

Rapport de conception



UNE APPLICATION POUR EVALUER LES EMOTIONS SUSCITEES PAR DES PRODUITS INNOVANTS (V3)



Université Pierre Mendès-France
Sciences sociales & humaines

Projet encadré par :

- Damien Dupré, *Laboratoire Interuniversitaire de Psychologie*
- Anna Tcherkassof, *Laboratoire Interuniversitaire de Psychologie*
- Manuel ATENCIA ARCAS, *Laboratoire Informatique de Grenoble*

Équipe de développement du projet :

- Alizée Arnaud
- Jordan Daïta
- Rémy Drouet

Préface

Dans le cadre de la formation de Master WIC 1ère année nous avons la chance de réaliser un projet qui nous servira de fil rouge tout au long de l'année d'étude. Parmi les projets qui nous ont été proposés nous avons choisi celui-ci. L'aspect développement mobile sur support tactile et le challenge qu'il suscite ont capté toute notre attention. Nous sommes heureux aujourd'hui de participer à ce projet.

Remerciements

Nous tenons à remercier Anna Tcherkassof et Damien Dupré pour nous avoir permis de réaliser ce projet dans des conditions optimum. En effet, leurs disponibilités pour les réunions d'avancement ainsi que la mise à disposition de locaux pour effectuer ce stage de développement nous ont été plus que bénéfiques.

Nous remercions aussi Manuel Atencia Arcas et Benoît Lemaire qui nous ont permis de réaliser ce projet annuel au cours de la formation de Master WIC.

Au travers de ces personnes nous remercions bien évidemment les structures : LIG, LIP et EXMO, qui soutiennent et investissent dans ces initiatives.

Table des matières

Préface	1
Remerciements	2
Introduction	5
1 Analyse et conception	6
1.1 Synthèse de la phase d'analyse	6
1.1.1 Le projet	6
1.1.2 L'application	6
1.2 Synthèse de la phase de conception	9
1.2.1 Système de versionning	9
1.2.2 Répartition des tâches (Annexe 3)	9
1.2.3 Choix de conception	9
1.2.4 Politique de Test	10
1.2.5 Réalisation des tests	11
1.2.6 Cas d'utilisation (Annexe 4)	11
2 Gestion de projet et organisation de la phase de développement	12
3 L'application	14
3.1 Gestion multilingue	14
3.1.1 Changement de langue	14
3.2 Section administrateur	14
3.2.1 Gestion des expériences	14
3.2.2 Gestion des langues	18
3.2.3 Difficultés rencontrés	20
3.3 Choix de développement	22
3.3.1 Structure des fichiers & maintenabilité (Annexes 1 & 2)	22
3.3.2 Retour sur le WebGL	22
3.3.3 Base de données	22
4 WEBGL	23
4.1 Utilisation de la librairie ThreeJS	23
4.1.1 Principe	23
4.1.2 Prise en main	23

4.1.3	Rappel des besoins de l'expérience	23
4.2	Les étapes de réalisation	23
4.2.1	Comprendre la technologie	23
4.2.2	Création de l'environnement et de l'avatar	24
4.2.3	Intégration et animation de l'avatar	25
4.2.4	Création et enregistrement des données	30
5	Manuel utilisateur	31
6	Améliorations possibles	32
7	Conclusion	33
	Webographie	34
	Information sur le WebGL	34
	La bibliothèque ThreeJS	34
	Outils de modélisation 3D	34
	Outils d'aide au développement frontend	34
	Sites d'entraide généralistes	35
	Annexes	36
	Annexe 1 : Base de données Conception	36
	Annexe 2 : Base de données Développement	37
	Annexe 3 : Diagramme de Pert	38
	Annexe 4 : Cas d'utilisation	39

Introduction

L'émotion est au centre de ce projet. Elle est définie comme le résultat de modifications internes et externes, spontanées et transitoires initiées par un "objet". C'est un phénomène psychophysiologique très complexe associé à l'humeur, au tempérament et à la personnalité d'un individu.

Cinq composantes constituent l'émotion:

- Cognition
- Ressenti subjectif
- Expressive
- Motrice
- Périphérique et motivationnelle

Des recherches ont permis de mesurer certaines des composantes précitées, mais très peu mesurent la composante motivationnelle des émotions en réponse à l'évaluation de produits.

C'est le fondement du projet Emolyse, développer un outil plus adapté et performant pour mesurer les intentions comportementales chez des individus face aux tests de produits innovant.

Suite à l'édition de 2 versions du projet, Emolyse persévère dans sa recherche d'application qui répondra au mieux à ses attentes. C'est pourquoi aujourd'hui nous entamons une troisième version de l'application.

Dans ce troisième et dernier rapport nous traiterons de la phase de développement du projet. Les deux premiers nous ont permis de réaliser toute l'analyse et la conception de la demande, nous y ferons référence tout au long de ce rapport. Nous avons eu 1 mois pour effectuer la mise en œuvre technique, de plus nous avons eu la chance d'avoir à notre disposition des locaux au sein de la plateforme Domus.

1 Analyse et conception

1.1 Synthèse de la phase d'analyse

1.1.1 Le projet

L'application Emolyse sera utilisée lors d'expériences au Laboratoire Interuniversitaire de Psychologie pour évaluer l'émotion suscitée par un produit innovant. Elle doit répondre à un besoin né lors des recherches du laboratoire.

Damien Dupré, Doctorant, et Anna Tcherkassof, Maître de conférence, sont à l'origine du projet Emolyse. Leur premier besoin est de créer une application tactile qu'ils utiliseront lors d'expérimentations. L'application doit être paramétrable, ce qui nécessite une interface d'administration afin de gérer les consignes, la langue, les fichiers téléchargeables...

L'utilisateur, un individu entre 18 et 50 ans, se servira de cet outil en s'identifiant à un personnage 3D qu'il choisira et pourra mouvoir au sein d'un environnement 3D également pour augmenter l'immersion.

Le but est de permettre à cet utilisateur de représenter son état émotionnel via cet avatar lorsqu'il testait des objets innovants.

A l'issue de ces expérimentations, le laboratoire a besoin de récupérer les données de positionnement de l'avatar après validation de l'utilisateur. Cela sera sous forme de fichier .csv comprenant la distance par rapport à l'objet, l'angle d'inclinaison du buste ou encore les angles des bras.

L'objectif prioritaire est de produire une application stable et fonctionnelle répondant aux besoins de base du client.

L'avatar et l'environnement 3D sont une contrainte du client, en effet les utilisateurs semblent bien plus réceptifs à ce genre d'environnement.

La portabilité est un élément clé du développement auquel nous ne pouvons déroger.

1.1.2 L'application

Deux versions du projet Emolyse existent déjà mais ne conviennent pas aux besoins réels du client.

Deux technologies ont été utilisées, à savoir:

- Unity, très puissant mais aussi très complexe à mettre en place
- HTML, 3D simulée (2D avec profondeur)

Celles-ci ne répondant pas parfaitement aux attentes nous avons effectué beaucoup de recherche afin de trouver une solution performante, immersive et attrayante.

C'est pourquoi nous avons choisi WebGL, une technologie web permettant le développement d'interfaces 3D dynamiques pour les pages et applications HTML5.

C'est une technologie qui permet de gérer dynamiquement des éléments graphiques complexes en 3D dans un navigateur.

Niveau compatibilité, pour répondre à la contrainte de portabilité, tous les navigateurs récents prennent en charge WebGL, nous pourrions lancer l'application de n'importe quel périphérique. Néanmoins, les pilotes de cartes graphiques anciennes ou bas de gamme ne sont pas toujours adaptés.

De plus WebGL est Open Source ce qui permettra si le laboratoire le souhaite de déposer une licence pour Emolyse. La communauté autour de WebGL semble très active, nous avons pu voir de nombreux forums et sites divers ce qui nous prouve une fois de plus que cet outil est réellement prometteur.

Le fait que cette solution soit une technologie Web basée sur des bibliothèques JavaScript (Three.js, Babylon.js etc.) est très motivant pour notre équipe car cela correspond parfaitement à notre formation en cours. De plus, cet outil étant très prometteur, il pourra permettre à notre application d'être évolutive.

