il perçoit le Web comme un ensemble de ressources dont l'une des représentations est le HTML, mais propose également des formats XML, JSON, CSV, etc. A l'inverse, SOAP permet des actions plus complexes et amène une meilleure sécurité, mais requiert plus de ressources et impose une représentation XML des données (Thomas ROSSIER 2019). Toutefois,

dans le cadre du projet VaudVin, nous utiliserons REST qui suffit largement à satisfaire les besoins au niveau du Back end de l'application.

## 7.3 Framework Back end

Les Framework Back end sont responsables de la logique côté serveur. C'est en quelque sorte la face cachée de l'iceberg, ce qui prend place derrière l'interface utilisateur et permet au Front end de pouvoir travailler avec une base de données (Hackr.io 2019).

Comment gérer ou stocker des données, répondre aux requêtes envoyées par le Front : tout ceci constitue un ensemble de fonctionnalités gérées par un Framework Back end. À titre de comparaison, j'ai sélectionné deux Frameworks de développement Back end très répandus en 2019 :

- Laravel
- Django

### 7.3.1 Laravel

Laravel est un Framework open-source et gratuit, basé sur le modèle d'architecture logicielle MVC (Modèle-Vue-contrôleur). Il a été créé par Taylor Otwell en 2011 et construit entièrement en PHP, un des langages les plus connus sur le web. Il a pour but de réduire le coût d'entrée en développement et d'améliorer la qualité du code en définissant des pratiques de design standards. En utilisant Laravel, un développeur économisera des heures de développement et réduira des milliers de lignes de codes grâce notamment à l'ORM (Object-Relational Mapping) du nom de "Eloquent" inclus avec Laravel, qui améliore grandement la syntaxe du code pour les requêtes complexes (Nilanchala, Stacktips 2017).

### 7.3.2 Django

Django est un Framework basé sur le modèle d'architecture MVT (Modèle-vue-template). Il a été construit en Python. C'est un Framework open-source et gratuit qui encourage le développement rapide et est relativement puissant et utilisé par plusieurs grandes entreprises dans le monde en tant que gestionnaire d'infrastructure Back end. Ces dernières incluent entre autres Pinterest, Udemy, la NASA et Instagram (Hackr.io, 2019).

## 7.4 Choix du Framework

A priori, Laravel et Django sont parfaitement équivalents en termes de caractéristiques et assurent, à quelques détails près, les mêmes fonctionnalités attendues par un Framework Back end hautement répandu. Voici ci-dessous un tableau comparatif des deux Framework sur l'aspect performance :

Tableau 18 : Comparatif – Django VS Laravel

| | **Django** | **Laravel** |
|---|---|---|
| **Définition** | Framework Full Stack d'application web écrit en Python. | Framework Full Stack d'application web écrit en PHP. |
| **Maintenance** | Maintenu par Django Software Foundation. | Maintenu par le développeur lui-même et sa communauté sous la licence MIT. |
| **Architecture** | Modèle-Vue-Template (MVT). | Modèle-Vue-Contrôleur (MVC). |
| **Plateforme** | Multi-plateformes. | Multi-plateformes. |
| **Généralité** | Privilégie un développement rapide avec une large communauté d'utilisateurs. | Privilégie une architecture de code élégante avec une communauté d'utilisateurs grandissante. |

| | | |
|---|---|---|
| **Montée en charge** | Supporte une forte montée en charge. | Supporte également une forte montée en charge. |
| **Standardisation** | Large communauté de développeurs. | Large communauté de développeurs. |
| **Compatibilité** | Plusieurs autres Frameworks existent avec Python (Web2py, Flask, etc.). | Basé sur le Framework Symfony et le seul Framework considéré pour PHP. |

(Educba.com 2019)

Après longue réflexion, mon choix s'est finalement orienté vers Laravel car j'ai déjà étudié et pratiqué le langage PHP auparavant et me sentais particulièrement à l'aise avec les différentes syntaxes du code. Il me semblait alors évident de choisir un Framework qui me permettrait de mettre en pratique mes connaissances déjà acquises afin d'accélérer le temps d'apprentissage.

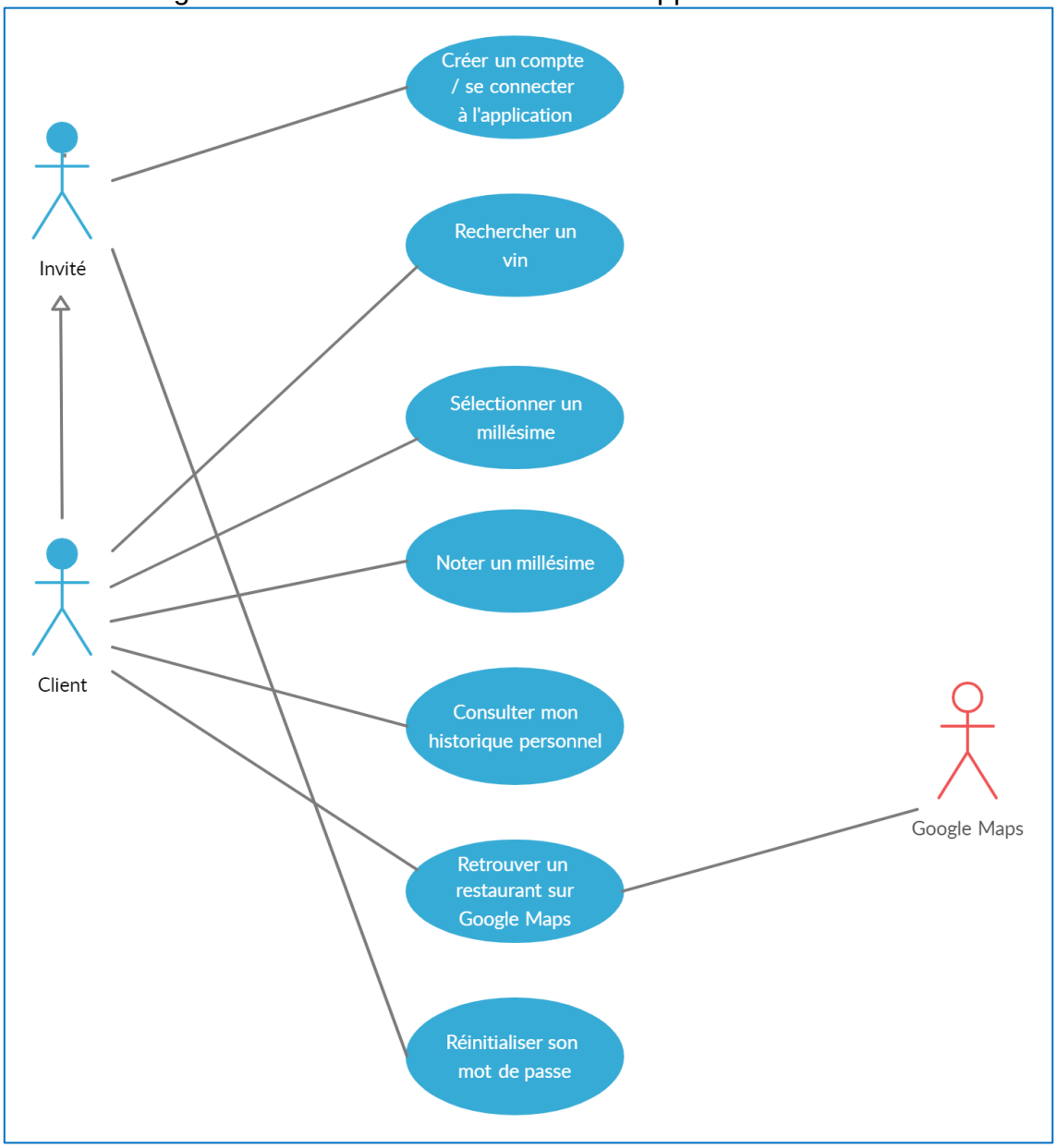
De plus, j'ai été agréablement surpris de constater à quel point l'ORM (Object-relational Mapping) Eloquent de Laravel facilite la syntaxe des requêtes SQL et permet d'effectuer très rapidement et aisément des requêtes complexes à travers des modèles liés aux données. Le Routing simple et optimisé fourni par Laravel, a aussi été un critère de choix afin de faciliter l'accès aux données à travers les méthodes existantes du Framework (Fullscale.io 2019).

A l'inverse, je n'ai jamais eu de familiarité avec la syntaxe du langage Python, ce qui m'aurait contraint d'apprendre la syntaxe et les particularités du langage en même temps que le Framework et par conséquent, davantage de temps et de ressources à investir dans le cadre d'un projet de telle envergure. J'ai donc préféré miser sur l'apprentissage rapide. Cela dit, il est important de souligner que les deux Frameworks présentés ci-dessus sont très puissants, complets et riches en documentation et s'équivalent parfaitement en termes de fonctionnalités, performance et sécurité.

# 8. Fonctionnalités de haut-niveau

Toutes les fonctionnalités de l'application ont été développées dans la mesure du possible, avec l'aide de plugins ou librairies externes déjà construites par des développeurs du Framework, dans le but de réutiliser ce qui existe déjà sans devoir réinventer la roue. Ainsi, on assure une maintenabilité et une compatibilité optimale pour la suite du projet. Voici ci-dessous un modèle présentant les fonctionnalités de haut-niveau de l'application VaudVin, plus connues sous le nom de "use-cases" (UC) :

Figure 5 : Modèle des use-cases de l'application VaudVin



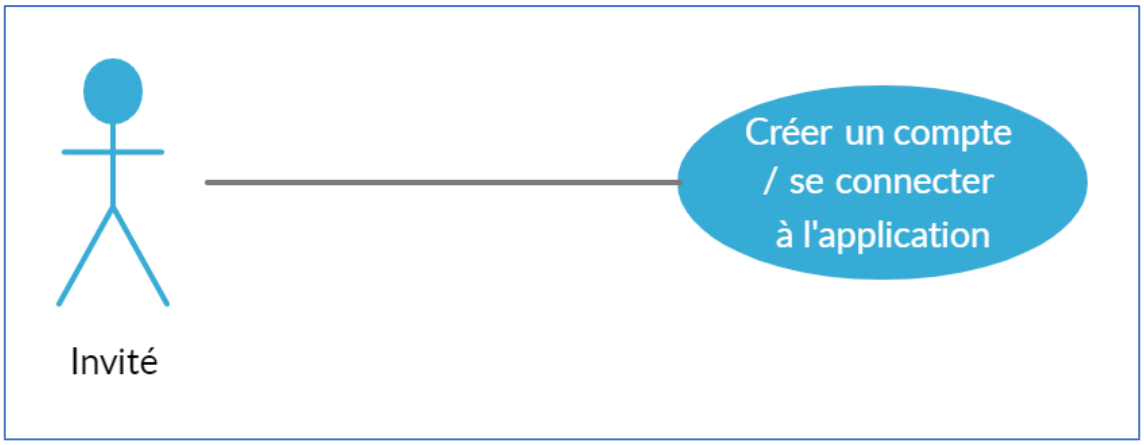

(Edouard Diep 2019)

Les use-cases listés ci-dessus ont été classés par ordre de Business Value (BV) selon accord avec le mandant du projet. La BV définit l'importance d'une fonctionnalité devant être implémentée aux yeux du mandant. On retrouve donc la plus importante en haut du schéma et la moins importante en bas (cf : Figure 5 ci-avant).

## 8.1 Créer un compte / se connecter à l'application

"*En tant qu'invité, je veux pouvoir créer un compte et me connecter à l'application afin de pouvoir utiliser la plate-forme.*"

Figure 6 : Modèle UC – Créer un compte / se connecter à l'application




(Edouard Diep 2019)

Dans un premier temps et en arrivant sur l'application, l'utilisateur doit pouvoir se connecter à l'application. Pour ce faire, il a la possibilité de créer un compte via un formulaire de création de compte ou de se logger avec son e-mail et son mot de passe. Le compte sert à stocker les informations (notes, commentaires, recommandations) sur un profil propre à chaque utilisateur de l'application.

Flot de base :

1. L'invité rentre son e-mail dans le champ approprié
2. L'invité rentre son mot de passe dans le champ approprié
3. L'invité clique sur le bouton : "Se connecter"
4. Le système redirige l'invité sur la page d'accueil de l'application

Flot alternatif :

3a. L'e-mail ou le mot de passe est incorrect

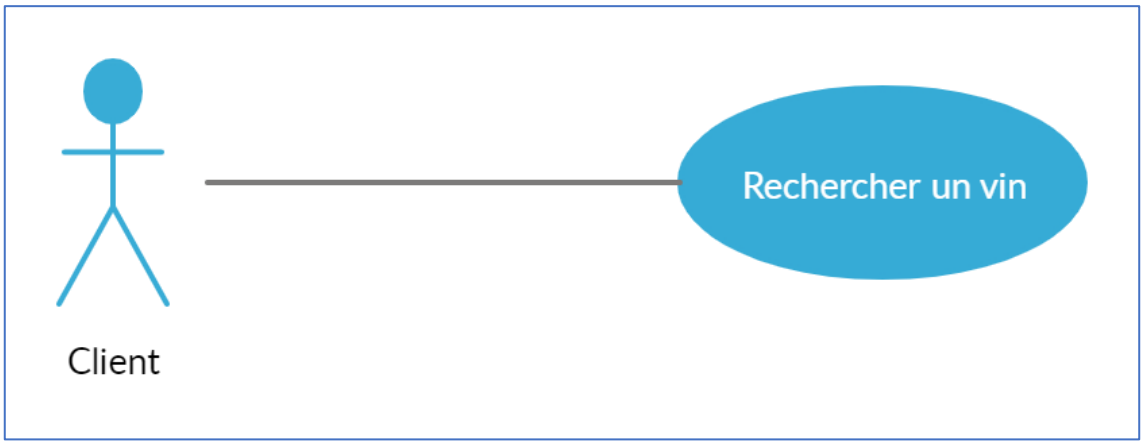
3a1. Le système affiche une alerte d'erreur pour informer l'utilisateur

3a2. Suite en 1

## 8.2 Rechercher un vin

*"En tant que client, je veux pouvoir rechercher un vin dans une liste afin de consulter sa fiche globale."*

Figure 7 : Modèle UC – Rechercher un vin




(Edouard Diep 2019)

Une fois connecté sur l'application, une des fonctionnalités phares du projet est de permettre au client de pouvoir rechercher un vin parmi une liste déjà existante dans la base de données VaudVin. Pour ce faire l'utilisateur doit parvenir, à l'aide du menu interactif, à la page "Noter / évaluer un vin" sur laquelle il trouvera la liste en question avec une barre de recherche.

Flot de base :

1. Le client défile la liste des vins existants
2. Le client clique sur un vin
3. Le système redirige le client vers la page de sélection du millésime

Flot alternatif :

1a. Le client ne trouve pas son vin en défilant la liste

1a1. Le client tape le nom du vin qu'il connaît dans la barre de recherche

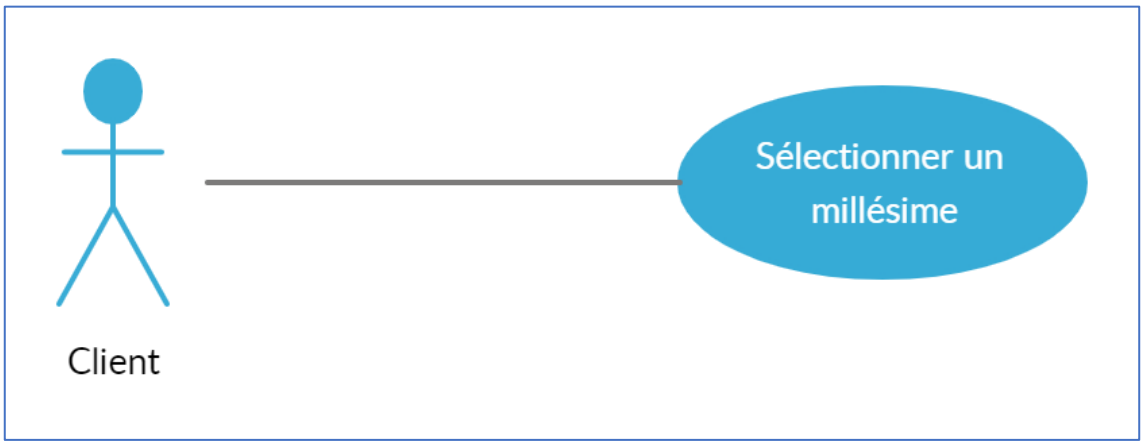
1a2. Le système filtre la liste des vins avec les résultats de recherche

1a3. Suite en 2

## 8.3 Sélectionner un millésime

*"En tant que client, je veux pouvoir sélectionner le millésime d'un vin afin de consulter sa fiche détaillée."*

Figure 8 : Modèle UC – Sélectionner un millésime




(Edouard Diep 2019)

Une fois le vin trouvé dans la liste, le client doit pouvoir sélectionner un millésime correspondant au modèle choisi. Il a alors la possibilité de faire un choix dans une liste déroulante contenant les millésimes triés par ordre décroissant, allant de 2019 à 1990.