1.1.2.1 Rappel des spécifications fonctionnelles

Les spécifications fonctionnelles vous indiquent l'ensemble des interactions que l'on pourra avoir avec l'application en tant qu'administrateur ou utilisateur :

Pour le participant :

- Le participant doit pouvoir réaliser une ou plusieurs expériences
- Le participant doit pouvoir animer l'avatar qu'il aura choisi à travers l'interface tactile (pas de bouton de commande)
- Création d'un profil participant à l'issue d'une expérience
- On visualise dans un même espace l'avatar et l'objet observé : Inclure l'utilisateur dans cet espace.
- Changer d'objet (possibilité de retour sur un objet déjà vu)

Pour l'expérimentateur :

- Création des expériences
- Ajout des produits, de l'environnement et modification de la consigne pour une expérience
- Recueillir les informations des expériences achevées à partir de sa page de configuration

- L'expérimentateur a la possibilité de choisir et de configurer une langue.
- L'expérimentateur a accès à l'ensemble des expériences (mode expérimentateur), profil utilisateur (mode expérimentateur) et peut télécharger les résultats
- En mode expérimentateur, on peut naviguer par lien direct entre expérience

Afin que l'utilisateur soit plongé dans l'environnement, l'intégralité des commandes de contrôle sera tactile. En cas d'impossibilité de commande tactile pour une interaction l'option sera disponible dans un menu rétractable afin de ne pas parasiter l'affichage de l'environnement.

Néanmoins le contrôle tactile nécessite une phase d'apprentissage (tutoriel) de contrôle de l'avatar qui sera disponible avant le lancement d'une expérience.

1.1.2.2 Rappel des spécifications non-fonctionnelles

Ces spécifications fixent nos contraintes et objectifs internes à l'application qui guideront notre développement du projet :

- Technologie WebGL (Portabilité, Interface sur navigateur)
- Fluidité de l'application
- Rapidité des chargements
- Mise en place d'une licence

1.1.2.3 Rappel des risques & solutions

Dans tout projet il y a des risques, voici une liste les mentionnant avec les solutions apportées.

- Perte de données lors du développement. Pour parer à cela nous sauvegarderons régulièrement notre travail sur plusieurs supports.
- Membre indisponible pendant plusieurs volumes horaires. La pluridisciplinarité des membres de l'équipe permettra de combler une éventuelle indisponibilité.
- Prise de retard sur une tâche. Prévoir un planning large avec des plages « vides » pour combler le retard si nécessaire.
- Non satisfaction du client. Prévoir des validations régulières à intervalles courts pour rendre compte de l'avancement du projet et éventuellement, pour recadrer l'équipe de développement.
- Manque de compétences de l'équipe. Ateliers de prise en mains des technologies nouvelles en amont de la phase de développement.

1.2 Synthèse de la phase de conception

1.2.1 Système de versionning

Au niveau de l'organisation et du partage du travail, nous avons utilisé le logiciel de gestion de version GitHub. Contrairement à ce que nous avons prévu l'intérêt de ce projet est d'être accessible publiquement. Le service qu'offre GitHub était donc complètement adapté à nos besoins.

Il nous a permis d'effectuer des tâches en parallèle pour un travail collaboratif productif.

1.2.2 Répartition des tâches (Annexe 3)

Nous avons réalisé un diagramme de Pert pour mieux visualiser et appliquer cette répartition des tâches. Nous avons essayé de faire en sorte que 3 tâches puissent se faire simultanément afin de répartir de manière efficace le travail. Il n'y a pas de chemin critique, mais nous avons identifié le chemin le plus dangereux qui est :

Initialisation → Création d'un avatar → Animation avatar → Expérience 1 objet →
Extraction des données → Enchaînement expérience → Finalisation

Mais comme indiqué précédemment, ce chemin dispose de tâches avec des prédictions assez larges car nous avons moins d'expérience dans ces domaines (création d'avatar, animation avatar). Lors de la phase de développement ce chemin a en effet été le plus difficile à gérer. Néanmoins nos évaluations de la durée des tâches ayant suffisamment de marge nous avons pu sans aucune appréhension terminer ce projet dans les délais imposés.

1.2.3 Choix de conception

1.2.3.1 Classe produit

Chaque produit est unique et lié à une expérience. En effet, nous avons convenu qu'il serait plus simple d'utilisation de créer un nouveau produit pour chaque expérience configurée plutôt que de récupérer un produit existant. Cette décision résulte du traitement des données par le client qui ne considère les produits que dans le cadre de l'expérience à laquelle il est lié. Les produits sont enregistrés dans la base de données.

1.2.3.2 Avatar

L'avatar peut avoir deux formes : un avatar homme et un avatar femme.

1.2.3.3 Identifiant - Langue - Vocabulaire

L'objectif est de créer une interface multilingue configurable par l'*expérimentateur* dans le menu **paramètre**. Les 2 langues par défaut disponibles seront l'anglais et le français.

Ainsi à chaque élément de l'interface incluant du texte on attribue un identifiant (table IDENTIFIANT). Pour chaque association IDENTIFIANT-LANGUE on aura un terme correspondant (voir base de données).

Attention, le client souhaitait pouvoir éditer une consigne d'expérience qui sera présentée au participant avant de commencer l'expérience. Une consigne par défaut sera alors configurée pour chaque langue dans le menu paramètre. Cette consigne par défaut sera considérée comme un terme.

Un aperçu du résultat de l'association sera :

Identifiant	Français	Anglais
btn_ajout	Ajouter	Add
btn_demarrer	Démarrer	Start
accueil	Accueil	Home

Ce contenu sera éditable dans le menu **Accueil > Expérimentateur > Paramètres**.

1.2.4 Politique de Test

Les critères de tests choisis sont issus de la norme ISO/CEI 9126. Cette partie reflète les choix de test déterminés lors de la phase de conception.

1.2.4.1 Capacité fonctionnelle

Ce projet étant à but expérimentale et les conditions d'utilisation étant clairement définies nous passerons les tests d'interopérabilité et de conformité. En outre, les données obtenues via l'application se doivent d'être pertinentes et exactes. Ainsi, on effectuera un traitement d'échantillon des données obtenues et on le comparera avec les données de l'existant déjà validées par le client. Concernant la performance, des tests réguliers seront effectués sur la tablette au fur et à mesure du développement des objets 3D afin de ne pas perdre en fluidité.

1.2.4.2 Fiabilité

La tolérance aux pannes est difficile à gérer pour une équipe non-expérimentée comme la nôtre sur ce type de technologie. C'est pourquoi nous nous efforcerons de gérer au mieux ce type d'inconvénient. Nous ne pouvons pas d'ors et déjà anticiper ces problèmes ni les tests qui correspondent. Les tests seront donc non-conventionnels et issue des cas rencontrés lors de la phase de développement. Il en sera de même pour la maturité de l'application.

1.2.4.3 Facilité d'utilisation : compréhension, apprentissage, exploitation

Pour ces critères nous nous baserons sur les retours réguliers de nos clients que nous obtiendrons lors de chaque mise au point et présentation de notre avancement dans le projet. De plus, nous gardons à l'esprit que notre application peut toucher un public large au travers des participants. C'est pourquoi ces critères ne sont pas négligés mais au contraire fortement considérés tout au long de notre réalisation du projet.

1.2.4.4 Maintenabilité

Notre équipe étant composée de trois membres nous pourrions effectuer des analyses mutuelles de nos tâches développées afin de valider ou d'améliorer les facilités d'analyse et de modification. Ainsi une relecture du travail d'autrui agrémenté de questions au développeur de la tâche permettra d'enrichir les explications/choix de développement et ainsi assurer la maintenabilité.

1.2.5 Réalisation des tests

Lors de la phase de conception nous avons déterminé une série de scénarios visant à détecter des problèmes liés au bon fonctionnement ou à l'ergonomie de l'application. En appliquant ces tests, nous avons en effet corrigé l'ensemble des problèmes repérés :

- Cliquer dans l'espace grisé entourant un pop-up pour le quitter.
- Bouton pour basculer en pleine écran (détecté lors d'un test sur tablette)

Additionné à ces tests développeur, les réunions avec les clients nous ont permis de détecter d'autres petits problèmes utilisateurs :

- Ordre des boutons dans l'accueil expérimentateur
- L'icône de suppression d'une expérience

1.2.6 Cas d'utilisation (Annexe 4)

L'application développée intègre la majorité des cas d'utilisation déterminés lors de la phase de conception, excepté le téléchargement des résultats sur la page Participants qui s'effectue désormais dans la page expériences.