Flot de base :

1. Le client ouvre la liste en cliquant sur : "Sélectionner un millésime"
2. Le client choisit un millésime dans la liste
3. Le client clique sur : "Valider" afin de confirmer son choix
4. Le système redirige le client sur la fiche du vin détaillée

Flot alternatif :

3a. Le millésime pour ce vin n'a jamais été sélectionné auparavant

3a1. Le système crée un nouveau millésime dans la base de données

3a2. Suite en 4

3b. Le millésime pour ce vin existe déjà dans la base de données

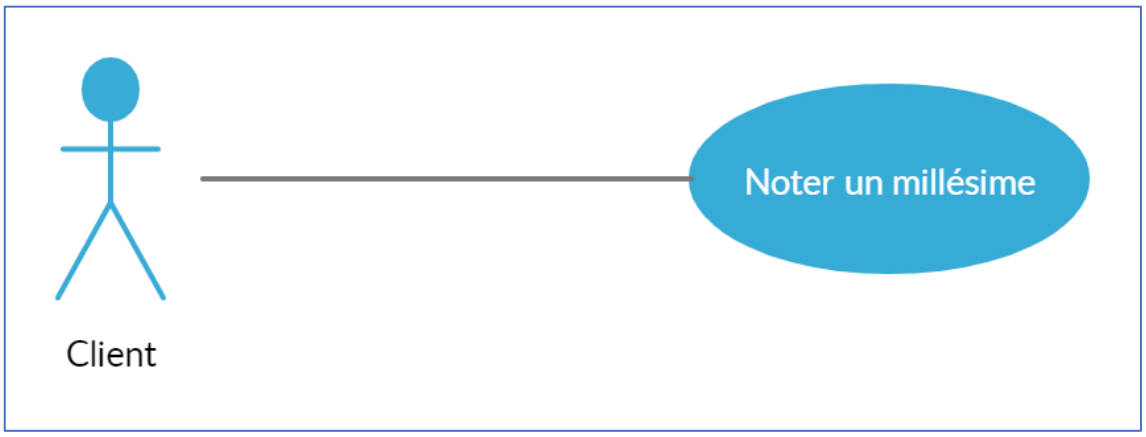
3b1. Le système récupère le millésime existant lié au vin sélectionné

3b2. Suite en 4

## 8.4 Noter un millésime

*"En tant que client, je veux pouvoir noter un millésime afin de le retrouver dans mon historique personnel."*

Figure 9 : Modèle UC – Noter un millésime




(Edouard Diep 2019)

Une fois sur la fiche du vin détaillé, le client doit pouvoir noter le millésime sur lequel il se trouve. Pour ce faire, une série de "cœurs" lui permet de sélectionner une note sur une échelle de 1 à 5 (1 étant médiocre et 5 excellent). Le client a ensuite le choix ou non d'écrire un commentaire, d'un maximum de 100 caractères, qui ne sera pas publié, mais restera uniquement visible par l'auteur de ce dernier. Une fois la note saisie, le client enregistre son choix et le système stocke la note dans la base de données.

Flot de base :

1. Le client clique sur le nombre de cœurs correspondant à sa note
2. Le client saisit un commentaire de maximum 100 caractères dans le champ texte correspondant
3. Le client clique sur : "Enregistrer l'avis" afin de sauvegarder sa note et éventuellement son commentaire
4. Le système enregistre l'avis du client dans la base de données
5. Le système propose au client de continuer à noter des vins ou consulter son historique personnel
6. Le client sélectionne un choix parmi deux propositions
7. Le système redirige le client vers la page correspondante

Flot alternatif :

3a. Le client n'a pas sélectionné de note

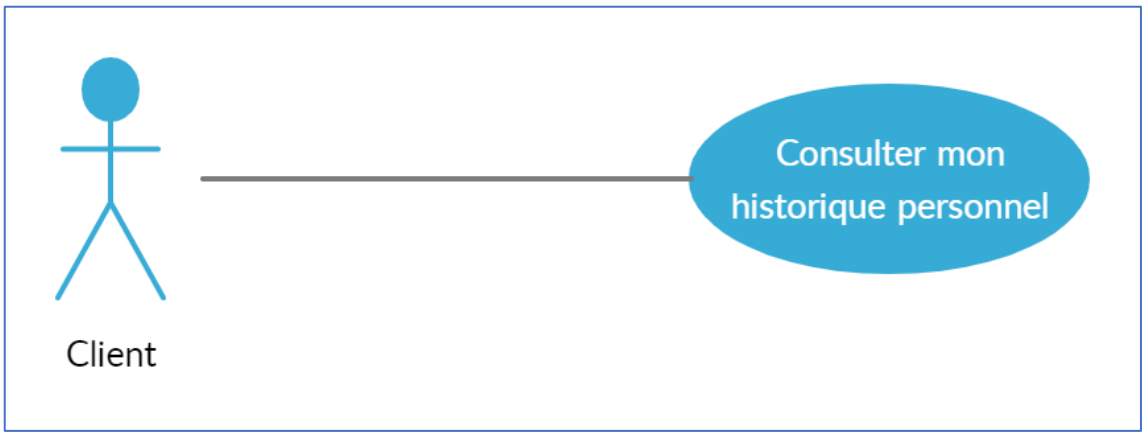
3a1. Le système affiche un message d'erreur demandant au client de corriger

3a2. Suite en 1

## 8.5 Consulter mon historique personnel

*"En tant que client, je veux pouvoir consulter mon historique personnel afin de retrouver mes vins déjà évalués."*

Figure 10 : Modèle UC – Consulter mon historique personnel




(Edouard Diep 2019)

Une fois un millésime évalué et l'avis enregistré dans la base de données, le client peut accéder à la page de son historique personnel via le menu interactif de l'application. Sur cette page, il peut consulter la liste des vins qu'il a évalués, triés par catégorie (rouges, blancs, rosés ou mousseux & spécialités). Les notes sont affichées sous forme de cœurs. Dans le cas où le client n'a évalué qu'un seul millésime, alors c'est la note de ce dernier qui est affichée. À l'inverse, si le client a évalué plusieurs millésimes pour un seul vin, alors la note affichée est la moyenne des notes de tous les millésimes du vin. Enfin, si le client n'a jamais évalué de vin en arrivant sur cette page, le système affiche un message indiquant au client qu'aucun vin n'a encore été évalué.

Flot de base :

1. Le client accède à la page de son historique personnel
2. Le client déroule la ou les catégories de vin dont il aimerait consulter les notes
3. Le client clique sur un vin dont il aimerait modifier la note
4. Le système redirige le client vers la page de sélection du millésime

Flot alternatif :

2a. Le client n'a pas encore évalué de vin en arrivant sur son historique

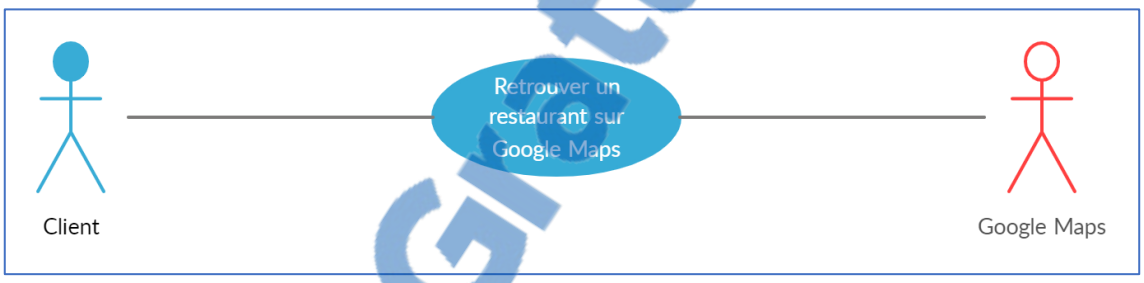
2a1. Le système affiche un message : "Aucun vin n'a encore été évalué"

2a2. Fin UC

## 8.6 Retrouver un restaurant sur Google Maps

*"En tant que client, je veux pouvoir retrouver sur une carte Google Maps la position d'un ou plusieurs restaurant(s) à proximité afin de consulter leur carte des vins liée."*

Figure 11 : Modèle UC – Retrouver un restaurant sur Google Maps




(Edouard Diep 2019)

Une fois connecté à l'application, la seconde grande fonctionnalité phare de l'application consiste à afficher une carte à l'aide de l'API Google Maps et de permettre au client de pouvoir sélectionner un des restaurants stockés dans la base de données VaudVin et affichés sur cette dernière dans le but de consulter leur carte des vins. Seuls les restaurants stockés en base de données sont affichés sur la carte. Cette fonctionnalité sera à améliorer dans une version future de l'application.

Flot de base :

1. Le client accède à la page de recherche d'un restaurant
2. Le client défile la carte afin de retrouver les restaurants sous forme de marqueurs rouges
3. Le client clique sur un marqueur afin d'afficher les détails du restaurant (adresse, code postal, numéro de téléphone, lien vers la carte)
4. Le client clique sur : "Afficher la carte des vins"
5. Le système redirige le client vers la carte des vins liée au restaurant sélectionné

### 8.6.1 API Google Maps

Google Maps est un service de cartographie en ligne proposé par le géant Google, initialement lancé en 2004 aux États-Unis puis introduit deux ans plus tard en Europe. C'est un service multi-plateformes qui permet, à partir de l'échelle mondiale, de zoomer jusqu'à l'échelle d'une habitation. Des prises de vue fixes montrant les détails de certaines rues sont également accessibles grâce à la fonctionnalité Google Street View (Wikipédia, 2019).

Le mandant du projet est à la recherche d'un moyen de pouvoir géolocaliser les emplacements des restaurants à proximité du client connecté à l'application et de pouvoir afficher ces restaurants sur une carte, puis de pouvoir appuyer dessus pour accéder à leur carte des vins : tout cela bien évidemment à travers l'application mobile.

La première idée qui m'est venue en tête était l'API Google Maps pour sa réputation et sa haute popularité. J'ai donc immédiatement proposé cela au mandant sans réellement avoir étudié le fonctionnement ni la marche à suivre pour implémenter cette fonctionnalité, en étant convaincu qu'il s'agissait d'un système fiable et très bien documenté.

J'ai été surpris d'apprendre que depuis le 11 juin 2018, le service de Google Maps est devenu payant pour les professionnels désirant étendre le nombre d'appels API par jour. En effet, la plateforme offre un "crédit" mensuel de $200 (deux cents dollars) au-delà duquel il faut débiter la carte bancaire. Auparavant, il était possible d'effectuer pas moins de 25'000 appels API par jour gratuitement. Désormais, le nombre d'appels au quotidien est limité en fonction du type d'utilisation (ZDNet 2018). Sur la page suivante, je vous présente un tableau listant les différents services possibles avec le nombre d'appels quotidiens pour chacun, compris dans le forfait de 200 dollars "offerts" par Google :

Tableau 19 : Nombre de requêtes API Google Maps par type de service

| **Maps** | |
|---|---|
| Cartes statiques natives mobiles | Sans limite |
| Cartes dynamiques natives mobiles | Sans limite |
| Intégration | Sans limite |
| Intégration avancée | 35 |
| Cartes statiques | 249 |
| Cartes dynamiques | 71 |
| Street View statique | 71 |
| Street View dynamique | 35 |
| **Itinéraires** | |
| Directions | 115 |
| Directions avancées | 57 |
| Matrice de distance | 115 |
| Matrice de distance avancée | 57 |
| Routes parcourues | 57 |
| Route la plus proche | 57 |
| **Lieux** | |
| Autocomplète par caractère | 176 |
| Requête autocomplète par caractère | 176 |
| Autocomplète (avec les détails des lieux) par session | 29 |
| Autocomplète (avec les détails des lieux) par session avec les contacts | 24 |
| Autocomplète (avec les détails des lieux) par session avec les contacts et données atmosphère | 19 |
| Autocomplète sans les détails de lieux par session | 29 |
| Détails de lieu | 29 |
| Détails de lieu et données de contact | 24 |
| Détails de lieu et données de contact avec données atmosphère | 22 |
| Détails de lieu et refresh de l'ID | Sans limite |
| Trouver le lieu | 29 |
| Trouver le lieu et données de contact | 24 |
| Trouver le lieu et données atmosphère | 22 |
| Trouver le lieu et données de contact et données atmosphère | 19 |
| Trouver le lieu uniquement avec IC | Sans limite |
| Photo des lieux | 71 |
| Lieux - recherche à proximité et données de contact et données atmosphère | 12 |
| Lieux - recherche textuelle et données de contact et données atmosphère | 12 |
| Géocodage | 99 |
| Géolocalisation | 99 |
| Times zone | 99 |
| Altitude | 99 |

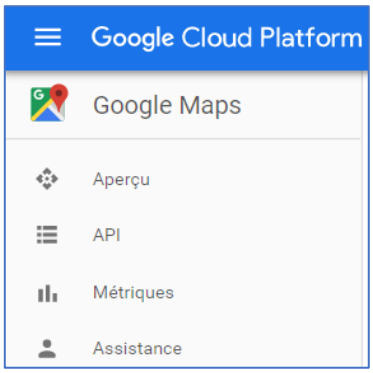
(La rédaction de ZDNet.fr 2018)

Comme nous pouvons le constater ci-dessus, le nombre de requêtes API par jour est très limité selon l'utilisation. Dans le contexte de VaudVin, nous nous servons des lieux pour obtenir les détails de chaque restaurant sur la carte, mais également le géocodage (décrit dans le sous-chapitre suivant) qui nous permet de situer la position de l'utilisateur et des restaurants sous forme de marqueurs sur la carte.

#### 8.6.1.1 Fonctionnement de l'API

Pour toute utilisation de l'API Google Maps, il est nécessaire de générer une API Key directement sur la console de la plateforme Google Cloud. Pour ce faire, il faut donc se rendre sur le lien de la plateforme et cliquer sur l'onglet "API" sur le menu latéral (cf : Figure 11 : console Google Cloud)

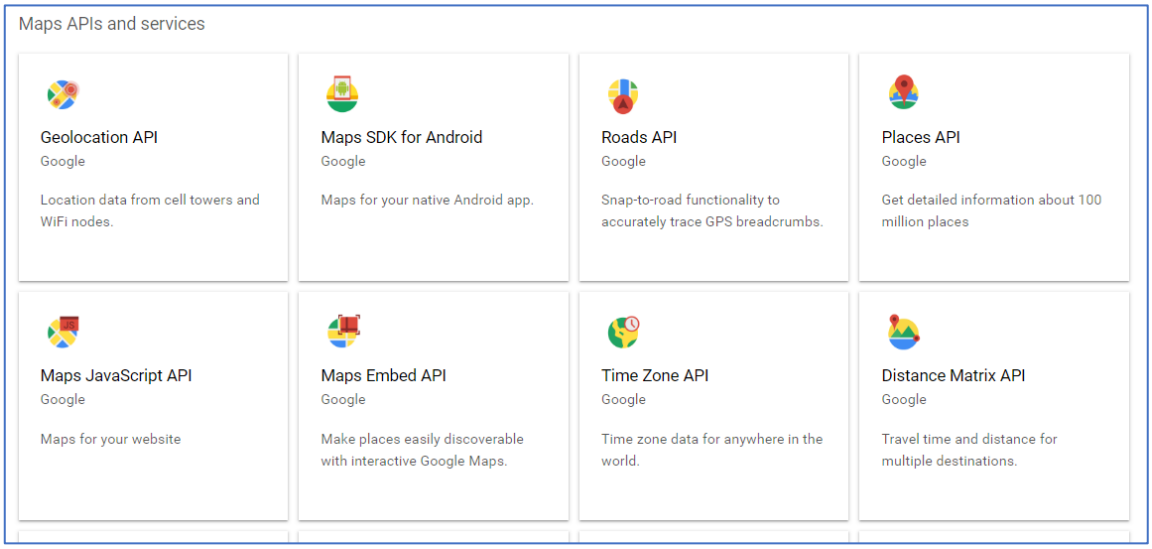
Figure 12 : Google Cloud Platform side menu




(Google Cloud Platform 2019)

Une fois sur la page de l'API, une liste de services nous est proposée et il s'agit de sélectionner celui désiré :

Figure 13 : Maps APIs and services




(Google Cloud Platform 2019)

Il faut ensuite inscrire ses coordonnées bancaires afin d'obtenir une clé selon l'offre sélectionnée. Google nous propose alors trois choix pour chacun des services :

- Maps
- Places
- Roads

En fonction des choix sélectionnés, une Key unique va être générée et cette dernière doit être rajoutée au projet Ionic dans le fichier "index.html" :

Figure 14 : API Google maps script – Fichier index.html

```
<!-- API Google maps necessary script -->
<script type="text/javascript" src="https://maps.googleapis.com/maps/api/js?key=XX"></script>
```

(Edouard Diep 2019)

Ainsi que dans le fichier "app.modules.ts" qui gère les importations des composants, directives et providers du projet :

Figure 15 : API Google maps unique key – Fichier app.modules.ts

```
  AgmCoreModule.forRoot({
    apiKey: 'AIzaSyCeRuufysM9JnUKkN5FXkd4vxDBIwHCV2Q',
  }),
],
```

#### 8.6.1.2 Géocodage

Le Géocodage ou Geocoding est un processus de conversion d'adresses (*"Rue de la Tambourine 17, 1227 Carouge")* en coordonnées géographiques, exprimées par une latitude (*46.175646*) et une longitude (*6.1391615*), qui nous permet de pouvoir utiliser des marqueurs que nous pouvons placer sur une carte, afin de situer des lieux précis. L'action inverse se nomme le "reverse geocoding" et est également géré par l'API Google Maps. On peut aussi y ajouter des paramètres qui permettent d'affiner les résultats de recherche. (Google Maps Platform – Geocoding Service 2019)

Dans le cadre de l'application VaudVin, la fonctionnalité Geocoding est utilisée afin de convertir les adresses des restaurants récupérées depuis la base de données, pour pouvoir placer des marqueurs rouges qui les situent sur la carte du monde. Le but étant de pouvoir récupérer les informations détaillées des restaurants géocodés et ainsi pouvoir accéder à leur carte des vins.