2 Gestion de projet et organisation de la phase de développement

Lors de la phase de conception nous avons mis en place un tableau des tâches à effectuer. Nous avons estimé la durée de chaque tâche et avons pu construire un diagramme de Pert (présenté lors de la phase de conception). Tout au long de cette phase de développement nous avons établi chaque lundi un tableau de « Sprint » (méthode agile) regroupant toutes les tâches sur lesquelles nous allions travailler. Tous les membres de l'équipe devaient exécuter 10 unités de temps, soit 5 journées. Si l'un d'entre nous achevait



son travail plus tôt, il pouvait s'ajouter une nouvelle « carte » pour avancer plus vite. Tous les soirs, nous faisons une petite réunion « Bilan » pour indiquer les durées réelles dans le tableau de tâches et expliquer nos actions de la journée avec les éventuelles difficultés rencontrées. Ces opérations quotidiennes nous ont permis d'avoir une bonne notion d'avancement du projet.

Au niveau de la répartition des tâches, nous nous sommes focalisé sur l'efficacité. Un membre de notre équipe étant particulièrement à l'aise avec le langage PHP, il a commencé directement le développement de la partie gestion de l'application. Le reste de notre équipe a pu travailler ensemble sur la prise en mains de la technologie WebGL. Une fois l'administration terminée, nous nous sommes réunis pour travailler ensemble sur la résolution des problèmes. Ce système nous a permis d'étudier le travail de chacun et de penser à d'éventuelles optimisations ou nouveautés.

Tout au long du développement nous sommes resté en contact avec Damien Dupré et Anna Tcherkassof. Nous avons fait deux réunions, la première que nous appellerons « d'étape » (28 juin 2015) avait pour objectif d'obtenir des réponses à nos diverses questions et de recueillir leur premières impressions face à l'application en cours de développement.

Utiliser une méthode de gestion de projet « agile » nous a permis de rebondir sur cette réunion et d'intégrer au projet tous les correctifs et nouvelles demandes du client. Une réflexion collective a apportée certaines remises en questions et modifications de l'application mais cela n'a pas du tout affecté notre planning. Nous avons pu faire preuve

d'une grande adaptabilité pour arriver au résultat attendu. De nouvelles cartes sont apparues et nous avons été capables de répondre aux nouvelles attentes.

La seconde réunion dite de « validation » a eu lieu deux semaines plus tard, le 8 juin 2015. A cette étape nous avons pu faire une démonstration du produit fini. L'application a été validée, seuls quelques détails devaient être modifiés. Nous avons également montré le manuel utilisateur que nous avons créé et fait le point sur l'ensemble du stage (déroulement, organisation). Il ne nous restait plus qu'à entamer le rapport de projet.

Pour conclure, nous sommes très satisfaits de la gestion de projet que nous avons pu mettre en place. Le fait d'avoir beaucoup anticipé lors de la phase de conception a été un véritable atout à la réussite de ce projet. Pour cette troisième phase, nous avons déjà attribué les tâches et prévu suffisamment de marge pour prendre en compte tous les retours des clients.

3 L'application

3.1 Gestion multilingue

3.1.1 Changement de langue

Une demande très importante était de pouvoir traduire l'application en plusieurs langues. Nous avons donc commencé le développement par la construction de l'interface administrateur en multi-langues. C'est sur la page d'accueil que l'expérimentateur sera en mesure de passer l'application d'une langue à l'autre. Nous savons qu'il n'y aura pas plus de trois ou quatre langues différentes, donc nous avons fait le choix de disposer directement les petits drapeaux des langues disponibles en haut à droite de la page d'accueil. C'était un choix logique car connu du grand public et peu coûteux en place.

Nous traiterons dans la section 2.2.2 la gestion des traductions qui se trouve dans la section administrateur.

3.2 Section administrateur

3.2.1 Gestion des expériences

Une part importante du projet est la gestion de l'application par un expérimentateur. Elle devait être simple et claire d'utilisation. Lors de la phase de conception nous avons réalisé toutes les maquettes graphiques mais quelques modifications mineures ont été apportées pour gagner en ergonomie et praticité. L'application est destinée à être utilisée sur un appareil tactile, dans cette optique nous avons fait attention à la taille des différents boutons pour éviter les erreurs de manipulation. Par exemple, les pages "accueil" et "accueil expérimentateur", qui sont des pages menu, sont composées de boutons très larges.

Dans la page "Expérimentateur", le premier lien permet d'accéder à la gestion des expériences. Nous retrouvons un tableau récapitulatif (voir Figure 1) de l'ensemble des expériences disponibles. Pour chacune d'entre elles on peut télécharger les données résultats

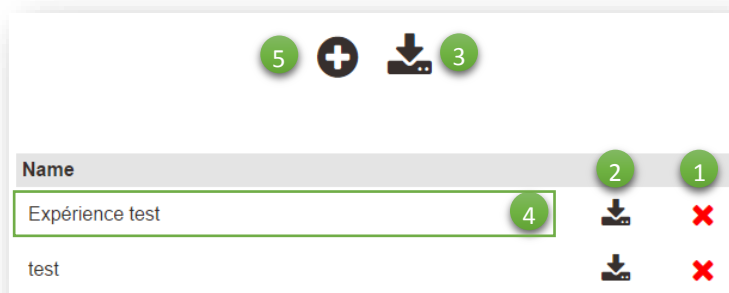


Figure 1 : Tableau des expériences

dans un fichier .csv ou la supprimer (Figure 1 : 1). Pour ne pas surcharger la page et le serveur, la création du fichier .csv est effectuée seulement au clic sur l'icône de téléchargement (Figure 1 : 2). La génération est invisible

pour l'utilisateur. Le téléchargement sera donc effectué directement après sur l'appareil. Si l'expérimentateur souhaite télécharger l'ensemble des fichiers .csv (pour toutes les expériences), il pourra en obtenir une archive zip au clic sur le plus gros bouton de

téléchargement au-dessus du tableau des expériences (Figure 1 : 3). Lorsque l'on quitte la page l'ensemble des fichiers générés sur le serveur sont supprimés.

Dans un fichier de résultat, les données résultantes de l'expérience d'un participant sont représentées sur 1 ligne (sur demande du client). Ainsi les premières colonnes représenteront les données d'identification du participant et de l'expérience. Ensuite chaque objet de l'expérience représentera une série de colonnes (une pour chacune des rotations et positions de l'avatar).

Un fichier résultat est composé des données suivantes :

- PARTICIPANT_ID
- PARTICIPANT_SURNAME
- PARTICIPANT_NAME
- DATE_OF_BIRTH
- PARTICIPANT_GENDER
- AVATAR_GENDER
- DATE_OF_PARTICIPATION

Puis, pour chaque objet nous avons (\$i correspond à l'id du produit pour les différencier les uns des autres) :

- PRODUCT_ID_\$i
- PRODUCT_NAME_\$i
- AVATAR_ROTATION_\$i
- LEFT_ARM_ANGLE_X_AXIS_\$i
- LEFT_ARM_ANGLE_Z_AXIS_\$i
- RIGHT_ARM_ANGLE_X_AXIS_\$i
- RIGHT_ARM_ANGLE_Z_AXIS_\$i
- BODY_ANGLE_\$i
- DISTANCE_\$i

Toutes ces informations ont été réfléchies et déterminées avec le laboratoire lors des réunions.

Si l'expérimentateur clique sur l'icône de suppression rouge, un pop-up de validation apparaîtra pour éviter les fausses manipulations et la perte de données importantes. Ainsi, à la suppression d'une expérience, toutes les données liées seront effacées. Plus précisément : les résultats de l'expérience concernée, les produits liés avec leur image sur le serveur, les participants et l'expérience elle-même.

Au clic sur le nom d'une expérience (Figure 1 : 4) ou sur l'icône « + » (Figure 1 : 5) d'ajout, nous arrivons sur une page commune de gestion d'expérience. Le seul changement est qu'à la sélection d'une expérience existante les informations et produits déjà renseignées seront présentes. Lors d'une nouvelle expérience, à l'ouverture de la page une nouvelle occurrence est ajoutée dans la base de données. Cela permet, comme lors d'une modification, une simple édition de l'expérience. Bien entendu, nous avons prévu le cas où un

expérimentateur se trompe et ne voulait en fait pas ajouter une nouvelle expérience. Il n'aura qu'à revenir en arrière et la détection du champ "nom de l'expérience" (Figure 2 : 1) vide supprimera la dernière entrée.

La figure 2 donne un aperçu de la page d'édition d'une expérience. Dans cette capture nous pouvons voir deux situations en cours d'exécution :

- Le clic sur le drapeau qui ouvre une petite « box » de sélection d'une langue pour l'expérience (Figure 2 : 2).
- Le drag & drop (cliqué glissé) d'un objet vers la poubelle pour le supprimer (Figure 2 : 3).

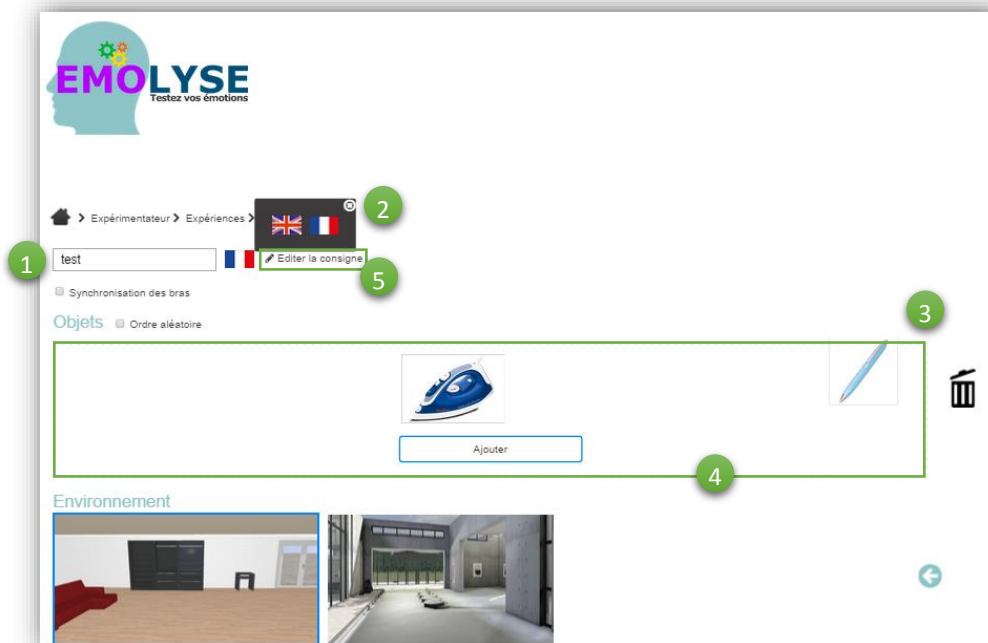


Figure 2 : Edition d'une expérience

Nous pouvons constater que le capot de la poubelle s'ouvre à l'approche de l'objet.

3.2.1.1 Méthode de développement

Plusieurs méthodes sont utilisées ici pour faire fonctionner correctement cette page. Tout d'abord, le système de drag & drop de jQuery qui permet de glisser directement des images dans l'application pour ensuite les télécharger et les enregistrer en tant qu'objets de l'expérience (Figure 2 : 4). Le format et la taille du fichier sont contrôlés lors de l'évènement « drop ».

L'interaction « Sortable » de jQuery UI est également utilisée pour trier les produits. Cette fonctionnalité était une demande importante du client pour augmenter le niveau de maniabilité de l'application. C'est aussi grâce à cette fonction que nous avons géré la suppression avec l'icône de la poubelle.

3.2.1.2 Enregistrement de l'expérience

Nous remarquons qu'il n'y pas de bouton de validation pour enregistrer les données modifiées. En réalité, à chaque action de l'expérimentateur, tout est automatiquement sauvegardé de façon dynamique. Si le nom est changé, à chaque événement "keyUp" le nom est modifié. Il en sera de même au clic sur un drapeau, à la sélection d'une checkbox, d'un environnement...

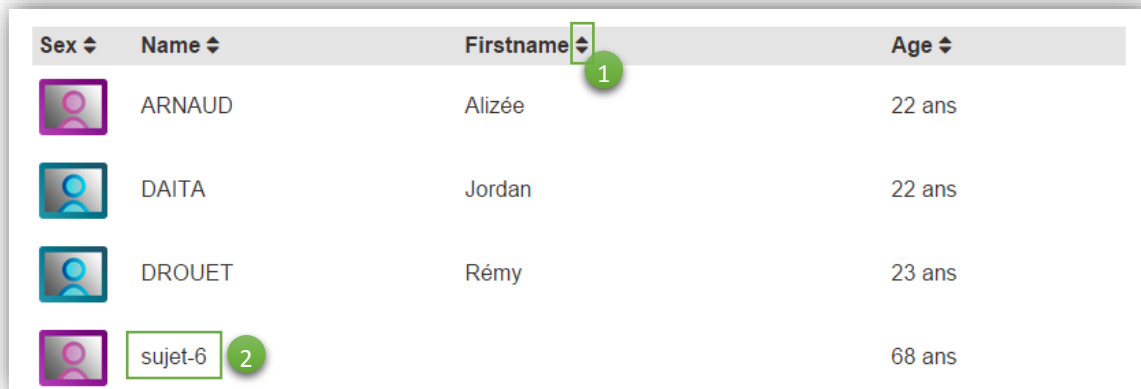
Tout ceci est possible grâce à des scripts JavaScript que nous avons voulu générique (nous pourrions nous en resservir dans des projets futurs). Nous avons donc créé des plugins que nous appelons dans la page. Ici, la plupart reçoivent ou écoutent certains paramètres qu'ils envoient ensuite à un script PHP pour modifier la base de données et enregistrer les données.

3.2.1.3 Gestion de la consigne

Pour la consigne de l'expérience, au clic sur le lien "Editer la consigne" (Figure 2 : 5) un pop-up s'affiche. La consigne par défaut sera alors éditée. Si elle est modifiée, ce n'est pas la consigne par défaut qui sera écrasée dans la base de données mais on l'ajoutera à l'expérience elle-même, nous considérons qu'elle devient spécifique à celle-ci.

3.2.1.4 Liste des participants

Une liste des participants est disponible. Elle se trouve dans l'onglet « participants » de la partie expérimentateur. Encore une fois cette liste est sous forme de tableau, cette fois-ci triable par colonne (Figure 3 : 1). Il n'est pas possible de supprimer un participant directement, mais pour ne pas avoir trop d'occurrences nous les supprimons en même temps que l'expérience à laquelle ils correspondent. Dans ce tableau pour différencier les deux sexes nous utilisons des pictogrammes de couleurs différentes et si elle est disponible la photo du participant. Nous avons établi avec les clients que seul l'âge et le sexe sont nécessaires. Pour avoir tout de même un nom pour chaque participant il nous a été demandé de renseigner un nom générique : « sujet- » suivi de son identifiant (Figure 3 : 2).







Sex	Name	Firstname	Age
	ARNAUD	Alizée	22 ans
	DAITA	Jordan	22 ans
	DROUET	Rémy	23 ans
	sujet-6		68 ans

Figure 3 : Tableau des participants

3.2.2 Gestion des langues

La troisième et dernière partie de la section expérimentateur correspond à la gestion des langues. C'est en fait la première chose que nous avons mis en œuvre. Cela nous a permis d'avoir au tout début du projet toutes les traductions et de pouvoir ajouter des fonctionnalités sans se soucier du multilingue.