#### 8.6.1.3 Géolocation

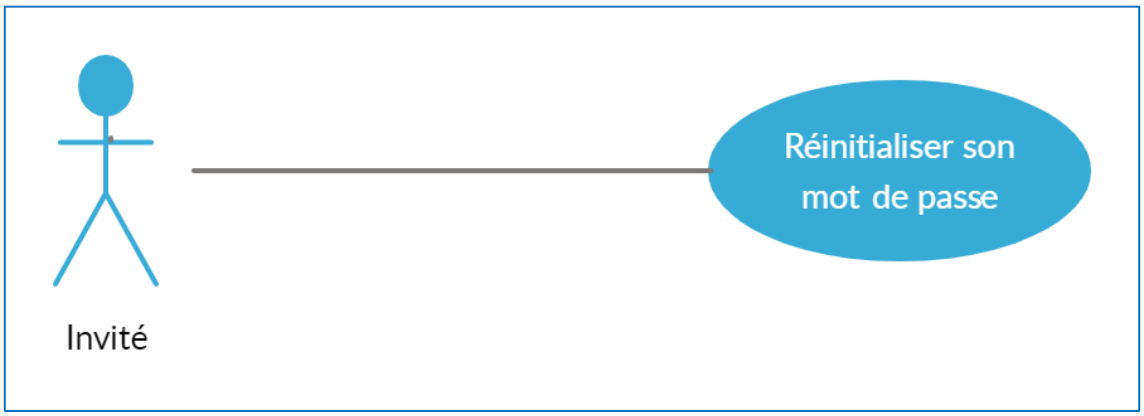
La Géolocation est un processus permettant de trouver, déterminer et fournir la position exacte d'un appareil connecté à un réseau cellulaire ou WiFi à travers sa puce réseau et/ou GPS intégré. Cela permet d'obtenir les coordonnées géographiques (latitude et longitude) de ce dernier et retourne sa position.

Vis-à-vis du projet VaudVin, la Géolocation est utilisée afin de situer la position du client ou l'utilisateur connecté à l'application à travers son appareil (téléphone ou tablette) et affiche un marqueur bleu sur la carte.

## 8.7 Réinitialiser son mot de passe

*"En tant qu'invité, je veux pouvoir réinitialiser mon mot de passe, afin de me reconnecter à l'application après avoir oublié mes identifiants."*

Figure 16 : Modèle UC – Réinitialiser son mot de passe




(Edouard Diep 2019)

Il n'est pas anodin de constater qu'un utilisateur ne se souvienne pas de son mot de passe lorsqu'il tente de se reconnecter après un certain temps. C'est pourquoi, j'ai jugé utile de donner la possibilité à l'invité ayant oublié ses identifiants, de pouvoir réinitialiser son mot de passe à l'aide d'un e-mail envoyé sur sa boîte de messagerie qui le redirige vers une page, afin de lui permettre de changer son mot de passe.

Flot de base :

1. Depuis la page login, l'invité clique sur le lien : "Mot de passe oublié ?"
2. Le système redirige l'invité vers une page de réinitialisation du mot de passe
3. L'invité rentre son adresse e-mail dans le champ texte
4. L'invité clique sur "Envoyer"
5. Le système envoie un e-mail de réinitialisation du mot de passe à l'invité et affiche un message de confirmation
6. L'invité ouvre l'e-mail et clique sur le bouton : "Réinitialiser le mot de passe"
7. Le système redirige l'invité vers un formulaire de réinitialisation
8. L'invité rentre son adresse e-mail ainsi que son nouveau mot de passe et confirme le mot de passe une seconde fois
9. L'invité clique sur le bouton : "Réinitialiser le mot de passe"
10. Le système enregistre le nouveau mot de passe dans la base de données et affiche un message de confirmation

Flot alternatif :

4a. L'e-mail est erroné ou mal saisi

4a1. Le système informe l'invité que l'e-mail n'existe pas et lui demande de corriger le champ correspondant

4a2. Suite en 3

9a. L'e-mail n'existe pas dans la base de données

9a1. Le système informe l'invité que l'e-mail n'existe pas et lui demande de corriger le champ correspondant

9a2. Suite en 8

# 9. Calcul du coefficient de correspondance

Le mandant du projet m'a communiqué un besoin très spécifique qu'il tenait à satisfaire avec le projet VaudVin : celui de vouloir recommander chaque vin à l'aide d'un pourcentage de recommandation, à des clients qui n'auraient pas encore évalué ni la bouteille, ni le millésime correspondant. Qui dit pourcentage, dit forcément calculs spécifiques devant être mis en place pour obtenir un résultat de correspondance précis.

J'ai donc dû implémenter un calcul permettant de retourner et afficher un coefficient de correspondance pour un utilisateur n'ayant pas encore évalué un vin en fonction de ses évaluations précédentes, ainsi que des notes des autres utilisateurs sur ce même vin.

Le calcul complet pour obtenir un coefficient de correspondance est le suivant :

*((NoteUtilisateurPondere / ratio) * NbNotesCommunes + MoyenneMillesime * CoefMoyenneMillesime + MoyenneConcours * CoefNoteConcours) / (NbNotesCommunes + CoefMoyenneMillesime + CoefNoteConcours) / 5*

J'ai décidé d'utiliser un code couleur dans le but de décomposer le calcul étape par étape, afin de décrire chaque donnée séparément l'une de l'autre. Je reprendrai les mêmes couleurs dans les sous-chapitres suivants, afin de détailler chaque étape du calcul.

Pour simplifier davantage la méthode de calcul, le mandant et moi-même nous sommes mis d'accord pour séparer les données à récupérer en quatre tables distinctes :

- Table des notes utilisateurs
- Table des notes concours
- Table de correspondance
- Table des pronostics

## 9.1 Table des notes utilisateurs

Cette table regroupe les notes attribuées par chacun des utilisateurs de l'application sur les différents vins stockés dans la base de données. En récupérant ces notes, nous pouvons dans un premier temps obtenir la note moyenne pour chaque millésime calculée comme suit : SommeNotesMillésime / NbNotes (cf : Tableau 20 ci-dessous).

Tableau 20 : Table des notes utilisateurs – Coefficient de correspondance

| Table des notes utilisateurs | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Vin** | **Millésime** | **Moyenne** | **User 1** | **User 2** | **User 3** | **User 4** | **User N** |
| Vin 1 | 2018 | 2,5 | | 3 | 2 | | |
| Vin 2 | 2017 | 4 | 3 | 5 | | 3 | |
| Vin 3 | 2017 | 2 | | 3 | 1 | 2 | |
| Vin 1 | 2017 | 4 | 4 | | 4 | | |
| Vin 1 | 2016 | 3,5 | 4 | | | 3 | |
| Vin n | | | | | | | |

(Edouard Diep 2019)

Dans un second temps, nous allons comparer les notes de chaque millésime pour un utilisateur qui n'a pas encore noté un vin et dont nous voulons connaître le coefficient de correspondance. Toujours en se référant au Tableau 20 ci-dessus, prenons pour exemple le User3 qui n'a pas encore évalué le Vin 2. Nous devons comparer les notes communes aux autres Users pour tous les autres vins (autres que Vin 2) que le User3 a déjà évalué, afin de stocker le nombre de notes communes ainsi que le nombre de notes similaires (qui nous sera utile pour obtenir la note de l'utilisateur pondérée). Si l'on devait analyser cela de plus près, nous pourrions isoler la partie du Tableau 20 comme suit :

Tableau 21 : Illustration – Notes communes et similaires

| | **User1** | **User2** | **User3** | **User4** |
|---|---|---|---|---|
| Vin 1 – 2018 | | 3 | 2 | |
| Vin 2 – 2017 | 3 | 5 | | 3 |
| Vin 3 – 2017 | | 3 | 1 | 2 |
| Vin 1 – 2017 | 4 | | 4 | |
| Vin 1 – 2016 | 4 | | | 3 |
| | | | | |

(Edouard Diep 2019)

Ci-dessus nous comparons les notes de User2 et User3, afin de déterminer lesquelles sont "communes" et lesquelles sont "similaires". Par définition, une note similaire est

forcément commune. Toujours du point de vue de User3, les notes communes définissent le nombre de millésimes que d'autres Users ont notés et qui ont été également notés par User3. Les notes similaires, quant à elles, représentent celles qui ont + ou – 1 d'écart entre une note et l'autre, c'est-à-dire soit 1 d'écart soit 0 (les deux mêmes notes). Dans l'exemple du Tableau 21 ci-avant, le User3 a deux notes communes (Vin 1 – 2018 et Vin 3 – 2017) et une seule note similaire (Vin 1 – 2018). Le fait de stocker le nombre de notes communes et similaires va permettre, par la suite, d'obtenir la note de l'utilisateur pondérée. Nous en discuterons plus en détail dans les sous-chapitres suivants.

## 9.2 Table des notes concours

C'est la partie la plus simple du calcul de coefficient de correspondance. En effet, les notes des concours étant stockées d'office dans la base de données et mises à jour une seule fois par année (après les concours annuels de vins), leurs valeurs restent inchangées et cette table nous permet uniquement de récupérer la moyenne des notes des concours pour chaque millésime. Cette table est très similaire à celle des notes utilisateurs et se présente comme suit :

Tableau 22 : Table des notes concours – Coefficient de correspondance

| Table des notes concours | | | | | | |
|---|---|---|---|---|---|---|
| **Vin** | **Millésime** | **Moyenne** | **Concours 1** | **Concours 2** | **Concours 3** | **Concours N** |
| Vin 1 | 2018 | 4,16 | 4,58 | 3,9 | 4 | |
| Vin 2 | 2017 | 4,01 | 4,2 | | 3,82 | |
| Vin 3 | 2017 | 3,1 | 3,2 | 3 | | |
| Vin 1 | 2017 | | | | | |
| Vin 1 | 2016 | 4,45 | 4,9 | | 4 | |

(Edouard Diep 2019)

## 9.3 Table de correspondance

A première vue, cette table peut paraître très simple et beaucoup plus concise que les précédentes vues dans ce chapitre 9. Les apparences sont trompeuses, car c'est à cette étape que le calcul de correspondance se complique. En effet, nous allons chercher ici à calculer la note des utilisateurs pondérée ainsi que le ratio grâce aux données que nous avons précédemment récupérées de la table des notes utilisateurs (nombre de notes communes et nombre de notes similaires). Sur la page suivante, un aperçu de la table de correspondance avec la suite des explications :

Tableau 23 : Table de correspondance – Coefficient de correspondance

| Table de correspondance | | | | | |
|---|---|---|---|---|---|
| | | | | | |
| | **User1** | **User2** | **User3** | **User4** | **UserN** |
| **User 1** | 0 | 0 | 1 | 1 | |
| **User 2** | 0 | 1 | 1/2 = 0.5* | 0,5 | |
| **User 3** | 0 | 0,5 | 1 | 1 | |
| **User 4** | 1 | 0,5 | 1 | 1 | |

(Edouard Diep 2019)

La table ci-dessus compare les utilisateurs sans tenir compte des vins ou des millésimes. Chaque user est comparé à l'autre et le calcul de correspondance s'effectue comme suit, avec les données calculées précédemment (cf : Tableau 21) ainsi que l'ajout d'un nouveau code couleur pour le coefficient de correspondance* :

NbNotesSimilaires / NbNotesCommunes = CoefCorrespondance

Comme nous l'avons vu ci-avant, le User3 a bel et bien une seule note similaire et deux notes communes avec User2 ainsi qu'une seule note commune et similaire avec User4 et User1 (cf : Tableau 21).

Pour la suite du calcul, il s'agit de reprendre les données du Tableau 21 et d'en extraire la note que les autres Users ont donnée sur le vin non évalué par User3. En l'occurrence dans l'exemple du Tableau 23, nous comparons User3 avec User2. Toutefois, User4 a aussi évalué le Vin 2 – 2017. Nous devons donc prendre sa note en considération :

Tableau 24 : Extrait du Tableau 21 – Récupération des notes

| | **User1** | **User2** | **User3** | **User4** |
|---|---|---|---|---|
| Vin 1 – 2018 | | 3 | 2 | |
| Vin 2 – 2017 | 3 | 5 | | 3 |
| Vin 3 – 2017 | | 3 | 1 | 2 |
| Vin 1 – 2017 | | | 4 | |
| Vin 1 – 2016 | 4 | | | 3 |
| | | | | |

(Edouard Diep 2019)

En rouge ci-dessus sont représentées toutes les notes que les autres Users ont donné sur le vin non évalué par User3. Ainsi, nous savons que le coefficient de correspondance pour User3 et User2 est de 0.5 et que User2 a donné la note de 5 au Vin 2 – 2017. Nous pouvons alors calculer la note pondérée → CoefCorrespondance * Note = 0.5 * 5 = 2.5

Toutefois, ce n'est pas encore terminé. Comme nous le constatons dans le Tableau 24 ci-avant, le User3 a une correspondance avec User1 et User4. Il faut donc calculer les notes pondérées pour chacune de ces correspondances, toujours avec la même formule ce qui donne les résultats suivants :

User2 -> User3 = 0.5 * 5 = 2.5
User4 -> User3 = 1 * 3 = 3

Je n'ai pas tenu compte de la note de User1 pour le Vin2 – 2017 pour la simple et bonne raison qu'il n'a évalué aucun autre vin en commun avec User3. Maintenant que nous avons obtenu la note pondérée pour chacune des correspondances entre User3 et les autres Users, nous pouvons passer au sous-chapitre suivant qui nous permettra de conclure le calcul du coefficient de correspondance : la table des pronostics.

## 9.4 Table des pronostics

Nous voici à présent à la dernière étape du calcul du coefficient de correspondance. Il s'agit ici de concaténer tous les résultats obtenus et de suivre la formule présentée au début du chapitre 9, afin d'obtenir le pourcentage de correspondance du Vin 2 pour le User3. Tout d'abord, voici un aperçu de la table des pronostics avec le développement du calcul de coefficient de correspondance :

Tableau 25 : Table des pronostics – Coefficient de correspondance

| Tables des pronostics | | | | | |
|---|---|---|---|---|---|
| **Vin** | **Millésime** | **User1** | **User2** | **User3** | **User4** |
| Vin 1 | 2018 | | 3 | 2 | |
| Vin 2 | 2017 | 3 | 5 | ((0.5*5+1*3/1.5)*2+4*3+4.01*3)/(2+3+3)/5 = 78% | 3 |
| Vin 3 | 2017 | | 3 | 1 | 2 |
| Vin 1 | 2017 | 4 | 2 | 4 | |
| Vin 1 | 2016 | 4 | | | 3 |

(Edouard Diep 2019)

Pour résumer la ligne de calcul que nous voyons ci-dessus :

La partie en **vert** correspond à la note de l'utilisateur pondérée. C'est la somme de chaque note pondérée obtenue dans le sous-chapitre précédent (9.3).

La partie en **orange** représente le ratio qui n'est autre que la somme des coefficients de correspondance, également calculés dans le sous-chapitre précédent (9.3).

La partie en **rouge** est la multiplication par le nombre de notes communes entre User3 et User2.

La partie en **bleue** correspond à la note moyenne des utilisateurs obtenue pour le millésime courant multiplié par un coefficient de moyenne. Ce dernier a été décidé par le mandant du projet qui a estimé que 3 serait la bonne pondération pour une moyenne (sachant que la note minimale est 1 et la note maximale est 5).

La partie en **violet** correspond à la note moyenne des concours obtenue pour le millésime courant multiplié par un coefficient de moyenne, lui-même également décidé par le mandant du projet selon la même logique que celle citée dans le paragraphe précédent.

Enfin, la partie en **marron** correspond à la somme du nombre de notes communes et des coefficients de moyenne utilisateurs et concours.

Le tout est divisé par 5 (la note maximale) afin d'obtenir un pourcentage qui correspond au coefficient de correspondance.

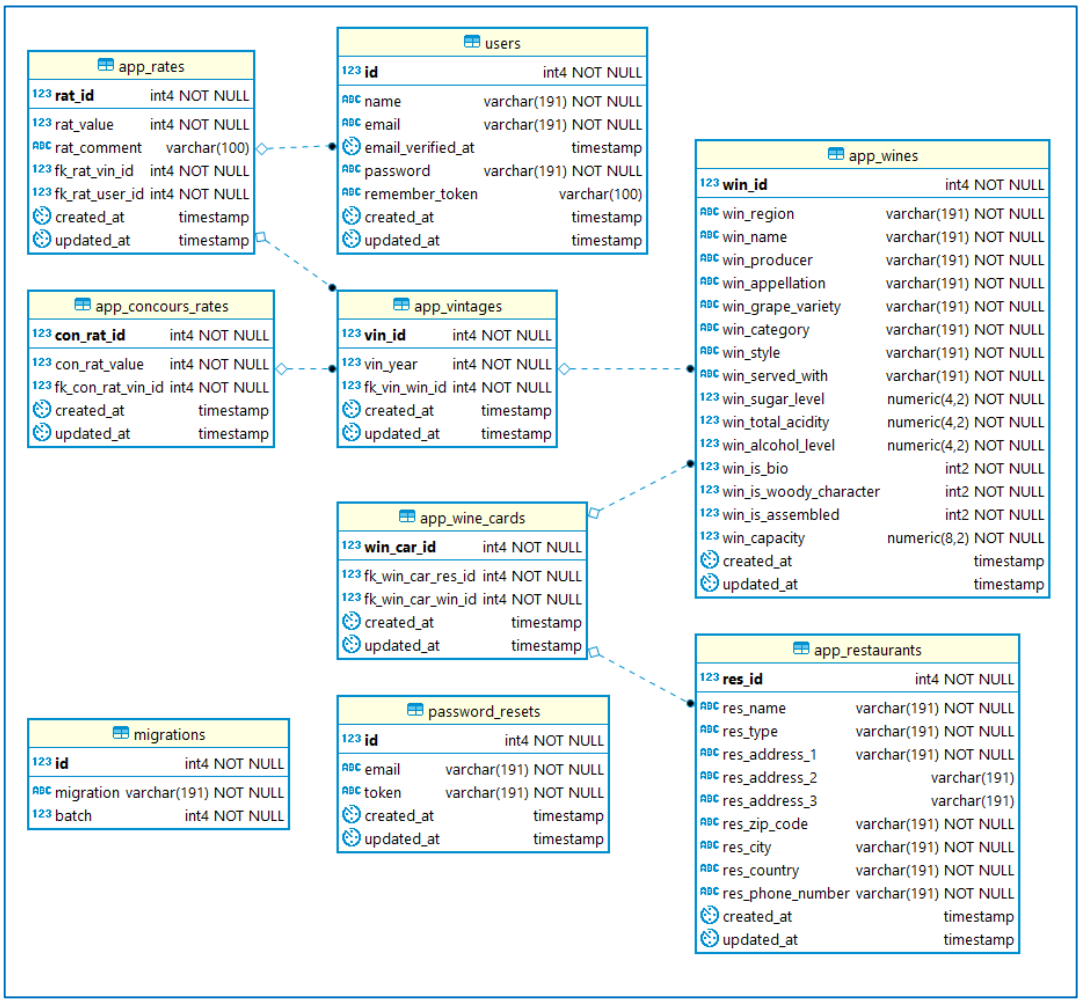
# 10. Modélisation des données

Désormais nous savons avec quels outils nous allons développer l'application ainsi que les fonctionnalités clés à implémenter tout au long du développement. Comme tout projet informatique, il est essentiel de modéliser la structure des données avec lesquelles nous travaillons, afin de savoir comment les tables sont liées entre elles, dans le but de comprendre comment les données sont structurées. Pour ce faire, j'ai utilisé une base de données PostgreSQL car l'interface m'était déjà familière.

## 10.1 Modèle UML

UML (Unified Modeling Language) définit un langage de modélisation graphique conçu pour fournir une schématisation normalisée pour visualiser la conception d'un système (Wikipédia 2019). Voici ci-dessous le modèle de données UML pour le projet VaudVin :

Figure 17 : Modèle UML de la base de données PostgreSQL




(Edouard Diep 2019)

## 10.2 Les tables

Les différentes tables visualisées dans le modèle de données (cf : Figure 17), ont été créées dans le but de stocker les données des diverses ressources nécessaires aux fonctionnalités de l'application. Chaque champ des tables a été nommé selon les conventions apprises lors des mes cours de modélisation et base de données et respecte une certaine logique qui va vous êtes décrite plus en détail dans les sous-chapitres suivants :

### 10.2.1 app_wines

Cette table contient les données liées à un vin :

Tableau 26 : Table app_wines – Base de données

| | | | |
|---|---|---|---|
| win_id | ID du vin | win_region | Région |
| win_name | Nom | win_producer | Producteur |
| win_appellation | Appellation AOC | win_grape_variety | Cépage |
| win_category | Catégorie | win_style | Style |
| win_served_with | Accords mets-vins | win_sugar_level | Taux de sucre |
| win_total_acidity | Niveau d'acidité | win_alcohol_level | Degré d'alcool |
| win_is_bio | Vin bio ou non | win_is_woody_character | Vin à caractère boisé ou non |
| win_is_assembled | Vin assemblé ou non | win_capacity | Contenance de la bouteille |

(Edouard Diep 2019)

### 10.2.2 app_restaurants

Cette table contient les données liées à un restaurant :

Tableau 27 : Table app_restaurants – Base de données

| | | | |
|---|---|---|---|
| res_id | ID du restaurant | res_name | Nom |
| res_type | Type | res_address_1 | Adresse |
| res_address_2 | Complément d'adresse | res_address_3 | Complément d'adresse 2 |
| res_zip_code | Code postal | res_city | Ville |
| res_country | Pays | res_phone_number | Téléphone |

(Edouard Diep 2019)

### 10.2.3 app_vintages

Cette table contient les données liées à un millésime ainsi qu'une clé étrangère qui fait le lien vers la table app_wines

Tableau 28 : Table app_vintages – Base de données

| | | | |
|---|---|---|---|
| vin_id | ID du millésime | vin_year | Année |
| fk_vin_win_id | ID du vin lié | | |

(Edouard Diep 2019)

### 10.2.4 users

Cette table contient les données liées à un utilisateur de l'application. Elle est générée automatiquement grâce au Framework Laravel. C'est pour cela que je n'ai pas renommé les champs de la même manière que les autres tables.

Tableau 29 : Table users – Base de données

| | | | |
|---|---|---|---|
| id | ID du user | Name | Nom |
| email | E-mail | email_verified_at | Vérification de l'adresse e-mail |
| password | Mot de passe | remember_token | Token |

(Edouard Diep 2019)

### 10.2.5 app_rates

Cette table contient les données liées aux notes des utilisateurs. Elle fait également le lien vers les tables app_vintages et users grâce à deux clés étrangères.

Tableau 30 : Table app_rates – Base de données

| | | | |
|---|---|---|---|
| rat_id | ID de la note | rat_value | Valeur |
| rat_comment | Commentaire | fk_rat_vin_id | ID du millésime lié |
| fk_rat_user_id | ID du user lié | | |

(Edouard Diep 2019)

### 10.2.6 app_concours_rates

Cette table contient les données liées aux notes des concours. Elle a un lien vers la table app_vintages avec une clé étrangère.

Tableau 31 : Table app_concours_rates – Base de données

| | | | |
|---|---|---|---|
| con_rat_id | ID de la note | con_rat_value | Valeur |
| fk_con_rat_vin_id | ID du millésime lié | | |

(Edouard Diep 2019)

### 10.2.7 app_wine_cards

Cette table contient les données liées aux cartes des vins des restaurants. Elle possède un lien vers les tables app_wines et app_restaurants grâce à deux clés étrangères.

Tableau 32 : Table app_wine_cards – Base de données

| | | | |
|---|---|---|---|
| win_car_id | ID de la carte | fk_win_car_res_id | ID du restaurant lié |
| fk_win_car_win_id | ID du vin lié | | |

(Edouard Diep 2019)

### 10.2.8 password_resets

Cette table contient toutes les données nécessaires à la fonctionnalité de réinitialisation du mot de passe (cf : chapitre 8.7). Elle est cependant générée par le Framework Laravel.

Tableau 33 : Table password_resets – Base de données

| | | | |
|---|---|---|---|
| email | E-mail de l'utilisateur | token | Token de l'utilisateur |

(Edouard Diep 2019)

### 10.2.9 migrations

Cette table est générée automatiquement en même temps que toutes les autres tables au moment de la migration Laravel. Elle contient les données de timestamps permettant de rollback des migrations qui ont été faites antérieurement. Dans le cadre du projet VaudVin, cette table ne sera pas sollicitée pour traiter des données, mais reste tout de même disponible dans la base au cas où il faudrait revenir en arrière après avoir exécuté une opération avec Laravel.

Tableau 34 : Table migrations – Base de données

| | | | |
|---|---|---|---|
| Id | ID de la migration | Migration | Nom de la migration |

(Edouard Diep 2019)

# 11. Architecture de l'application

Désormais nous connaissons la structure des données de l'application. Il est temps de voir l'architecture de l'application en elle-même, c'est-à-dire comment sont organisés les projets Front et Back end avec leurs différents packages, répertoires et fichiers permettant de structurer le code, afin de comprendre et retrouver les fonctionnalités implémentées.

Nous diviserons donc ce chapitre en deux grands sous-chapitres :

- Front end de l'application
- Back end de l'application

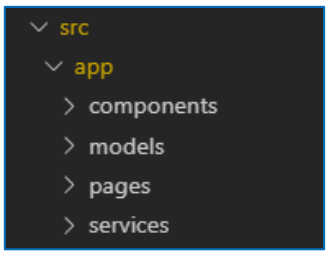
## 11.1 Front End

La partie Front end s'occupe de la couche qui concerne les interfaces d'une application, généralement codées en HTML, CSS et JavaScript, avec lesquelles interagit l'utilisateur final (Wikipédia 2019). Elle traite principalement les vues et l'affichage des données formatées. C'est la partie visible de l'iceberg. Dans les sous-chapitres suivants, je vais détailler chaque partie de l'application Front end.

### 11.1.1 Structure de base Ionic

Dans un projet créé avec Ionic, la structure des répertoires principaux de l'application, soit ceux permettant de gérer et afficher des données, est présentée comme suit :

Figure 18 : Structure de base – Ionic




(Edouard Diep 2019)

Comme nous le constatons ci-dessus, les dossiers sont au nombre de quatre : *components*, *models*, *pages* et *services*. Il existe bien d'autres fichiers de configuration dans le projet, mais dans le cadre de ce mémoire, nous nous cantonnerons aux répertoires et fichiers créés de toutes pièces. Chacun des quatre répertoires cités ci-avant sera détaillé dans les sous-chapitres suivants.

### 11.1.2 Components

Ce dossier regroupe toutes les parties de l'application qui ont la particularité d'être réutilisables sur d'autres pages ou d'autres composants. A la différence d'une page, un composant ne sera pas obligatoirement lié à une URL et pourra être appelé à différents endroits de l'application sans devoir réécrire le code et il est possible d'adapter sa logique de calcul ou ses méthodes en fonction de la page depuis laquelle il est appelé (Ionic Framework 2019). Le concept de composant permet d'éviter le code redondant.

#### 11.1.2.1 Création

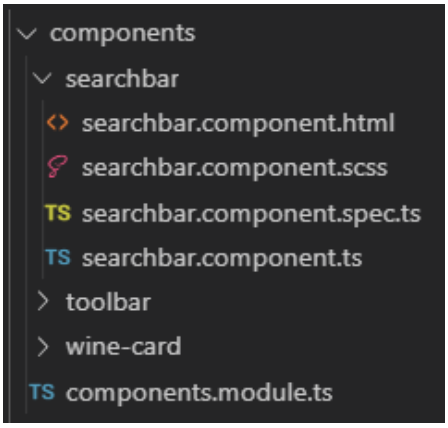
Pour générer un component sur l'application, nous pouvons exécuter la commande suivante sur le CLI (Command Line Interface) Ionic : *ionic generate component <nomComposant>*

#### 11.1.2.2 Structure

Chaque composant est constitué de quatre fichiers distincts, mais qui forment un tout lui permettant d'exister et de fonctionner à travers l'application :

- Un fichier .html servant à structurer l'affichage du composant
- Un fichier .scss servant à formater le style de l'information affichée à partir du fichier HTML
- Un fichier .spec.ts uniquement utile lors des tests unitaires
- Un fichier .ts servant à définir les attributs, les propriétés ainsi que les fonctions du composant. C'est le contrôleur du composant. Il contient la logique de calcul et utilise le langage de programmation orientée objet : TypeScript.

Figure 19 : Structure components – Ionic




(Edouard Diep 2019)

Quant au fichier *components.module.ts*, il contient la déclaration de tous les composants de l'application afin qu'ils soient visibles par les autres pages et qu'ils puissent être appelés à l'aide d'un sélecteur correspondant au nom du composant. Dans le cas de la searchbar par exemple, elle est appelée ainsi : *<app-searchbar></app-searchbar>*

#### 11.1.2.3 Implémentation

Dans le fichier *components.module.ts*, il est nécessaire d'importer, déclarer et ajouter chaque composant aux exports, afin de pouvoir les appeler dans un tag HTML d'un autre composant ou d'une autre page de l'application.

Figure 20 : Fichier components.module.ts – Ionic

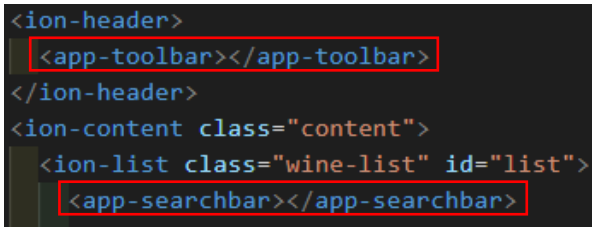
```
/*
 * @author Edouard Diep
 */
import { NgModule } from '@angular/core';
import { SearchbarComponent } from './searchbar/searchbar.component';
import { IonicModule } from '@ionic/angular';
import { WineCardComponent } from './wine-card/wine-card.component';
import { CommonModule } from '@angular/common';
import { ToolbarComponent } from './toolbar/toolbar.component';

@NgModule({
    declarations: [
        SearchbarComponent,
        WineCardComponent,
        ToolbarComponent,
    ],
    imports: [
        IonicModule,
        CommonModule,
    ],
    exports: [
        SearchbarComponent,
        WineCardComponent,
        ToolbarComponent,
    ]
})
export class ComponentsModule { }
```

(Edouard Diep 2019)

Une fois les composants déclarés dans le fichier module.ts, il suffit de les appeler par leur tag dans n'importe quelle structure HTML, comme présenté ci-dessous :

Figure 21 : Appels de composants – Ionic




(Edouard Diep 2019)

### 11.1.3 Models

Cette partie de l'application regroupe les classes correspondant aux objets métiers, c'est-à-dire ceux qui contiennent les données de notre application qui vont être envoyées à un service, qui lui transmettra ces mêmes données à la base de données afin de les stocker, les modifier ou les supprimer (cf : sous-chapitre 11.1.5 pour plus de précisions).

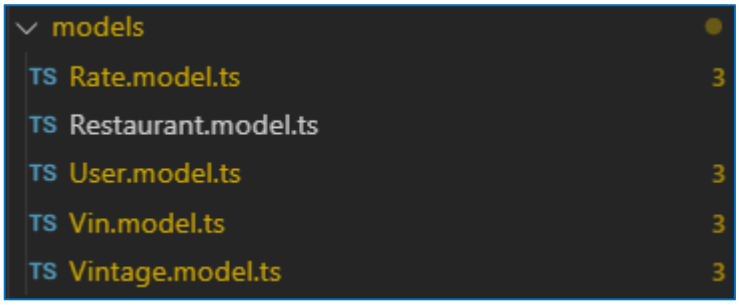
#### 11.1.3.1 Création

Pour créer un modèle dans l'application, il suffit de se placer dans le répertoire "Models" et créer un nouveau fichier du nom de l'objet que nous voulons manipuler, suivi de la dénomination "model.ts". Par exemple : *Vin.model.ts*

#### 11.1.3.2 Structure

Les modèles sont des fichiers contenant du code en langage TypeScript simplement stockés dans le répertoire "Models" de l'application. Ils sont placés les uns après les autres comme suit :

Figure 22 : Structure models – Ionic




(Edouard Diep 2019)

#### 11.1.3.3 Implémentation

Dans un modèle, nous retrouvons la déclaration de la classe du modèle ainsi que ses attributs liés. Eventuellement, nous pouvons déclarer une méthode statique qui a pour but d'instancier le modèle dans le cas où nous voulons créer un objet pour l'envoyer dans la base de données à l'aide d'un service.

Figure 23 : Classe Model – Ionic

```
 * @author Edouard Diep
 */
export class Rate {
    rat_id: number;
    rat_value: number;
    rat_comment: string;
    fk_rat_vin_id: number;
    fk_rat_user_id: number;

    /** Fonction statique permettant d'instancier un Rate */
    static createRate(value: number, comment: string, vint_id, user_id): Rate {
        const rate = new Rate();
        rate.rat_value = value;
        rate.rat_comment = comment;
        rate.fk_rat_vin_id = vint_id;
        rate.fk_rat_user_id = user_id;
        return rate;
    }
}
```

(Edouard Diep 2019)

Comme nous pouvons le voir ci-dessus, une classe Model est définie par une liste d'attributs liés à l'objet métier. Pour que cet objet soit créé et envoyé dans la base de données, il faut impérativement que les noms des attributs soient les mêmes que ceux des champs de la base de données.

La fonction createRate() (cf : Figure 23) permet d'instancier un objet de type Rate, d'assigner les valeurs reçues en paramètres et de retourner le nouvel objet créé.

### 11.1.4 Pages

Ce dossier contient et structure les différentes pages navigables de l'application. Une page est un composant Ionic avec des directives déjà fournies, prêtes à être chargées dans le système de navigation Ionic. Etant donné que les pages sont chargées dynamiquement dans l'application, elles ne nécessitent pas de sélecteur afin d'être affichées contrairement aux composants (Ionic Framework 2019).

#### 11.1.4.1 Création

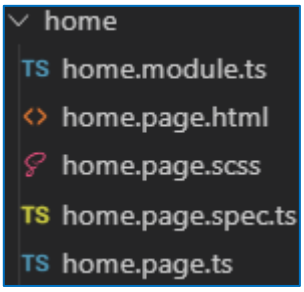
Pour générer une page sur l'application, nous pouvons exécuter la commande suivante sur le CLI (Command Line Interface) Ionic : *ionic generate page <nomPage>*

#### 11.1.4.2 Structure

Chaque page contient, à un élément près, les mêmes fichiers que les composants, soit :

- Un fichier .html servant à structurer l'affichage de la page
- Un fichier .scss servant à formater le style de l'information affichée à partir du fichier HTML
- Un fichier .spec.ts uniquement utile lors des tests unitaires
- Un fichier .ts servant à définir les attributs, les propriétés ainsi que les fonctions de la page en langage TypeScript
- Un fichier module.ts servant à déclarer les modules nécessaires à la page et permettant d'implémenter le concept de Lazy Loading du Framework Ionic dont la définition vous sera décrite dans le sous-chapitre 11.1.4.4

Figure 24 : Structure pages – Ionic




(Edouard Diep 2019)

#### 11.1.4.3 Implémentation

A l'intérieur du fichier module.ts d'une page, nous devons importer chaque module que cette dernière utilise dans son implémentation, afin de s'assurer que les librairies externes sont correctement importées à l'intérieur du projet, que les services sont injectés à travers les providers et que tout fonctionne correctement. Ci-dessous, vous est présentée une illustration du fichier module.ts pour la page "Home" :

Figure 25 : Fichier module.ts – Ionic

```
import { NgModule } from '@angular/core';
import { CommonModule } from '@angular/common';
import { FormsModule } from '@angular/forms';
import { Routes, RouterModule } from '@angular/router';

import { IonicModule } from '@ionic/angular';

import { HomePage } from './home.page';
import { UserService } from 'src/app/services/user.service';
import { ComponentsModule } from 'src/app/components/components.module';

const routes: Routes = [
  {
    path: '',
    component: HomePage
  }
];

@NgModule({
  imports: [
    CommonModule,
    FormsModule,
    IonicModule,
    ComponentsModule,
    RouterModule.forChild(routes)
  ],
  providers: [UserService],
  declarations: [HomePage]
})
export class HomePageModule {}
```

(Edouard Diep 2019)

Ci-dessus, sur la Figure 25, nous voyons les imports nécessaires pour que la page "Home" puisse afficher les informations correctement et faire ses traitements sans erreurs d'importation au niveau de la console du navigateur. De plus, le service UserService est injecté à travers les providers de ce même fichier. Nous reviendrons plus en détail sur la notion de service dans le sous-chapitre 11.1.5.