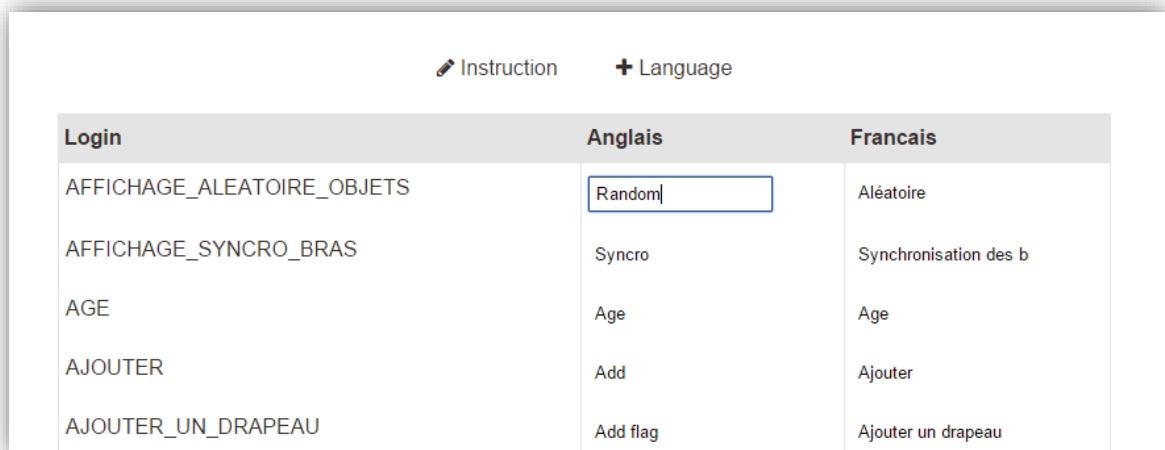
Pour chacun des mots ou chacune des phrases de l'application nous avons attribué un code langue qui est défini à la main dans une table dédiées de la base de données. C'est ensuite dans une table nommée "traduction" que nous attribuons une traduction pour une langue (voir figure 4).

EN	AGE	Age
FR	AGE	Age
EN	AJOUTER	Add
FR	AJOUTER	Ajouter
EN	AJOUTER_UN_DRAPEAU	Add flag
FR	AJOUTER_UN_DRAPEAU	Ajouter un drapeau
EN	AJOUTER_UNE_LANGUE	Add language
FR	AJOUTER_UNE_LANGUE	Ajouter une langue
EN	ANGLAIS	English

Figure 4 : Extrait de la table "Traduction"

Nous avons fait le choix de ne pas donner à l'expérimentateur la possibilité d'ajouter un identifiant car les appels se font directement dans le code PHP.

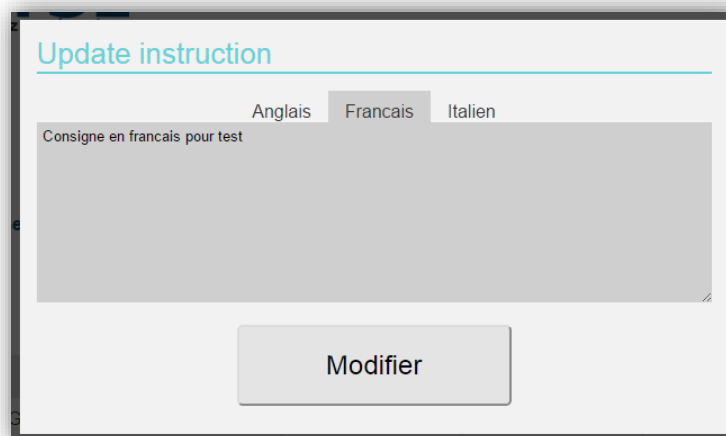
En revanche, il peut modifier les traductions. Dans la figure 5 ci-dessous nous voyons que nous pouvons modifier la traduction dynamiquement. Il suffit de cliquer sur le mot, celui-ci se transforme en champs texte, il ne reste plus qu'à effectuer les changements. A chaque événement "KeyUp" il y a un enregistrement il n'y a donc pas de bouton de validation.



Login	Anglais	Francais
AFFICHAGE_ALEATOIRE_OBJETS	<input type="text" value="Random"/>	Aléatoire
AFFICHAGE_SYNCRO_BRAS	Syncro	Synchronisation des b
AGE	Age	Age
AJOUTER	Add	Ajouter
AJOUTER_UN_DRAPEAU	Add flag	Ajouter un drapeau

Figure 5 : Tableau de gestion des traductions

Si l'on souhaite modifier la consigne par défaut (dans toutes les langues) il faut cliquer sur le lien « consigne ». Une fenêtre s'ouvrira dans l'application (figure 6). Sous forme d'onglets, la consigne sera disponible pour toutes les langues. Il ne restera plus qu'à la modifier et valider les changements.



Update instruction

Anglais Francais Italien

Consigne en francais pour test

Modifier

Figure 6 : Pop-up de la consigne

De la même manière que pour la consigne par défaut, pour ajouter une langue il faut cliquer sur le lien « Langue » et un pop-up s'affichera. Toutes les informations (code langue, nom de la langue, image du drapeau) sont requises sinon l'enregistrement sera refusé. Cela est fait dynamiquement afin de conserver les informations déjà entrées par l'expérimentateur.

Au moment de la validation de l'ajout d'une langue, une ligne est ajoutée dans la table « Langue » de la base de données. De plus, pour chaque identifiant de traduction, on insère automatiquement une traduction par défaut (anglais) dans la table traduction pour cette

nouvelle langue. Cela permet d'éviter certains problèmes mentionnés dans le paragraphe suivant.

3.2.3 Difficultés rencontrés

3.2.3.1 Le tableau de langues

Pour mener à bien cette partie de l'application une difficulté assez générale a été de toujours bien contrôler l'utilisabilité avec le tactile.

La première difficulté est survenue au tout début du projet, lors de la construction du tableau de traductions (figure 5). Une première requête va rechercher en base de données toutes les occurrences de la table "identifiants" pour les afficher dans la première colonne. Ensuite, dans cette même boucle nous allons chercher les traductions correspondantes. Cela nous a posé quelques problèmes car si nous avons trois langues et que toutes les traductions n'avaient pas été ajoutées pour chacune, le tableau ne laissait pas un trou vide mais il décalait les résultats (voir figure 7). Nous voyons que cela n'est pas du tout adapté, par exemple si la traduction de "AVRIL" n'existait pas en anglais, le français venait se mettre à la place.

AOUT	August	Août
AVRIL	Avril	
BTN_EXPERIENCES	Experiences	Expériences

Figure 7 : Décalage de la traduction du mot "Avril"

Pour pallier à ce problème nous avons donc décidé qu'à la création d'une nouvelle langue, nous attribuerions aussi à chaque identifiant une valeur par défaut pour cette langue. L'anglais a été choisi. La figure 8 illustre l'ajout d'une nouvelle langue. Notre solution palie au problème, une nouvelle colonne est apparue, avec toutes les traductions en anglais. L'expérimentateur n'a plus qu'à traduire l'ensemble en Italien.

✎ Instruction + Language			
Login	Anglais	Francais	Italien
AFFICHAGE_ALEATOIRE_OBJETS	Random	Aléatoire	Random
AFFICHAGE_SYNCRO_BRAS	Syncro	Synchronisation des b	Syncro
AGE	Age	Age	Age
AJOUTER	Add	Ajouter	Add

Figure 8 : Ajout d'une nouvelle langue avec des valeurs par défaut

Enfin, cette solution nous permet même d'avoir des valeurs vides (même si cela est déconseillé). Nous pouvons voir cela dans la figure 9. Le champ existe tout de même dans la base de données donc la valeur sera aussi dans le résultat de la requête.

✎ Instruction + Language			
Login	Anglais	Francais	Italien
AFFICHAGE_ALEATOIRE_OBJETS	Random	Aléatoire	Random
AFFICHAGE_SYNCRO_BRAS	Syncro	Synchronisation des b	Syncro
AGE	Age	Age	Age
AJOUTER		Ajouter	Add
AJOUTER_UN_DRAPEAU	Add flag	Ajouter un drapeau	Add flag

Figure 9 : Valeur vide dans le tableau

3.2.3.2 La gestion des produits d'une expérience

La gestion des produits d'une expérience a été le deuxième point chronophage de l'administration. Le système de Drag & Drop pour l'upload de fichiers a été très complexe et difficile à mettre en œuvre.

Nous avons donc choisi de créer un plugin jQuery à part entière afin de pouvoir le réutiliser dans toute l'application si besoin. Il détecte de nombreux événements tel que "dragenter", "dragover" et "dragleave" mais celui qui nous intéresse le plus est "drop" (le fait de déposer l'élément). C'est à ce moment-là que nous lançons une fonction avec en paramètre le fichier « capté » par drop. Cette fonction va envoyer toutes les informations de l'image à un script PHP. Nous nous servons de l'objet JavaScript XMLHttpRequest pour accéder à un fichier PHP en POST, et envoyer des informations via le Header de la requête grâce à la méthode setRequestHeader(). Concrètement nous rajoutons des données dans l'en-tête que nous récupérons dans le fichier PHP avec getAllheaders() qui extrait toutes les données du header dans un tableau. Il n'y a plus qu'à extraire ces informations et les enregistrer.

Cette démarche peut paraître un peu compliquée c'est pour cela que nous n'allons pas rentrer plus en détails dans la partie technique. Les fichiers correspondants sont dropfile.js et upload.php.

Pour nous la difficulté résidait dans l'enregistrement de l'image et son passage de JavaScript à PHP. Il fallait de plus actualiser son affichage dans la zone des produits, ce dynamiquement et sans recharger la page.

Il est aussi possible d'envoyer plusieurs images en une seule fois en rappelant la fonction JavaScript d'upload autant de fois que le drop a détecté d'images.

3.3 Choix de développement

3.3.1 Structure des fichiers & maintenabilité (Annexes 1 & 2)

Nous n'avons pas jugé utile de mettre en place un développement orienté objet. La répartition de chaque fonctionnalité dans des fichiers différents aura suffi à rendre modulaire notre code. Augmentant ainsi nos critères de maintenabilité et réutilisabilité.

Par ailleurs, le fichier expérience est quant à lui très dense. En effet, il rassemble toutes les fonctionnalités qui touchent au déroulement de l'expérience. L'environnement étant développé avec la technologie WebGL qui est une technologie JavaScript il était plus aisé d'enrichir le fichier experience.php. Néanmoins, nous avons veillé à commenter l'intégralité de ses fonctionnalités (en anglais) afin de conserver une clarté de lecture et de compréhension.

3.3.2 Retour sur le WebGL

Pour ce qui est du choix de la technologie WebGL que nous utilisons au travers de la librairie ThreeJS, il s'est avéré payant. Malgré une prise en main complexe, car basée sur l'exemple, nous sommes parvenus à créer une expérience fidèle à nos attentes et maquettes. WebGL assure une fluidité et une maniabilité épatantes au sein d'un navigateur web (sur les appareils compatibles → récents). ThreeJS est donc l'outil qui nous a permis d'intégrer la technologie WebGL dans notre application.

3.3.3 Base de données

Le choix du développement prioritaire de la section administrateur et du module multilingues nous a permis de construire et structurer notre base de données pour l'ensemble du projet. Nous nous sommes basés sur le schéma conçu lors de la phase de conception et nous l'avons adapté à nos choix de développement.

C'est pourquoi, on peut constater l'ajout des données « random » et « synchroBras » dans la table expérience qui sont des paramètres de l'expérience ajoutés sur demande du client.

On remarque aussi la suppression de la table « Avatar » qui n'est plus nécessaire car son utilité s'est simplifiée par l'attribut avatar dans la table résultat. De même nous avons ajouté l'attribut position qui indique l'ordre de passage d'un produit dans une expérience. Comme indiqué lors de la phase de conception les données résultats sont plus fournies.

En bref, la table résultat a été complètement remodelée pour s'adapter à nos exigences lors du développement de l'application.

4 WEBGL

4.1 Utilisation de la librairie ThreeJS

4.1.1 Principe

ThreeJS permet de visualiser et d'interagir avec des réalisations 3D directement dans le navigateur WEB. Trois éléments sont essentiels lors de l'utilisation de cette bibliothèque :

- Une scène
- Une caméra
- Une zone de rendu

Le principe est ensuite d'ajouter des éléments à la scène et de placer la caméra pour visualiser ces éléments. On peut par exemple ajouter des Mesh, c'est-à-dire des éléments qui possèdent une géométrie (cube, sphères ou des éléments créés à partir de logiciels de modélisation 3D comme Blender, Maya) ainsi qu'un matériau pour conditionner l'apparence de l'objet. On peut également insérer des lumières. On associe ensuite la scène et la caméra pour effectuer le rendu.

4.1.2 Prise en main

La prise en main de ThreeJS n'a pas été des plus simples. En effet, nous avons trouvé que la documentation était très complexe et nécessite d'être apprivoisée. En revanche, les exemples de réalisation avec ThreeJS étaient très nombreux, nous pouvions ainsi visualiser le code pour mieux comprendre la technologie. L'apprentissage de cette technologie s'est donc basé dans un premier temps sur l'analyse de fichiers existant, puis sur la documentation.

4.1.3 Rappel des besoins de l'expérience

L'immersion du participant était un élément primordial. ThreeJS devait nous permettre de manipuler un avatar en 3 dimensions dans un environnement vis-à-vis d'un produit. La manipulation de l'avatar devait être instinctive, fluide et simple.

4.2 Les étapes de réalisation

4.2.1 Comprendre la technologie

Cette technologie étant totalement nouvelle pour nous, l'étape de prise en main était vraiment nécessaire avant de démarrer. Il fallait comprendre et refaire des exemples d'application. Les exemples étant très nombreux, nous pouvions nous appuyer sur des réalisations qui se rapprochent de ce que nous voulions faire. Il fallait faire attention néanmoins à ne pas passer notre temps à chercher de nouveaux exemples, mais plutôt apprivoiser la technologie pour parer à tous les problèmes futurs. Tout au long du projet, nous avons continué à apprendre et comprendre de nouvelles choses par rapport à l'utilisation de ThreeJS.

Nous avons donc réalisé que ThreeJS est une technologie dynamique en pleine effervescence. Elle résulte de la participation de plusieurs développeurs sur un dépôt GitHub en libre accès, ce qui explique les changements permanents. Pour éviter tout problème de compatibilité futur nous travaillons sur la révision 71 présente dans un fichier local de l'application.

4.2.2 Création de l'environnement et de l'avatar

Ces deux étapes ont pu être faites de manière simultanée puisqu'elles ne dépendaient pas immédiatement l'une de l'autre. Au niveau de la création de l'environnement, nous pensions au départ créer nos propres environnements en construisant les éléments à l'aide de ThreeJS. Il était cependant très difficile à notre niveau d'obtenir des environnements réalistes et immersifs assez rapidement. De plus, une image de fond fixe était suffisante puisque notre camera restait fixe. Nous avons donc opté pour la création d'environnements en utilisant une image de fond. Soit en récupérant une image libre de droits trouvée sur internet ou alors en concevant nous-même l'environnement avec une application de modélisation 3D comme "wanaplan".



Figure 10 : Environnement créé avec le l'application web wanaplan

Concernant la création de l'avatar, l'idée au départ était de trouver deux avatars 3D (une femme et un homme) libre de droits pour pouvoir les utiliser dans notre projet. Nous nous sommes très rapidement rendu compte que la structure des avatars (essentielle à notre développement) différée à chaque avatar obtenu sur internet. Une solution pour palier à ce problème aurait été de concevoir nous même les avatars via Blender.

N'ayant pas les connaissances de modélisation Blender nous avons cherché une autre solution. Cette recherche s'est concrétisée par la découverte d'un logiciel libre MakeHuman qui permet de générer des avatars détaillés exportables sous différent format 3D.

4.2.3 Intégration et animation de l'avatar

L'intégration de l'avatar dépendait des formats 3D supportés par ThreeJS. Ils étaient au nombre de 3 : Collada (.dae), Wavefront (.obj) et un fichier JSON issu d'un plugin de conversion Blender -> ThreeJS. Après l'essai des 3 formats nous avons sélectionné le format Collada. En effet, le convertisseur ne semblait plus à jour avec la révision 71. Le format Collada nous permettait d'importer un objet structuré (SkinnedMesh) sous ThreeJS. Soit un objet 3D rattaché à un squelette qui le rend manipulable.

L'importation étant faite, nous avons pu débiter l'animation de l'avatar. Cette étape a été divisée en 2 tâches distinctes :

- Le déplacement de l'avatar qui nécessite une animation prédéfinie réalisée sous Blender
- La manipulation de l'avatar qui nécessite une interaction de l'utilisateur avec l'environnement 3D.

La difficulté a donc été de déplacer les os du squelette de l'avatar au sein de l'application web.