#### 11.1.4.4 Lazy Loading

La technique de Lazy Loading existe depuis la version 3 du Framework Ionic, sortie en 2018. Depuis Ionic 4, cette dernière est configurée par défaut dès le moment où l'on génère une page avec la commande dans le CLI (cf : chapitre 11.1.4.1).

Le but de Lazy Loading est d'optimiser la performance en ne chargeant que les modules nécessaires lors du démarrage de l'application, puis, le reste est importé uniquement lorsque l'utilisateur accède à la page et pas avant. En effet, le fait d'importer tout dans un seul et même fichier *app.modules.ts,* signifie que l'application va charger l'entièreté des modules dès lors qu'elle démarre, ce qui ralentit la performance puisque la page de démarrage n'a pas besoin d'utiliser tous les modules. Ainsi, en travaillant avec un fichier "module.ts" individuel pour chaque page de l'application, Lazy Loading permet à cette dernière d'optimiser considérablement la performance et le chargement des pages notamment en diminuant le nombre d'imports au démarrage de l'application (Ely Lucas, Ionic Framework 2019).

En conclusion, Lazy Loading est une bonne pratique, mais n'est pas nécessairement plus performant que son antagoniste, le Eager Loading, qui lui consiste à charger tous les modules lors du démarrage de l'application. Cela dépend de l'application créée et du nombre de pages devant être chargées au démarrage. Il faut analyser les performances pour savoir quelle technique correspond le mieux aux besoins du développeur (Ionic Framework 2019).

### 11.1.5 Services

Ce répertoire regroupe tous les éléments permettant de manipuler l'accès, la création, la modification et la suppression de données. Un service permet de faire le lien avec la base de données, envoie ou récupère des objets issus des modèles vers le Back End et retourne une réponse du côté Front (Angular.io 2019).

Par ailleurs, l'utilisation de services est un très bon moyen de partager de l'information entre plusieurs pages qui ne se connaissent pas entre elles.

#### 11.1.5.1 Création

Pour générer un service sur l'application, nous pouvons exécuter la commande suivante sur le CLI (Command Line Interface) Ionic : *ionic generate service <nomService>*

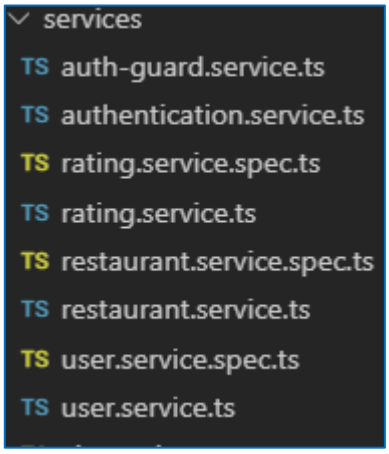
#### 11.1.5.2 Structure

Lorsqu'on génère un service à l'aide de la commande appropriée, deux fichiers sont alors créés :

- Un fichier .spec.ts uniquement utile lors des tests unitaires
- Un fichier .ts qui va contenir toute la logique et les fonctions permettant de manipuler les données, le tout écrit en langage de programmation TypeScript

Ci-dessous, vous pouvez voir une illustration de la structure des services dans le projet Ionic :

Figure 26 : Structure services – Ionic




(Edouard Diep 2019)

#### 11.1.5.3 Implémentation

L'implémentation d'un service peut varier en fonction des besoins de l'application et des objets à manipuler. Par exemple, un service peut contenir des méthodes qui vont récupérer, créer, modifier ou supprimer des données dans le Back End à l'aide de diverses méthodes provenant d'une classe HttpClient, qui elle-même doit être injectée dans le constructeur du service de la manière suivante :

Figure 27 : Injection HttpClient – Ionic

```
constructor(private http: HttpClient) { }
```

(Edouard Diep 2019)

Une fois le client injecté, il est alors possible d'appeler les méthodes "get", "post", "put" ou "delete" de la classe, afin d'effectuer les manipulations sur les données. En pratique, cela ressemble à ça :

Figure 28 : Exemple manipulation de données – Ionic

```
/** Fonction qui crée une nouvelle note dans la bdd */
public postRate(rate: Rate): Observable<any> {
  return this.http.post(URL.domaine + URL.rate.verb, rate, {
    headers: { 'Content-Type': 'application/json' }
  });
}

  /** Fonction qui met à jour une note dans la bdd */
public updateRate(rate: Rate): Observable<any> {
  return this.http.put(URL.domaine + URL.rate.verb + rate.rat_id, rate);
}

/** Fonction qui récupère toutes les notes */
public getRates(): Observable<Array<Rate>> {
  return this.http.get<Array<Rate>>(URL.domaine + URL.rate);
}
```

(Edouard Diep 2019)

Pour récapituler la Figure 28 ci-dessus, chacune des fonctions retourne un Observable, soit un objet qui contient la publication d'un message et transporte une information, mais qui n'est pas exécuté à moins qu'un client "subscribe" l'information retournée par l'Observable (Angular.io 2019). Dans le contexte de notre application, ce sont les pages et les composants qui vont potentiellement subscribe les Observable retournés par les services. Pour pouvoir accéder à ces derniers, il faut injecter les services dans les constructeurs des pages ou des composants. En pratique, l'injection d'un service ressemble à ça :

Figure 29 : Injection de services – Ionic

```
constructor( private us: UserService, private rs: RatingService,
```

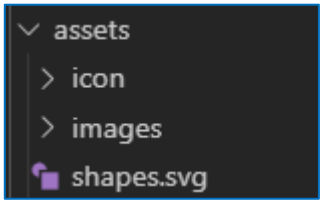
(Edouard Diep 2019)

Ce n'est qu'après avoir injecté les services qu'il est possible d'appeler les méthodes et fonctions de manipulation de données que ceux-ci possèdent dans leur implémentation.

### 11.1.6 Assets

Les assets regroupent principalement les images, logos et icônes de l'application (Ionic Framework 2019).

Figure 30 : Dossier assets – Framework Ionic




(Edouard Diep 2019)

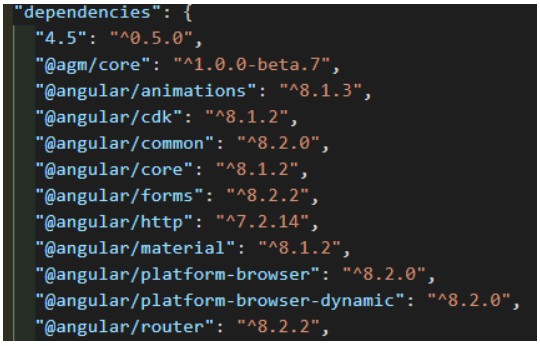
### 11.1.7 index.html

Ce fichier est généré en même temps que le projet Ionic. C'est le point d'entrée principal de l'application, la première page chargée lors du démarrage. Son but est de charger les scripts, les fichiers CSS ainsi que les plugins d'affichages tels que bootstrap. En l'absence de ce fichier, l'application ne peut simplement pas démarrer (Ionic Framework 2019).

### 11.1.8 package.json

Ce fichier liste les dépendances du projet, c'est-à-dire les librairies et packages externes dont dépend l'exécution de l'application. Il spécifie également les versions d'un package et permet de charger des dépendances automatiquement, ce qui facilite le partage de l'application avec d'autres développeurs, sur d'autres machines.

Figure 31 : Extrait du fichier package.json – Ionic




(Edouard Diep 2019)

Afin de pouvoir installer les dépendances du fichier package.json, il faut exécuter la commande suivante sur le CLI (Command Line Interface) Ionic : *npm install*

## 11.2 Back End

La partie Back end quant à elle, s'occupe de traiter l'accès et les interactions avec la base de données et traite les données brutes en arrière-plan. C'est la face cachée de l'iceberg dont l'utilisateur final ignore généralement le fonctionnement. C'est pourquoi il est utile de développer la structure pour comprendre comment marche l'application dans son ensemble. Cependant, dans le cadre de ce mémoire, nous allons prioriser la structure globale et les fichiers créés de toutes pièces.

### 11.2.1 Controllers

Nous avons vu dans le chapitre précédent, qu'il était possible d'envoyer une requête http depuis le Front, afin de manipuler les données du Back. Lorsque cette requête est reçue par Laravel, celle-ci est assignée à une route qui exécute le code en fonction de l'URL à laquelle elle est reliée. On pourrait écrire ce code dans les routes, mais cela deviendrait ingérable à partir d'un certain nombre de routes. C'est pourquoi, nous utilisons les contrôleurs qui nous permettent d'organiser les différents comportements des données en fonction de la requête http reçue du Front (Laravel 5.8 – Docs 2019). Voici l'extrait d'un contrôleur sur Laravel :

Figure 32 : Extrait de Controller – Laravel

```php
class RateController extends Controller{

    public function getRates(){
        return \DB::table('app_rates')->get(); // requête permettant de récupérer toutes les notes
    }

    public function getAverageRateByWine($user_id, $win_id){
        return \DB::table('app_rates')
        ->join('app_vintages', 'app_rates.fk_rat_vin_id', '=', 'app_vintages.vin_id')
        ->join('app_wines', 'app_vintages.fk_vin_win_id', '=', 'app_wines.win_id')
        ->where(['fk_rat_user_id' => $user_id, 'app_wines.win_id' => $win_id])
        ->avg('rat_value');
    }
```

(Edouard Diep 2019)

### 11.2.2 Models

Les modèles du point de vue Laravel, sont des représentations de la base de données sous forme d'objets. Ils permettent entre autres, de simplifier les requêtes POST notamment avec Eloquent qui se sert de modèles, afin de créer la ressource à envoyer sur la base de données. Pour générer un modèle sur Laravel, il faut taper la commande suivante dans le CLI : *php artisan make:model <nomModel>*

Exemple d'un modèle de données :

Figure 33 : Exemple de Model – Laravel

```php
class Rate extends Model
{

    protected $table = 'app_rates';
    protected $guarded = ['rat_id'];
    protected $primaryKey = 'rat_id';
    protected $fillable = [
        'rat_value',
        'rat_comment',
        'fk_rat_vin_id',
        'fk_rat_user_id',
    ];
}
```

(Edouard Diep 2019)

### 11.2.3 Migrations

Les migrations sont en quelque sorte les scripts exécutés à l'aide du CLI Laravel et qui nous permettent de générer les tables dans la base de données liée au projet. Les noms des fichiers de migrations sur Laravel ont la particularité d'être horodatés, c'est-à-dire que la date et l'heure de la migration sont inscrites dans le nom des fichiers. Ceci est utile lors d'une opération rollback (retour en arrière) par exemple.

Pour générer une migration sur Laravel, il faut taper la commande suivante dans le CLI : *php artisan make:migration <nomMigration>*

Ci-dessous un extrait d'une migration Laravel que j'ai généré :

Figure 34 : Extrait de Migration – Laravel

```php
class CreateUsersTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('users', function (Blueprint $table) {
            $table->increments('id');
            $table->string('name');
            $table->string('email')->unique();
            $table->timestamp('email_verified_at')->nullable();
            $table->string('password');
            $table->rememberToken();
            $table->timestamps();
        });
    }
```

(Edouard Diep 2019)

### 11.2.4 Routes

Toutes les routes Laravel sont définies dans un seul et même répertoire du nom de "routes" et sont chargées automatiquement par le Framework. Le fichier "web.php" définit les routes de l'interface web qui sont accessibles par l'URL spécifié pour ces routes, à travers le navigateur. Le fichier "api.php" contient des routes, dont l'état n'est pas stocké côté serveur (Laravel 5.8 – Documentation, 2019). Cela signifie qu'elles sont inaccessibles à travers le navigateur et traitent des opérations particulières comme l'enregistrement, le login ou la réinitialisation du mot de passe. Dans le cadre du projet VaudVin, la grande majorité des routes a été définie dans le fichier "web.php", dont un extrait vous est présenté ci-dessous :

Figure 35 : Extrait du fichier web.php – Laravel

```
Route::get('/users', 'UserController@getAllUsers');
Route::get('/users/{user_id}', 'UserController@getUserById');
Route::get('/users/{user_id}/rates/distinct', 'RateController@getUserDistinctRates');
Route::get('/users/{user_id}/rates', 'RateController@getUserRates');
Route::get('/users/{user_id}/vintages/{vin_id}', 'RateController@getUserRatesByVintages');
Route::get('/users/{user_id}/wines/{win_id}', 'RateController@getUserRatesByWines');
Route::get('/users/{user_id}/wines/{win_id}/rates/avg', 'RateController@getAverageRateByWine');
Route::get('/user_id', 'UserController@getAuthenticatedUserId');

/** ROUTES POST */
Route::post('rates', 'RateController@postRate');
Route::post('vintages', 'VintageController@postVintage');

/** ROUTES PUT */
Route::put('rates/{rat_id}', 'RateController@updateRate');

/** ROUTES POUR AUTHENTICATIONS*/
Auth::routes();
```

(Edouard Diep 2019)

#### 11.2.4.1 Les routes du projet

Tableau 35 : Liste des routes Back end de l'application

| N° | Méthode | URL | Paramètres | Code | Description |
|---|---|---|---|---|---|
| 1 | GET | /wines | { - win_id } | 200 | Récupère tous les vins de la base de données ou les informations d'un seul vin avec son ID. |
| 2 | GET | /wines/rates | { win_id } | 200 | Récupère toutes les notes liées à un vin. |
| 3 | GET | /wines/vintages | { win_id } | 200 | Récupère tous les millésimes d'un vin. |

| | | | | | |
|---|---|---|---|---|---|
| 4 | GET | /wines/concoursrates | { win_id } | 200 | Récupère toutes les notes concours liées à un vin. |
| 5 | GET | /restaurants | { - res_id } | 200 | Récupère tous les restaurants de la base de données ou les informations d'un seul restaurant avec son ID. |
| 6 | GET | /restaurants/wines | { res_id } | 200 | Récupère tous les vins liés à un restaurant avec son ID. |
| 7 | GET | /vintages | { - } | 200 | Récupère tous les millésimes de la base de données. |
| 8 | GET | /users | { - user_id } | 200 | Récupère tous les users de la base de données ou les informations d'un seul user avec son ID. |
| 9 | GET | /users/rates | { user_id } | 200 | Récupère toutes les notes liées à un seul user avec son ID. |
| 10 | GET | /users/vintages | { user_id } | 200 | Récupère tous les millésimes liés à un seul user avec son ID. |
| 11 | GET | /users/wines | { user_id win_id } | 200 | Récupère les données d'un vin pour un seul utilisateur lié à son ID. |
| 12 | POST | /rates | - | 201 | Crée une ressource contenant une note. |
| 13 | POST | /vintages | - | 201 | Crée une ressource contenant un millésime. |
| 14 | PUT | /rates | { rat_id } | 200 | Modifie la représentation d'une note avec son ID. |

(Edouard Diep 2019)

### 11.2.5 Views

Les vues dans Laravel contiennent le code HTML et Blade (moteur de template intégré dans le Framework) interprété par l'application et qui sépare la logique de l'application de la présentation. Les vues sont stockées dans le répertoire "views" (Laravel 5.8 – Docs 2019). Dans le cadre du projet VaudVin, je n'ai pas eu à me servir du plein potentiel des vues étant donné que la seule utilité de Laravel était pour moi l'API Back End, malgré la gestion Full Stack du Framework (c'est-à-dire qu'il peut gérer à la fois le Back et le Front end d'une application).

Toutefois, j'ai dû créer une vue pour l'e-mail de réinitialisation du mot de passe envoyé à l'utilisateur lors de sa requête du côté Front.

Exemple de vue Laravel :

Figure 36 : Extrait d'une vue – Laravel

```
@component('mail::message')
# Demande de réinitialisation

Bonjour,

Pour réinitialiser votre mot de passe VaudVin, veuillez cliquer sur le lien ci-dessous.
Si vous n''avez pas demandé de nouveau mot de passe, veuillez ignorer cet e-mail.

@component('mail::button', ['url' => 'http://localhost:8100/reset-password-form?
token='.$token])
Réinitialiser le mot de passe
@endcomponent

Merci,<br>
Le développeur de VaudVin.

Ce message a été envoyé automatiquement. Veuillez ne pas répondre à cette adresse.

@endcomponent
```

(Edouard Diep 2019)

### 11.2.6 .env

Ce fichier a la particularité d'être à la racine du projet Laravel, lorsque ce dernier a été généré. Il contient toute la configuration de l'environnement de l'application. Les paramètres de configuration sont définis sous forme de variables d'environnement, avec des valeurs qui leur sont assignées en clair. Dans le cadre de VaudVin, je me suis servi de ce fichier pour indiquer à Laravel avec quelle base de données je voulais travailler et à travers quel serveur mail je voulais communiquer, ainsi que les informations de connexion, l'encryption désirée, l'adresse d'expéditeur et le nom, afin de mettre en place la base de données PostgreSQL et la fonctionnalité de réinitialisation du mot de passe.