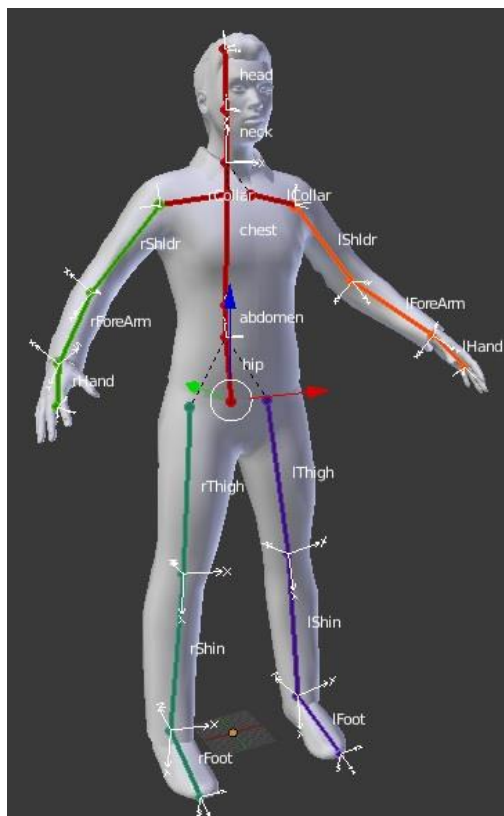


Figure 11 : Structure avatar

Afin de simplifier cette explication voici un schéma de l'avatar tel que nous le visualisons lors du développement :

Comme on peut le voir sur le schéma, chaque os est dans un repère local qui lui est propre ce qui rend plus difficile la projection dans l'espace des mouvements que nous devons effectuer.

Au niveau de ThreeJS, chaque Mesh se traduit par un objet JavaScript complexe. Nous avons donc étudié l'objet de l'avatar qui se compose :

- Materials : Les matériaux appliqués à chaque partie du corps
- THREE.Skeleton : le squelette de l'avatar
- THREE.Bones : les os qui composent le squelette (dans notre cas 19)

Afin de manipuler l'avatar nous devons simplement faire pivoter les os. Les matériaux s'adaptent à cette

rotation grâce à l'option « skinning : true » qui les compose. De plus chaque matériaux est associé (bind) au squelette modifié afin que chaque événement de modification du squelette soit pris en compte.

Ainsi nous savions comment manipuler l'avatar à l'aide des fonctions ThreeJS. Il nous fallait alors déterminer l'interface de manipulation et son lien avec l'environnement 3D créé par ThreeJS.

Pour simplifier l'apprentissage de l'interface par l'utilisateur nous avons convenu que le multitouch ralentirait la vitesse d'apprentissage. Ainsi nous avons fixé des cibles sur certains os de l'avatar, leur déplacement entrainerait la rotation de certains os. Pour la réalisation de ces cibles 2 possibilités s'offraient à nous :

- Des cibles HTML (ex : div) rattaché aux os :
 - Avantage : Interception du clic simplifiée
 - Difficulté : Association d'une div à un objet l'environnement 3D
- Des cibles 3D ThreeJS (Mesh) associées aux os :
 - Avantage : Association Cibles/Os simplifié (même environnement)
 - Difficulté : interception du clic utilisateur provenant d'un environnement extérieur 2D

Dans les deux cas nous devons mettre en place une fonction permettant de passer des coordonnées de l'environnement 3D de ThreeJS à l'environnement 2D de la fenêtre du navigateur ou inversement.

Pour pallier à cette difficulté nous avons créé 2 fonctions :



Figure 12 : Cibles de manipulation

- `worldToScreen(coordonnees)` : Pour passer de la 3D à la 2D.
- `mouseToWorld(evt)` : Pour passer de la 2D à la 3D.

Les fonctions `unproject()` et `project()` de ThreeJS nous permettaient cette conversion une fois les vecteurs de position normalisés. Elles prennent en compte le point de vu de la camera. De plus un résultat « intersects » nous permettait d'établir si une coordonnée 2D rentrait en collision avec un/plusieurs objet(s) de l'environnement 3D.

Afin d'optimiser l'affichage et la fluidité des mouvements nous avons choisi l'option de créer des cibles (sphères) à l'intérieur de l'environnement 3D que l'on retrouve dans la partie tutoriel de l'expérience.

A partir de ce moment, tous les clics sur l'environnement 3D sont traités. Si on clique sur une cible on effectue une rotation d'un des os ou de l'avatar (suivant la cible), si on clique en dehors des cibles l'interface écoute alors un déplacement de l'avatar. Il est important de signaler que sur demande du client nous avons intégré dans le menu gestion l'option « synchronisation des bras ». Cela permet de manipuler chaque bras indépendamment ou simultanément pendant l'expérience.

Toujours dans l'optique de conserver un temps d'apprentissage convenable tout en obtenant un maximum de données nous avons limité la manipulation à 3 rotations :

- Le buste avec la cible située sur la tête
- Les bras que l'on peut faire pivoter sur 2 axes suivant si on est de face (et de dos) ou de profil avec les cibles situées au niveau des poignets.
- L'avatar avec une cible situé entre ses pieds.

Nous avons convenu de limiter les rotations de l'avatar à des limites humaines. En outre, La position de l'avatar dans l'environnement nécessite à chaque manipulation de calculer si le mouvement est possible. En effet les rotations et calculs d'angles s'inversent suivant la position de la souris et les valeurs de rotations de l'avatar. Ce fut un point complexe du développement car nous devons garder une interaction intuitive tout en modifiant les repères de l'avatar.

Voici les détails des rotations fournis avec le manuel pour la compréhension des données extraites :

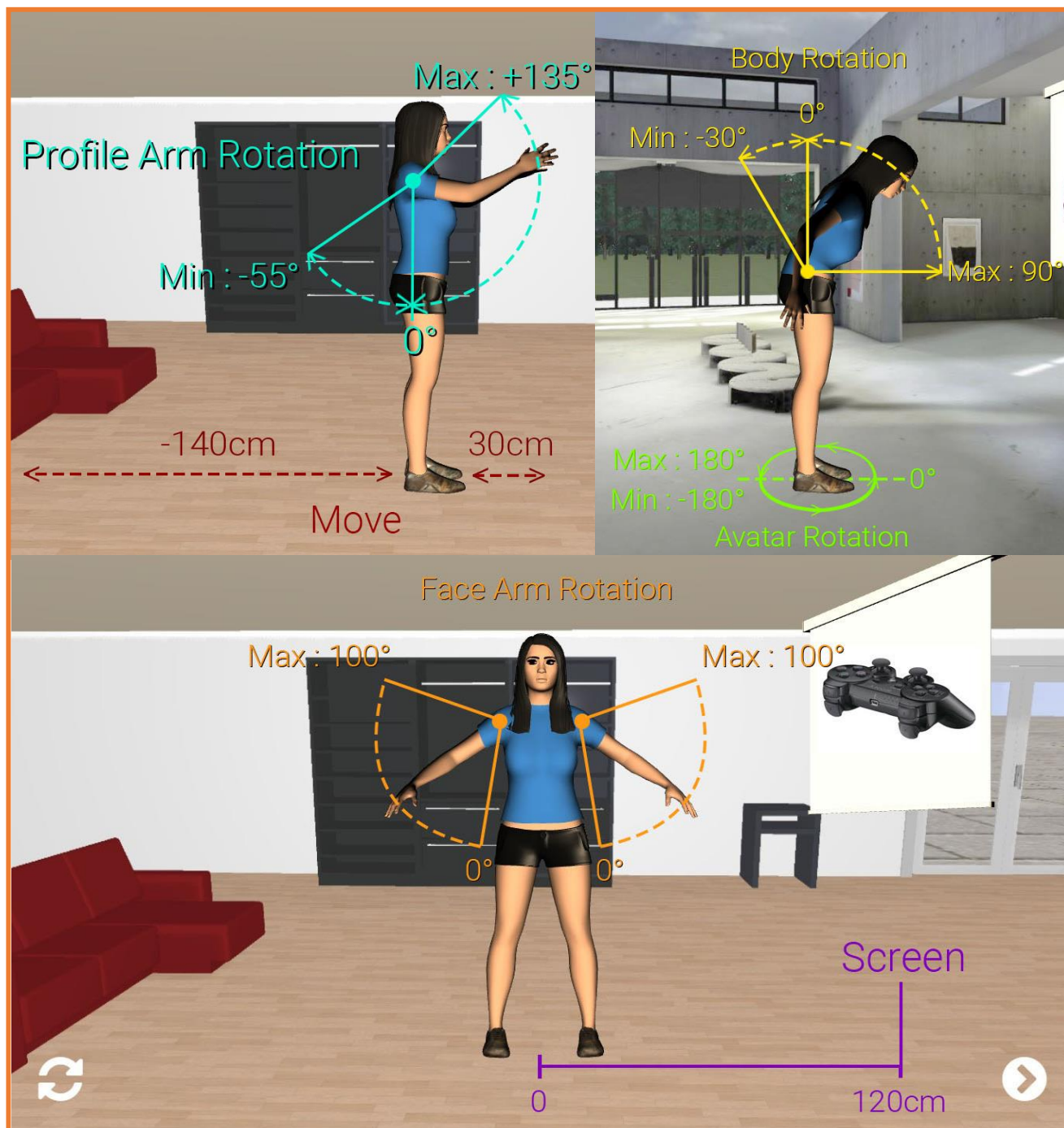


Figure 13 : Limites de mouvement

Pour faciliter la prise en main de l'application nous avons mis en place un tutoriel interactif optionnel superposé à la réalisation d'une expérience (pour poser un contexte familier). On explique et demande au participant de réaliser chaque mouvement les uns à la suite des autres dans un contexte de manipulation restreint (rotations non demandées bloquées).