Figure 37 : Extrait du fichier .env – Laravel

```
DB_CONNECTION=pgsql
DB_HOST=127.0.0.1
DB_PORT=5432
DB_DATABASE=vaudvin
DB_USERNAME=postgres
DB_PASSWORD=admin
```

```
MAIL_DRIVER=smtp
MAIL_HOST=smtp.googlemail.com
MAIL_PORT=465
MAIL_USERNAME=vaudvin.no.reply@gmail.com
MAIL_PASSWORD=vaudvin3152
MAIL_ENCRYPTION=ssl
MAIL_FROM_ADDRESS=no-reply@vaudvin.ch
MAIL_FROM_NAME=VaudVin
```

(Edouard Diep 2019)

# 12. Interface de l’application

Nous avons jusqu’ici vu les technologies, techniques et architectures utilisées pour développer le projet. À présent, il est temps de passer à l’aspect visuel : les différentes interfaces de l’application VaudVin.

Tout d’abord, il faut savoir que l’utilisateur peut en tout temps naviguer d’une page à l’autre de l’application, en touchant l’icône du menu latéral situé en haut à gauche de l’interface sur la grande majorité des pages.

## 12.1 Les composants

Pour ce projet, j’ai développé trois composants, afin de pouvoir les réutiliser dans plusieurs pages sans devoir réécrire le code de leur implémentation. Cela m’a permis d’économiser beaucoup de temps et de ressource, notamment en termes de performance. Les trois composants sont les suivants :

- Searchbar
- Toolbar
- Winecard

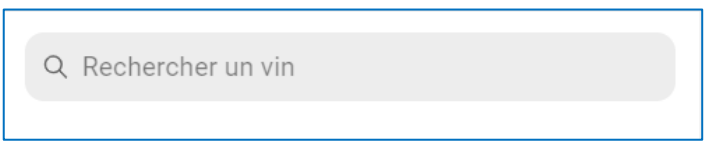
### 12.1.1 Searchbar

Ce composant utilise le style du composant natif d’Ionic, nommé : "*ion-searchbar*" (Ionic Framework 2019). C’est une barre de recherche dynamique permettant à l’utilisateur de pouvoir rechercher un vin et met à jour l’affichage de la liste des vins se trouvant en dessous, à chaque frappe au clavier, à l’aide d’un évènement de type "*keydown"* lié à la saisie de l’utilisateur. Ce composant est utilisé sur les pages suivantes :

- Liste des vins
- Historique personnel
- Carte des vins

Voici un aperçu de la searchbar :

Figure 38 : Composant Searchbar – Interface Front end




(Edouard Diep 2019)

### 12.1.2 Toolbar

Ce composant utilise le style du composant natif d'Ionic, nommé : "*ion-toolbar*" (Ionic Framework 2019). C'est une barre d'outils venant se placer tout en haut de l'écran sur la majorité des pages de l'application et permet à l'utilisateur de pouvoir interagir avec un menu latéral lui permettant de naviguer à travers l'application. La toolbar affiche également le nom de la page courante sur laquelle se trouve l'utilisateur ainsi que le logo VaudVin. Elle est utilisée sur les pages suivantes :

- Accueil
- Liste des vins
- Trouver un restaurant
- Aide

Voici un aperçu de la toolbar :

Figure 39 : Composant Toolbar – Interface Front end

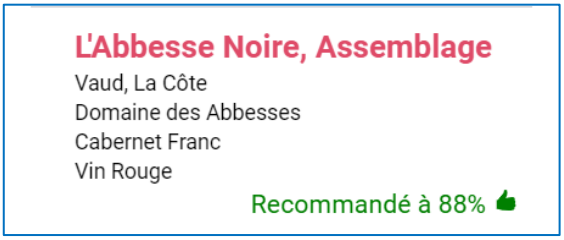

(Edouard Diep 2019)

### 12.1.3 Winecard

Ce composant utilise le style du composant natif d'Ionic nommé : "*ion-item*" (Ionic Framework 2019). C'est une carte qui affiche plusieurs informations sur un vin notamment son nom, sa région, son producteur, son cépage ainsi que son type.

La winecard affiche également la note du vin calculée sur la moyenne des millésimes (dans le cas où l'utilisateur a déjà noté ce vin) ou, dans le cas inverse, le coefficient de correspondance calculé avec un label : "*Recommandé à X%*", X étant le coefficient en question. Voici un exemple de winecard avec le label de recommandation :

Figure 40 : Composant Winecard – Interface Front end




(Edouard Diep 2019)

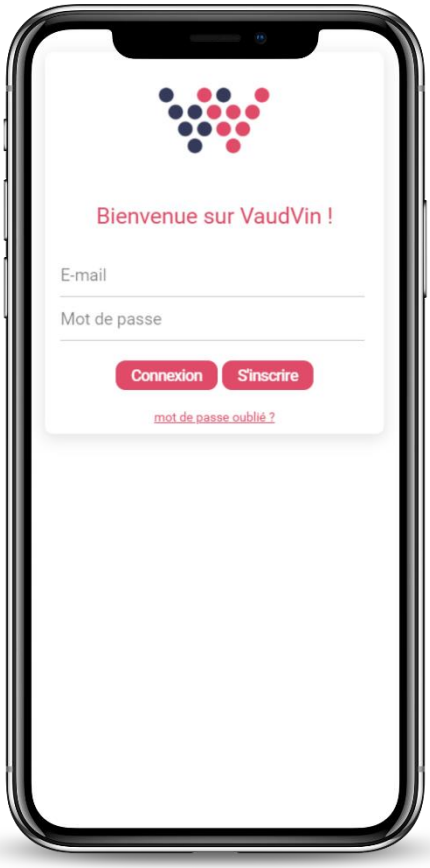
## 12.2 Les pages

Pas moins de douze pages ont été développées dans le cadre du projet VaudVin. Chacune de ces pages est soit accessible via le menu latéral généré par le projet à l'aide du starter template nommé : "*sidemenu*" (Ionic Framework 2019), soit elle est disposée à suivre un cheminement précis dans l'application. Je détaillerai tout cela dans les sous-chapitres qui suivent.

### 12.2.1 Login

Cette page est la première page visible au démarrage de l'application. Elle permet à l'invité de s'authentifier avec son adresse e-mail et son mot de passe s'il a déjà un compte, ou d'en créer un si c'est la première fois qu'il se connecte à l'aide du bouton : "s'inscrire".

Figure 41 : Fenêtre login – Interface Front end

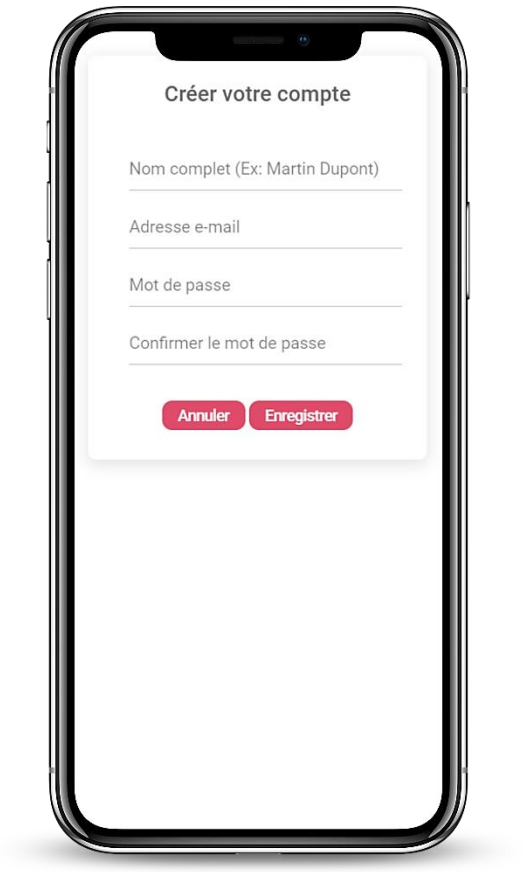



(Edouard Diep 2019)

### 12.2.2 Création de compte

L'utilisateur peut s'authentifier grâce à un formulaire avec son nom complet, son adresse e-mail, son mot de passe et un champ pour confirmer le mot de passe. Une validation est faite sur tous les champs, afin d'assurer que l'invité ne saisisse pas n'importe quoi. Cette page est accessible uniquement depuis la fenêtre login.

Figure 42 : Fenêtre Création de compte – Interface Front end




(Edouard Diep 2019)

### 12.2.3 Accueil

C'est la page que l'utilisateur voit dès qu'il s'est connecté à l'application. Pas de fonctionnalité implémentée ici, mais uniquement un message saluant l'utilisateur et le guidant vers le menu latéral, afin de commencer à naviguer sur l'application (cf : Figure 43 page suivante).

Figure 43 : Fenêtre Accueil – Interface Front end


(Edouard Diep 2019)

### 12.2.4 Liste des vins

Sur cette page l'utilisateur consulte la liste exhaustive des vins stockés dans la base de données de l'application. Le composant Searchbar permet de rechercher un vin dans la liste et met à jour les résultats de recherche à chaque frappe de l'utilisateur. Le vin recherché s'affiche ou, dans le cas d'une recherche erronée, un message prévient l'utilisateur qu'aucun vin ne correspond à sa recherche.

Le composant Winecard quant à lui, est appelé dans une boucle qui parcourt la liste de tous les vins de la base de données, qu'ils aient été notés ou non par l'utilisateur qui consulte cette page. Dans le cas où un vin n'a pas été noté, un pourcentage de recommandation est affiché (cf : chapitre 9). Dans le cas inverse, la note du vin calculée sur la moyenne des notes du millésime s'affiche (cf : Figure 44 page suivante). L'utilisateur peut toucher un vin pour sélectionner son millésime (cf : chapitre 12.2.5).

Figure 44 : Fenêtre Liste des vins – Interface Front end


(Edouard Diep 2019)

### 12.2.5 Sélection du millésime

Cette page est assez simpliste visuellement et permet uniquement à l'utilisateur de pouvoir sélectionner le millésime du vin sur lequel il a cliqué avant d'arriver sur cette page (cf : Figure 45 page suivante). Une fois le millésime sélectionné dans la liste déroulante, l'utilisateur clique sur : "Valider" et à ce moment-là, deux actions peuvent s'effectuer :

- Soit le millésime a déjà été noté et la note est récupérée dans la base de données
- Soit le millésime n'a jamais été noté et il est créé et stocké dans la base de données

Dans les deux cas, l'utilisateur est ensuite redirigé vers la page : "Fiche du vin" (cf : chapitre 12.2.6).

Figure 45 : Fenêtre Sélection du millésime – Interface Front end


(Edouard Diep 2019)

### 12.2.6 Fiche du vin

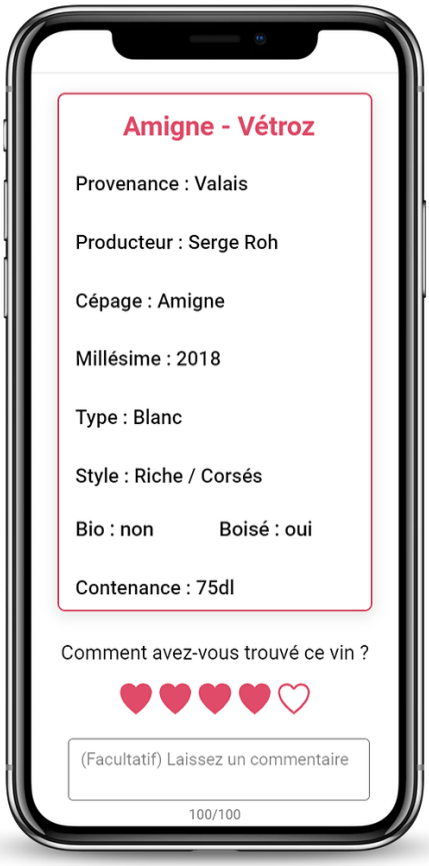
Cette page défilante présente à l'utilisateur la fiche détaillée du vin et du millésime qu'il a sélectionné auparavant (cf : chapitre 12.2.5). Hormis la fiche, elle permet surtout de donner son avis sur un vin à l'aide d'une notation sous forme de cœurs. L'utilisateur a également la possibilité de laisser un commentaire, facultatif, d'un maximum de 100 caractères sur le vin qu'il a dégusté (cf : Figure 46 page suivante).

Lorsqu'il a enregistré son avis, l'utilisateur a alors deux choix qui s'offrent à lui :

- Soit il retourne sur la page : "Liste des vins" et continue de noter des vins
- Soit il accède à la page "Historique personnel" et consulte son historique

Dans les deux cas, la note et/ou le commentaire saisis par l'utilisateur sont enregistrés dans la base de données.

Figure 46 : Fenêtre Fiche du vin – Interface Front end

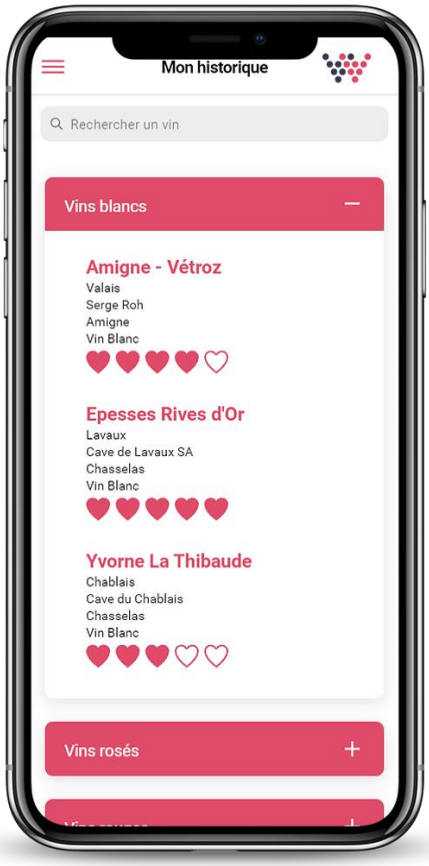



(Edouard Diep 2019)

### 12.2.7 Historique personnel

Cette page liste les vins notés par l'utilisateur. Tous les vins sont triés par catégorie (rouges, blancs, rosés, mousseux et spécialités). Lorsque l'utilisateur clique sur une catégorie, cette dernière s'étend avec une petite animation et laisse place aux vins avec leurs notes exprimées sous forme de cœurs (cf : Figure 47 page suivante). Les notes affichées sont constituées de la moyenne des notes des millésimes notés pour le vin courant. L'utilisateur peut rechercher un vin à l'aide du composant Searchbar présent en haut de la page. Si l'utilisateur clique sur ce dernier, il accède à la page : " Sélection du millésime" (cf : chapitre 12.2.5).

Figure 47 : Fenêtre Historique personnel – Interface Front end

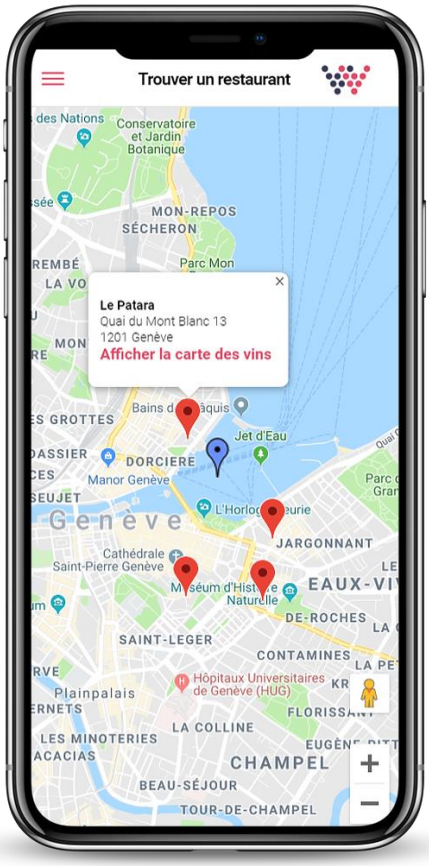



(Edouard Diep 2019)

### 12.2.8 Trouver un restaurant

Cette page implémente l'API Google Maps. Elle propose à l'utilisateur une carte avec des marqueurs rouges représentant les différents restaurants stockés dans la base de données, tandis que la position de l'utilisateur courant, est marquée en bleu. L'utilisateur a alors le choix de toucher un des marqueurs rouges pour ainsi afficher une infobulle avec les détails du restaurant, ainsi qu'un lien permettant d'afficher la carte des vins du restaurant en question (cf : Figure 48 page suivante). Si l'utilisateur clique sur ce lien, il est redirigé vers la page : "Carte des vins".

Figure 48 : Fenêtre Trouver un restaurant – Interface Front end

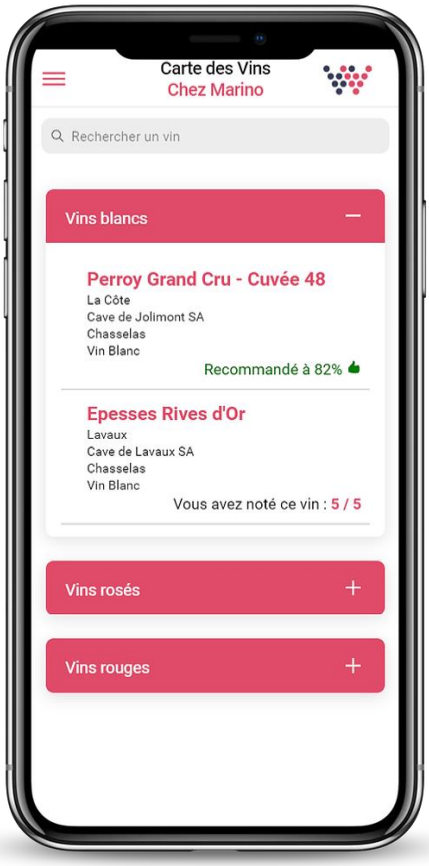



(Edouard Diep 2019)

### 12.2.9 Carte des vins

Cette page est très similaire à la page : "Historique personnel" (cf : chapitre 12.2.7). En effet, les vins sont triés par catégorie et affichés sous la forme d'accordion que l'utilisateur peut étendre lorsqu'il clique sur une catégorie. Ici, les notes des vins sont affichées en chiffres et non en cœurs et dans le cas où l'utilisateur qui consulte cette page n'a pas encore noté un vin, le label de recommandation vient s'afficher sur la carte (cf : Figure 49 page suivante). Tout comme les pages "Liste des vins" et "Historique personnel", l'utilisateur peut rechercher un vin à l'aide du composant Searchbar situé en haut de la page. Si ce dernier clique sur un vin, il est redirigé vers la page : "Sélection du millésime" pour le vin choisi.

Figure 49 : Fenêtre Carte des vins – Interface Front end



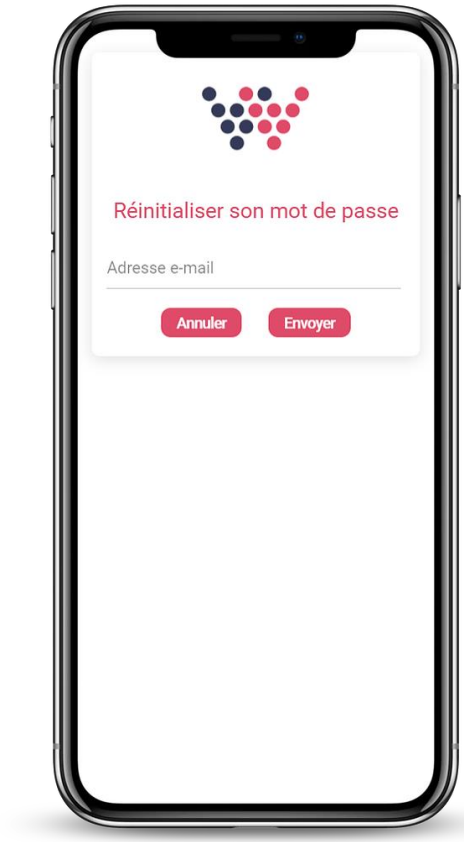

(Edouard Diep 2019)

### 12.2.10 Réinitialiser son mot de passe

Cette page permet à l'utilisateur de pouvoir changer son mot de passe dans le cas où il l'aurait oublié. Elle est accessible uniquement depuis la page : "Login" à l'aide du lien cliquable : "Mot de passe oublié ?". Sur cette page, un champ permet à l'invité de saisir son adresse e-mail (cf : Figure 50 page suivante). Une fois validé, le système envoie un e-mail de réinitialisation du mot de passe à l'utilisateur, qui doit consulter sa boîte de messagerie.

Une fois à l'intérieur de l'e-mail (cf : Figure 52 page suivante), ce dernier clique sur le bouton : "Réinitialiser votre mot de passe" et il est redirigé vers un formulaire l'invitant à saisir son adresse e-mail, un nouveau mot de passe et la confirmation de ce dernier (cf : Figure 51 page suivante).

Figure 50 : Fenêtre Réinitialiser son mot de passe – Interface Front end

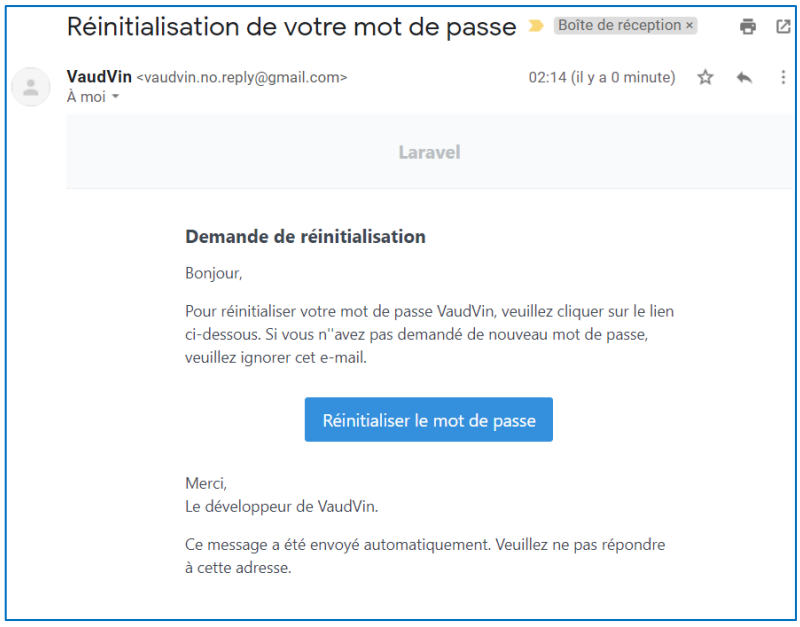



(Edouard Diep 2019)

Figure 51 : Formulaire de réinitialisation du mot de passe


(Edouard Diep 2019)

Figure 52 : E-mail de réinitialisation du mot de passe

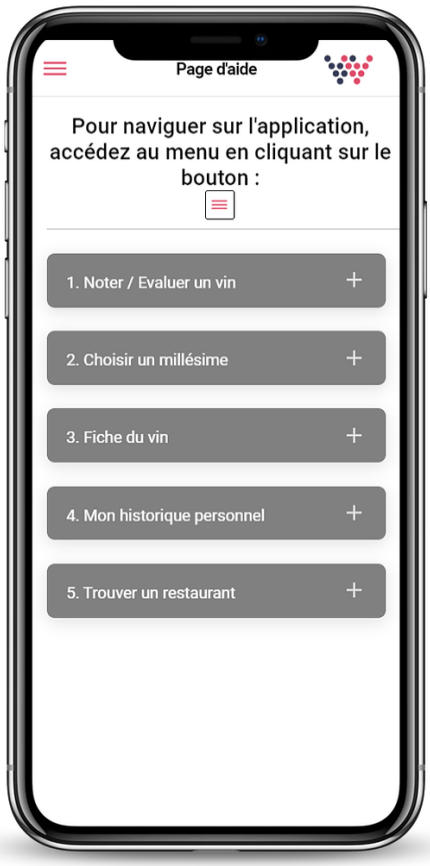



(Edouard Diep 2019)

### 12.2.11 Aide

A la demande du mandant du projet et en vue de la complexité des fonctionnalités implémentées dans l'application, il a été convenu de mettre en place une rubrique d'aide, afin d'assister l'utilisateur connecté en cas où l'utilisation de l'interface se verrait confuse ou difficile à comprendre. La page : "Aide" regroupe plusieurs chapitres, un par fonctionnalité, affichés sous la forme de composants accordions à dérouler (cf : Figure 53 ci-dessous). L'utilisateur n'a juste qu'à toucher une rubrique pour qu'elle s'étende avec une petite animation et laisse place à la marche à suivre.

Figure 53 : Page Aide – Interface Front end




(Edouard Diep 2019)

# 13. Difficultés rencontrées

Comme dans quelconque projet pratique, tout ne se déroule pas forcément comme prévu. Cela a malheureusement été le cas avec plusieurs aspects techniques dont je ne maîtrisais pas ou très peu le fonctionnement et je ne m'en suis rendu compte que lorsque j'étais en train de construire l'application. Par ailleurs, le délai relativement court alloué au développement du projet m'a limité en termes d'apprentissage. Cela a été pour moi une contrainte difficile et stressante, sachant que mon client attendait un prototype fonctionnel avec des fonctionnalités implémentées à sa demande. Je n'ai donc pas pu lui fournir tout ce qu'il attendait comme prévu.

## 13.1 Coefficient de correspondance

Sans grande surprise, l'aspect mathématique du projet m'a donné du fil à retordre dès lors qu'il s'agissait de lier le calcul théorique avec le code de l'application. Moi-même n'ayant jamais été un mordu des chiffres et des calculs, j'étais convaincu que le fait d'aller chercher les données nécessaires dans les tables de la base de données, afin de calculer le coefficient de correspondance, était une demande simple et réalisable. J'ai été très surpris de découvrir la complexité du problème, notamment en devant imbriquer des requêtes asynchrones et parcourir des structures pour comparer tous les utilisateurs les uns avec les autres, afin de récupérer les bonnes notes pour X vin et procéder au calcul à l'intérieur de boucles de parcours. Ma logique selon moi était bonne, mais malheureusement l'application ralentissait énormément et la performance en a pris un sacré coup. Je n'ai pas été capable de résoudre le problème à temps pour un délivrable fonctionnel et me suis contenté de retourner temporairement un pourcentage généré aléatoirement avec une librairie.

Solution : Stocker des calculs dans la base de données était inenvisageable car cela ne respectait pas les conventions apprises lors de ma formation scolaire, bien que cela m'aurait permis d'améliorer la performance de l'application. J'ai alors pensé à créer une table supplémentaire "app_correspondances", dans laquelle je stockerais deux notes, deux utilisateurs et un champ qui m'indique si les notes sont similaires ou non (cf : chapitre 9). Ainsi, je n'aurais qu'à récupérer les correspondances entre un utilisateur et l'autre et réduirais les appels vers le Back ainsi que les boucles de parcours, ce qui optimiserait la performance de manière générale. Cette solution serait envisageable à l'avenir.

## 13.2 API Google Maps

Il m'a fallu un certain temps avant de pouvoir utiliser l'API Google Maps et faire les tests fonctionnels, car j'ai dû patienter sur le mandant du projet qui devait rentrer ses coordonnées bancaires, afin de m'envoyer la clé API me permettant d'afficher la carte et ainsi pouvoir faire mes tests, ce qui a retardé le développement de la fonctionnalité de recherche des restaurants.

A la base, le mandant voulait une carte dynamique avec des marqueurs placés sur tous les restaurants de la Suisse dans un rayon de 500 mètres à proximité de la position de l'utilisateur connecté. Je n'avais aucune idée de comment mettre en place une telle fonctionnalité sans devoir stocker l'entièreté des restaurants du pays dans une base de données, ce qui est considérablement lourd en termes de ressources. Par conséquent, je me suis contenté de faire simple afin d'optimiser la performance de l'application : mettre en place des marqueurs sur les quelques restaurants stockés en base de données, afin d'afficher leur carte des vins, elle-même aussi stockée localement.

J'ai également tenté d'implémenter la fonctionnalité Autocomplete de l'API Google Maps, afin de pouvoir retrouver un lieu en saisissant l'adresse dans la barre de recherche, mais j'ai eu quelques problèmes d'incompatibilité avec les versions des dépendances du projet qui m'ont forcé à renoncer. De plus, je ne savais pas comment restreindre les lieux dans la barre de recherche pour n'afficher que les adresses des restaurants lors de la saisie Autocomplete.

Solution : En fin de compte, je n'ai eu qu'une courte semaine pour mettre en place la fonctionnalité des marqueurs sur la carte Google Maps. Avec la contrainte du temps et la rédaction du mémoire en parallèle, je n'ai pas pu me focaliser dessus à 100%. Il m'aurait fallu davantage de temps pour apprendre et comprendre les diverses possibilités qu'offre l'API, afin de l'utiliser à son plein potentiel. En ce qui concerne la fonctionnalité du radius de 500 mètres, l'API propose une librairie nommée "Geometry" qui permet, à l'aide d'une méthode "computeDistanceBetween()" de calculer la distance entre un marqueur et des adresses placées à proximité. Le but étant de montrer les marqueurs qui se trouvent à moins d'une certaine distance de la position de l'utilisateur et de cacher les autres. Cette fonctionnalité sera très probablement mise en place dans une future version de l'application.

# 14. Perspectives d'avenir

Le projet désormais abouti, il est intéressant d'analyser ce qui peut être actuellement amélioré et ce qui va venir se rajouter à l'avenir : les nouvelles fonctionnalités, mais également les futurs besoins du mandant vis-à-vis de l'application.

## 14.1 Améliorations de l'existant

### 14.1.1 Photos des vins

Il a été conclu entre le mandant et moi-même que les photos des vins ne seront pas stockées ni affichées, ceci pour deux raisons : la première est qu'il ne garantit pas de pouvoir trouver des images de bonne qualité pour chacune des bouteilles et chacun des millésimes stockés en base de données. Deuxièmement, selon le mandant, les photos inciteraient les utilisateurs de l'application à vouloir acheter le vin visualisé à travers la plateforme, ce qui place alors sur le second plan l'objectif principal du projet : celui de pouvoir évaluer et recommander un vin.

A l'avenir, il serait tout-de-même intéressant pour l'expérience utilisateur, de pouvoir visualiser le vin, afin de donner envie et par conséquent optimiser la rétention client. Le fait de n'avoir qu'un label avec un nom n'est pas assez "tape à l'œil" et peut potentiellement faire fuir le consommateur qui trouvera l'application trop simpliste et sans intérêt par rapport à d'autres concurrents plus beaux visuellement.

### 14.1.2 Recherche de restaurants

La fonctionnalité de Places API permettant de placer les marqueurs sur la carte Google Maps, afin de retrouver leur carte des vins est certes très pratique, mais oblige l'utilisateur à devoir consulter les restaurants stockés dans la base de données locale. Il serait envisageable de pouvoir rechercher n'importe quel restaurant stocké sur le serveur Google Maps, afin de pouvoir retrouver les adresses. Le fait est qu'il faut pouvoir accéder à la carte des vins pour chacun de ces restaurants, ce qui nécessite forcément de devoir stocker des milliers de vins et des milliers d'entrée dans la base de données pour chacun des restaurants. Malgré que cela soit relativement lourd en termes de performance, nous n'excluons pas la possibilité de pouvoir le faire.

### 14.1.3 Filtres de recherche

Le composant Searchbar fonctionne actuellement et effectue une recherche sur tous les champs d'un vin, peu importe ce que l'on saisit dans la barre de recherche. C'est certes

dynamique, mais pas très intuitif pour l'utilisateur, qui ne peut pas forcément filtrer sa recherche. C'est pourquoi, le mandant et moi-même avons discuté de la possibilité de pouvoir restreindre l'affichage des vins par catégorie en fonction du style ("puissant", "léger", "fruité & gouleyant", etc.) ou de la région ("Vaud", "Valais", etc.). Le but étant de guider l'utilisateur vers une recherche approfondie, sans forcément connaître le style ou la catégorie d'un vin qu'il ne peut saisir de lui-même dans la barre de recherche.

## 14.2 Les futures fonctionnalités

### 14.2.1 Scanner d'étiquettes

Afin de se démarquer de la concurrence rude sur le marché des applications mobiles, il serait judicieux d'implémenter une fonction permettant de reconnaître les données d'un vin à travers son étiquette, en utilisant l'appareil photo du Smartphone sur lequel est installée l'application. La mise en place d'une telle fonctionnalité requiert des coûts supplémentaires pour se procurer un moteur externe déjà tout conçu comme Vuforia, le service de reconnaissance Cloud utilisé par Vivino, le plus gros concurrent du marché (Vuforia engine 2019).

### 14.2.2 Partage de vin sur les réseaux sociaux

Une fois le vin évalué avec une note et/ou un commentaire, la possibilité de partager son évaluation à travers WhatsApp, Instagram ou même par e-mail serait la bienvenue, afin de maximiser la rétention client, tout en proposant à l'utilisateur d'envoyer à ses proches sa note avec son commentaire.

### 14.2.3 Commande de vin directement chez le producteur

VaudVin n'étant pas une plateforme d'achat, mais pourrait le devenir avec la possibilité de contacter le producteur d'un vin qui vient d'être évalué et le commander chez lui, puis d'obtenir la localisation du vigneron, afin de pouvoir aller récupérer la bouteille en personne.

### 14.2.4 Accès à des statistiques sur les évaluations

Les producteurs pourraient avoir accès à un certain nombre d'informations, comme des statistiques sur les évaluations de leurs vins. Cela viserait un public plus large, puisque les vignerons eux-mêmes viendraient sur l'application afin de savoir lesquels de leurs vins peuvent potentiellement se vendre le mieux, selon les goûts des consommateurs.

# 15. Conclusion

Ce projet pratique et concret d'assez grande envergure m'a permis d'ouvrir les yeux sur une facette essentielle du métier qui m'attend à la suite de mes études. Il m'a également permis de mettre en pratique mes connaissances scolaires fraichement acquises et je ne pouvais pas rêver mieux avant de me lancer dans le monde professionnel où un nombre inconsidérable de projets similaires à celui-ci m'attendent très certainement.

Seul sur la partie développement de ce projet, j'ai pris conscience de l'importance de plusieurs détails, dont je ne m'étais pas focalisé lors de travaux en groupe durant mon cursus scolaire. Notamment le fait d'établir en début de projet une documentation minimale, mais nécessaire, m'a permis de structurer l'avancement de manière générale. Néanmoins, je regrette de ne pas avoir établi de planification globale que j'ai jugée secondaire sur le plan développement, d'autant plus que j'étais seul à assumer à la fois le rôle de développeur et de Product Owner. Finalement, je me suis retrouvé à prendre du retard sur certaines fonctionnalités et par conséquent je n'ai pas pu délivrer un prototype qui respectait toutes les attentes du mandant.

En conclusion, j'ai retenu beaucoup de leçons en développant cette application et en rédigeant ce mémoire. Entre autres, l'utilité de rédiger la documentation dès le début du projet, afin d'être au clair avec le cheminement des étapes pour arriver jusqu'au bout, mais également la communication. Cette dernière avec le mandant, m'a permis de me rassurer, de savoir que j'avance dans le bon sens et de rectifier le tir lors de certaines fonctionnalités à implémenter ou des aspects visuels à corriger. Je ressors désormais plus confiant en termes de méthodologie de gestion de projet, mais également avec les connaissances des langages de programmation web pour me lancer dans le monde pratique avec des projets de grande envergure dans le domaine qui me plaît : celui du développement Full Stack que ce soit dans le web ou le mobile.

# Bibliographie


EDOUARD DIEP, 31 août 2019.

ANGULAR GUIDE, 2019. Angular - Observables. [en ligne].
[Consulté le 23 août 2019]. Disponible à l'adresse : https://angular.io/guide/observables

ANGULAR TUTORIAL, 2019. Angular - Services. [en ligne].
[Consulté le 23 août 2019]. Disponible à l'adresse : https://angular.io/tutorial/toh-pt4

APPARENCE, 2019. Application native ou hybride ? [en ligne]. 26 février 2019.
[Consulté le 19 août 2019]. Disponible à l'adresse :
https://apparence.io/blog/application-native-ou-hybride

DORIS BOEHLEN, 2018. Rapport agricole 2018 - Vin. | *OFAG* [en ligne].
[Consulté le 17 août 2019]. Disponible à l'adresse :
https://www.agrarbericht.ch/fr/marche/produits-vegetaux/vin

EDUCBA, 2019. Django vs Laravel - Know The 8 Most Valuable Differences. [en ligne].
[Consulté le 21 août 2019]. Disponible à l'adresse : https://www.educba.com/django-vs-laravel

FABIAN ROPARS, 2018. Comment continuer à utiliser l'API Google Maps gratuitement. *BDM* [en ligne]. 17 juillet 2018. [Consulté le 21 août 2019]. Disponible à l'adresse : https://www.blogdumoderateur.com/google-maps-gratuit-professionnels

FULLSCALE., 2019. Django vs Laravel: Performance Comparison of Web Application Frameworks. *Full Scale* [en ligne]. 10 mars 2019. [Consulté le 21 août 2019]. Disponible à l'adresse : https://fullscale.io/django-vs-laravel-performance-comparison-of-web-application-frameworks

GOOGLE CLOUD PLATFORM, 2019. API – Google Maps – VaudVin-map – Google Cloud Platform. [en ligne]. [Consulté le 21 août 2019]. Disponible à l'adresse :
https://console.cloud.google.com/google/maps-apis/api-list?authuser=0&project=vaudvin-map&folder=&organizationId=&supportedpurview=project

GOOGLE DEVELOPERS, 2019. Geocoding Service | Maps JavaScript API. *Google Developers* [en ligne]. [Consulté le 22 août 2019]. Disponible à l'adresse :
https://developers.google.com/maps/documentation/javascript/geocoding

GOOGLE DEVELOPERS, 2019. Place Search | Places API. *Google Developers* [en ligne]. [Consulté le 24 août 2019]. Disponible à l'adresse :
https://developers.google.com/places/web-service/search

INFOWEBMASTER, 2019. Responsive web design - Définition. [en ligne].
[Consulté le 19 août 2019]. Disponible à l'adresse :
http://glossaire.infowebmaster.fr/responsive-web-design

INDIA, SPEC, 2018. React Native vs. Ionic Explained: A Step-by-Step Evaluation. *Medium* [en ligne]. 28 août 2018. [Consulté le 19 août 2019]. Disponible à l'adresse :
https://codeburst.io/react-native-vs-ionic-explained-a-step-by-step-evaluation-23975999887

INDIA, SPEC, 2019. React Native vs. Ionic vs. Flutter: Comparison of Top Cross-Platform App Development Tools. *Medium* [en ligne]. 5 juillet 2019. [Consulté le 20 août 2019]. Disponible à l'adresse : https://codeburst.io/react-native-vs-ionic-vs-flutter-comparison-of-top-cross-platform-app-development-tools-71c8011309ac

IONIC FRAMEWORK, 2019. Adding-pages. *Ionic Framework* [en ligne]. [Consulté le 23 août 2019]. Disponible à l'adresse : https://ionicframework.com/docs/v3/intro/tutorial/adding-pages

IONIC FRAMEWORK, 2019. ion-searchbar component - Ionic Documentation. *Ionic Docs* [en ligne]. [Consulté le 24 août 2019]. Disponible à l'adresse : https://ionicframework.com/docs/api/searchbar

IONIC FRAMEWORK, 2019. ion-toolbar component - Ionic Documentation. *Ionic Docs* [en ligne]. [Consulté le 24 août 2019]. Disponible à l'adresse : https://ionicframework.com/docs/api/toolbar

IONIC FRAMEWORK, 2019. Project-structure. *Ionic Framework* [en ligne]. [Consulté le 23 août 2019]. Disponible à l'adresse : https://ionicframework.com/docs/v3/intro/tutorial/project-structure

IONIC FRAMEWORK, 2019. Starters. *Ionic Framework* [en ligne]. [Consulté le 24 août 2019]. Disponible à l'adresse : https://ionicframework.com/docs/v3/cli/starters.html

JENIECE PETTITT & LAUREN THOMAS, CNBC 2016. Artificial intelligence and wine. [en ligne]. 29 juillet 2016. [Consulté le 18 août 2019]. Disponible à l'adresse : https://www.cnbc.com/2016/07/29/artificial-intelligence-and-wine.html

LARAVEL DOCUMENTATION, 2019. Controllers - Laravel - The PHP Framework For Web Artisans. [en ligne]. [Consulté le 23 août 2019]. Disponible à l'adresse : https://laravel.com/docs/5.8/controllers

LÉA DELAGE, 2018. Vivino, l'appli qui passe le vin au rayon X. *Lemonde* [en ligne]. 9 avril 2018. [Consulté le 18 août 2019]. Disponible à l'adresse : https://www.lemonde.fr/m-perso/article/2018/04/09/vivino-l-appli-qui-passe-le-vin-au-rayon-x_5283017_4497916.html

LECLERC, 2019. Guide du vin - WineAdvisor : l'application qui vous aide à bien choisir votre vin. [en ligne]. [Consulté le 18 août 2019]. Disponible à l'adresse : https://www.macave.leclerc/guide-vin/cat/tout_savoir_sur_le_vin/post/wine-advisor-application

LENOUVELLISTE, 2018. La consommation de vin continue de baisser en Suisse. [en ligne]. 24 avril 2018. [Consulté le 18 août 2019]. Disponible à l'adresse : https://www.lenouvelliste.ch/articles/suisse/la-consommation-de-vin-continue-de-baisser-en-suisse-les-rouges-davantage-touches-752471

LUCAS, Ely, 2019. How to Lazy Load in Ionic Angular. *The Ionic Blog* [en ligne]. 1 mars 2019. [Consulté le 23 août 2019]. Disponible à l'adresse : https://ionicframework.com/blog/how-to-lazy-load-in-ionic-angular

LUNABEANMEDIA, 2014. Delectable - The Whys and Hows of This Great Wine App - *Lunabean Media Wine Marketing & Web Design* [en ligne]. 10 avril 2014. [Consulté le 18 août 2019]. Disponible à l'adresse : https://www.lunabeanmedia.com/delectable-the-whys-and-hows-of-this-great-wine-app

MIKE BUTUSOV, 2019. Native vs Hybrid apps. What to choose in 2019? *TechMagic - Blog* [en ligne]. 6 février 2019. [Consulté le 19 août 2019]. Disponible à l'adresse : https://blog.techmagic.co/native-vs-hybrid-apps

NILANCHALA, 2017. A Brief Introduction to Laravel PHP Framework Features and Version History | Stacktips. *Stacktips - How-to Guides, Tutorials and Code Snippets* [en ligne]. 24 avril 2017. [Consulté le 25 août 2019]. Disponible à l'adresse : https://stacktips.com/laravel/intro-to-laravel-php-framework-and-features

FRANÇOIS POTEVIN, 2017. Les meilleures applications mobiles communautaires sur le vin | *Vins du monde* [en ligne]. 29 avril 2017. [Consulté le 18 août 2019]. Disponible à l'adresse : https://vinsdumonde.blog/les-meilleures-applications-mobiles-de-reseau-social-sur-le-vin

ROBYN – MACSOURCES, 2013. Hello Vino, iOS App Review | *Mac Sources* [en ligne]. 6 septembre 2013. [Consulté le 18 août 2019]. Disponible à l'adresse : https://macsources.com/hello-vino-ios-app-review

RTS, 2018. La consommation de vin a poursuivi sa baisse l'an dernier en Suisse - *rts.ch – Economie* [en ligne]. 24 avril 2018. [Consulté le 17 août 2019]. Disponible à l'adresse : https://www.rts.ch/info/economie/9516148-la-consommation-de-vin-a-poursuivi-sa-baisse-l-an-dernier-en-suisse.html

RUSHI TRIVEDI, 2019. Top 5 hybrid Mobile App Frameworks in 2019 – Choose the best one for you. *Websoptimization Official Blog* [en ligne]. 11 avril 2018. [Consulté le 19 août 2019]. Disponible à l'adresse : https://www.websoptimization.com/blog/hybrid-mobile-app-frameworks

GUILLAUME SIMARD, 2018. Angular 5 (Component, Template, Modèle et Service). *Atomrace* [en ligne]. 7 mai 2018. [Consulté le 23 août 2019]. Disponible à l'adresse : https://atomrace.com/angular-5-component-template-modele-et-service

SOON SOON SOON, 2015. Delectable – Tout savoir sur un vin grâce à son étiquette | *Soon Soon Soon* [en ligne]. 28 mai 2015 [Consulté le 18 août 2019]. Disponible à l'adresse : https://www.soonsoonsoon.com/fr/detections/tout-savoir-sur-un-vin-grace-son-etiquette

SOPHIE MARENNE, 2018. Wine Picker: l'e-sommelier dans votre poche | *Agefi.com* [en ligne]. 13 décembre 2018. [Consulté le 18 août 2019]. Disponible à l'adresse : https://www.agefi.com/home/entreprises/detail/edition/online/article/oenologie-developpee-entre-londres-et-renens-lapplication-britannique-seduit-deja-au-royaume-uni-aux-etats-unis-mais-aussi-en-suisse-ou-elle-repertorie-60-restaurants-483836.html

STACKSHARE, 2019. Flutter vs Ionic vs React Native | What are the differences? *StackShare* [en ligne]. 17 juillet 2019 [Consulté le 19 août 2019]. Disponible à l'adresse : https://stackshare.io/stackups/flutter-vs-ionic-vs-react-native

SWISSWINE, 2014. Chiffres clés | Swiss Wine. [en ligne]. [Consulté le 17 août 2019]. Disponible à l'adresse : https://swisswine.ch/fr/vignoble/chiffres-cles

THINKMOBILES, 2019. Responsive Website vs Mobile App: Comparison. Thinkmobiles [en ligne]. [Consulté le 19 août 2019]. Disponible à l'adresse : https://thinkmobiles.com/blog/responsive-website-vs-mobile-app

THOMASVINO, 2018. Suisse et monde: les chiffres-clés du vin 2017 - *Thomasvino*. [en ligne]. 26 avril 2018. [Consulté le 18 août 2019]. Disponible à l'adresse : https://thomasvino.ch/?p=15078

VUFORIA ENGINE, 2019. Vuforia for Vivino. [en ligne]. [Consulté le 24 août 2019]. Disponible à l'adresse : https://engine.vuforia.com/case-studies/vivino.html

WIKIPEDIA., 2017. *Media queries* [en ligne]. [Consulté le 19 août 2019]. Disponible à l'adresse : https://fr.wikipedia.org/w/index.php?title=Media_queries&oldid=134794385

WIKIPEDIA, 2018. *Développement web frontal* [en ligne]. [Consulté le 23 août 2019]. Disponible à l'adresse : https://fr.wikipedia.org/w/index.php?title=D%C3%A9veloppement_web_frontal&oldid=151656771

WIKIPEDIA., 2019. *Framework* [en ligne]. [Consulté le 19 août 2019]. Disponible à l'adresse : https://fr.wikipedia.org/w/index.php?title=Framework&oldid=159097339

WIKIPEDIA, 2019. *Google Maps* [en ligne]. 14 août 2019 [Consulté le 21 août 2019]. Disponible à l'adresse : https://fr.wikipedia.org/w/index.php?title=Google_Maps&oldid=161813405

WIKIPEDIA, 2019. *UML (informatique)* [en ligne]. 26 juin 2019. [Consulté le 22 août 2019]. Disponible à l'adresse : https://fr.wikipedia.org/w/index.php?title=UML_(informatique)&oldid=160426787

# Annexe 1 : présentation du calcul de coefficient de correspondance

Données de la table des notes utilisateurs :

| | A | B | C | D | E | F | G | H | I | J | K | L |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Vin | Millésime | Moyenne | Personne 1 | personne 2 | personne 3 | personne 4 | personne 5 | personne 6 | personne 7 | personne 8 | personne 9 |
| 2 | Vin1 | 2018 | 2,2 | | 3 | 2 | | | 3 | 2 | | 1 |
| 3 | Vin2 | 2017 | 3,85714286 | 3 | 5 | | 3 | 3 | 5 | | 3 | 5 |
| 4 | Vin3 | 2017 | 2 | | 3 | 1 | 2 | | 3 | 1 | 2 | 2 |
| 5 | Vin1 | 2017 | 3,33333333 | 4 | 2 | 4 | | 4 | 2 | 4 | | |
| 6 | Vin1 | 2016 | 3,6 | 4 | | | 3 | 4 | | | 3 | 4 |
| 7 | Vin4 | 2017 | 2 | | 2 | | 2 | | | | | |
| 8 | Vin5 | 2018 | 4,33333333 | | | 5 | | 4 | | 4 | | |
| 9 | Vin6 | 2018 | 2,75 | 2 | | 2 | | 3 | | | 4 | |
| 10 | Vin4 | 2016 | 3,5 | | 2 | | 5 | | 4 | 3 | | |
| 11 | Vin5 | 2016 | 3 | 2 | | 3 | | 4 | 5 | | 2 | 2 |

Données de la table des notes concours :

| | A | B | C | D | E | F | G |
|---|---|---|---|---|---|---|---|
| 1 | Vin | Millésime | Moyenne | concours 1 | concours 2 | concours 3 | concours 4 |
| 2 | Vin1 | 2018 | 4,725 | 4,25 | 5,2 | | |
| 3 | Vin2 | 2017 | 4,27 | | 4,74 | 3,8 | |
| 4 | Vin3 | 2017 | 4,85 | 4,8 | | | 4,9 |
| 5 | Vin1 | 2017 | 2,9 | 2,8 | 3 | 2,9 | |
| 6 | Vin1 | 2016 | 2,5 | 3 | 2 | | |
| 7 | Vin4 | 2017 | 4,65 | | | 4,5 | 4,8 |
| 8 | Vin5 | 2018 | 4,125 | 4,5 | 3,8 | 4 | 4,2 |
| 9 | Vin6 | 2018 | 2,9 | 3 | | 2,8 | |
| 10 | Vin4 | 2016 | 4 | | 4 | | 4 |
| 11 | Vin5 | 2016 | 0 | | | | |

Données de la table de correspondance :

| | A | B | C | D | E | F | G | H | I | J |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Table de correspondance | Personne 1 | Personne 2 | Personne 3 | Personne 4 | Personne 5 | Personne 6 | Personne 7 | Personne 8 | Personne 9 |
| 2 | Personne 1 | 1 | 0 | 1 | 1 | 0,80000001 | 0 | 1 | 0,75 | 0,66666669 |
| 3 | Personne 2 | 0 | 1 | 0,33333334 | 0,5 | 0 | 0,80000001 | 0,5 | 0,5 | 0,66666669 |
| 4 | Personne 3 | 1 | 0,33333334 | 1 | 1 | 1 | 0,25 | 1 | 0,66666669 | 1 |
| 5 | Personne 4 | 1 | 0,5 | 1 | 1 | 1 | 0,66666669 | 0,5 | 1 | 0,66666669 |
| 6 | Personne 5 | 0,80000001 | 0 | 1 | 1 | 1 | 0,33333334 | 1 | 0,75 | 0,33333334 |
| 7 | Personne 6 | 0 | 0,80000001 | 0,25 | 0,66666669 | 0,33333334 | 1 | 0,5 | 0,33333334 | 0,5 |
| 8 | Personne 7 | 1 | 0,5 | 1 | 0,5 | 1 | 0,5 | 1 | 1 | 1 |
| 9 | Personne 8 | 0,75 | 0,5 | 0,66666669 | 1 | 0,75 | 0,33333334 | 1 | 1 | 0,75 |
| 10 | Personne 9 | 0,66666669 | 0,66666669 | 1 | 0,66666669 | 0,33333334 | 0,5 | 1 | 0,75 | 1 |

Coefficients de correspondance (table des pronostics) :

| | A | B | C | D | E | F | G | H | I | J | K |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Pronostics | mill | Personne 1 | Personne 2 | Personne 3 | Personne 4 | Personne 5 | Personne 6 | Personne 7 | Personne 8 | Personne 9 |
| 2 | Vin1 | 2018 | 0,57833332 | 3 | 2 | 0,57318181 | 0,57550001 | 3 | 2 | 0,56187648 | 1 |
| 3 | Vin2 | 2017 | 3 | 5 | 0,76313311 | 3 | 3 | 5 | 0,77649748 | 3 | 5 |
| 4 | Vin3 | 2017 | 0,51428819 | 3 | 1 | 2 | 0,50476414 | 3 | 1 | 2 | 2 |
| 5 | Vin1 | 2017 | 4 | 2 | 4 | 0,66166669 | 4 | 2 | 4 | 0,67000002 | 0,65566665 |
| 6 | Vin1 | 2016 | 4 | 0,63333333 | 0,66389608 | 3 | 4 | 0,64236361 | 0,66606063 | 3 | 4 |
| 7 | Vin4 | 2017 | 0,62714285 | 2 | 0,59875 | 2 | 0,62714285 | 0,59875 | 0,59875 | 0,59875 | 0,59875 |
| 8 | Vin5 | 2018 | 0,85436505 | 0,854375 | 5 | 0,8572222 | 4 | 0,84594017 | 4 | 0,84894633 | 0,85912699 |
| 9 | Vin6 | 2018 | 2 | 0,58375001 | 2 | 0,55900002 | 3 | 0,58272725 | 0,55900002 | 4 | 0,55233335 |
| 10 | Vin4 | 2016 | 0,76249999 | 2 | 0,74935484 | 5 | 0,76666665 | 4 | 3 | 0,74176472 | 0,72294116 |
| 11 | Vin5 | 2016 | 2 | 0,62153208 | 3 | 0,59166664 | 4 | 5 | 0,57575756 | 2 | 2 |