La seconde partie de l'animation correspond au déplacement de l'avatar en avant ou en arrière. Comme indiqué ci-dessus, l'animation se lance lors d'un « slide » (cliqué-glissé) où

le début du slide est en dehors d'une cible permettant de manipuler la posture de l'avatar. Pour une utilisation agréable l'animation se lance uniquement quand le « slide » est assez grand ce qui évite que l'avatar avance lors d'un clic par inadvertance sur l'environnement.

La création de cette animation de marche a été faite en dehors de ThreeJS avec le logiciel Blender et elle fait partie intégrante de l'avatar.

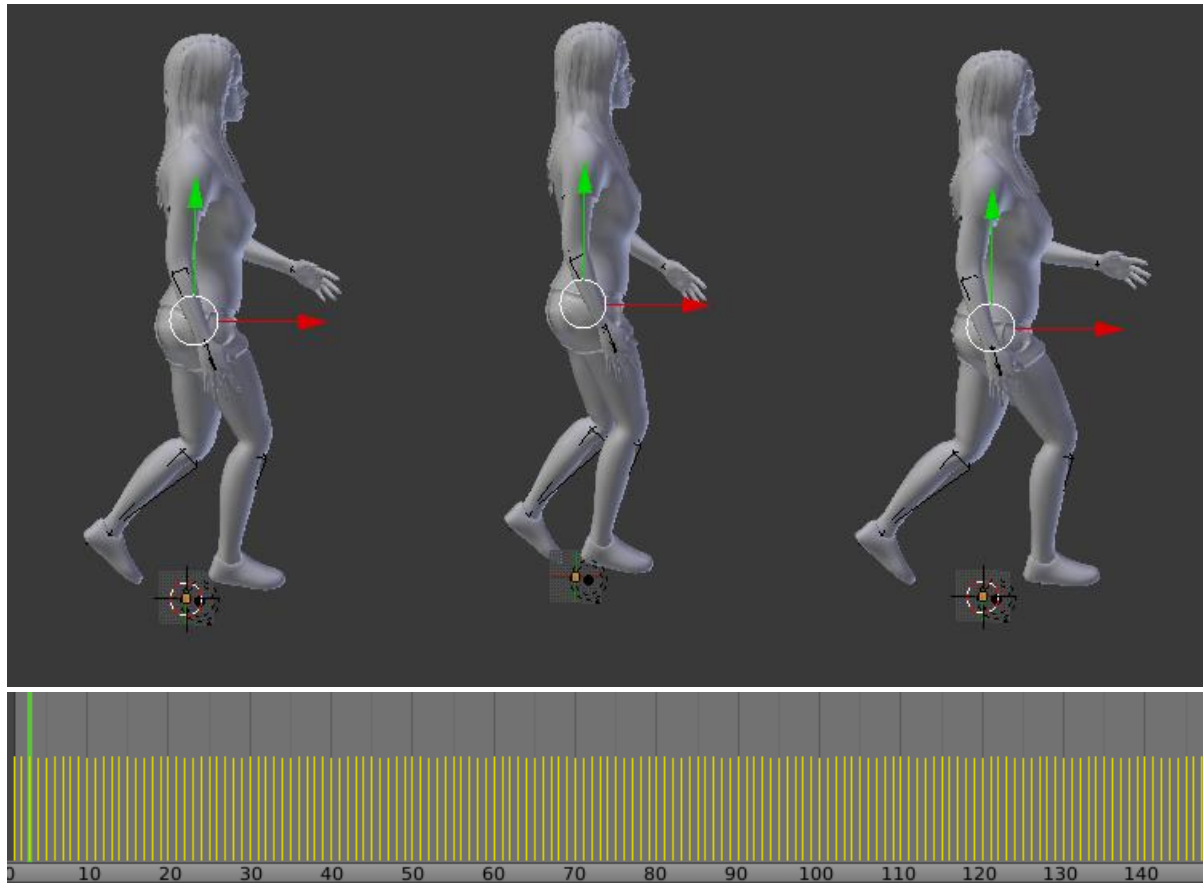


Figure 14 : Animation sous Blender

Le bas de la figure 14 montre la ligne de temps où chaque trait correspond à une image de l'avatar. Pour modifier cette image on effectue des rotations sur les différentes parties de son corps. L'animation représente l'enchaînement de ces différentes postures. Cela est semblable au principe du dessin animé. Seule l'animation des différents membres de l'avatar est gérée dans Blender, nous avons décidé de réaliser le déplacement global directement avec ThreeJS pour plus de simplicité.

Avant chaque déplacement, l'avatar se met de profil pour marcher vers l'avant ou à reculons suivant les différents cas :

- S'il est plutôt face au produit (Rotation de l'avatar entre -90° et 90°)
 - S'il se rapproche du produit : Il se met complètement face au produit et il marche vers l'avant.

- S'il s'éloigne du produit : Il se met complètement face au produit et marche à reculons
- S'il est plutôt dos au produit (Rotation de l'avatar entre 90° et 180° ou -180° et -90°)
 - S'il se rapproche du produit : Il se tourne complètement face au produit et il marche vers l'avant
 - S'il s'éloigne du produit : Il se met complètement de dos au produit et marche à vers l'avant

Ces choix de déplacement résultent de la première réunion client. Le fait que l'avatar puisse marcher à reculons tout en regardant l'objet a été ajouté.

La distance de déplacement est calculée à la fin du « slide » est correspond à la distance de ce « slide » tant que les bornes ne sont pas dépassées, sinon l'avatar se déplace jusqu'à la borne. Comme on peut le voir dans la figure 13, l'avatar peut se déplacer de 30 cm vers l'avant et de 140 cm vers l'arrière depuis sa position initiale. Comme expliqué précédemment, pour connaître la distance du « slide » en cm dans l'environnement 3D, on utilise la fonction `mouseToWorld(evt)`. Pour avoir un effet de déplacement et non de « téléportation » on effectue une très petite translation de l'avatar un grand nombre de fois tout en lançant l'animation créée sous Blender.

4.2.4 Création et enregistrement des données

Les données de l'expérience sont créées dans une variable « data » au format JSON. Cette variable possède les données pour chaque produit de l'expérience.

Pour chaque produit, on aura donc des informations sur :

- le produit (Identifiant, position dans l'expérience)
- la posture de l'avatar (rotation des bras, du buste, de l'avatar)
- la position de l'avatar

On a également dans cette variable l'identifiant de l'expérience ainsi que le sexe de l'avatar.

A la fin de l'expérience, cette variable de données est envoyée à la page de finalisation. Le participant indique ensuite ses données (sexe et date de naissance au minimum). Les informations sont alors insérées dans la base de données.

5 Manuel utilisateur

Afin de rendre l'utilisation de cette application disponible à tous et suite à une demande explicite du client nous avons réalisé un manuel d'installation, d'utilisation et d'exploitation de l'application. Il est disponible en version française et anglaise comme l'ensemble des éléments de ce projet. Vous pouvez retrouver l'application et sa documentation dans le dépôt GitHub suivant : [Emolyse](https://github.com/Emolyse/Emolyse) (<https://github.com/Emolyse/Emolyse>).

6 Améliorations possibles

Dans tout projet web il y a toujours de nouvelles idées qui peuvent survenir après la fin du développement. C'est pourquoi nous proposons ici une petite liste non exhaustive de nos idées pour améliorer ou compléter l'application :

- Permettre l'ajout de nouveaux environnements. Nous ne l'avons pas mis en œuvre maintenant car il faudrait être sûr que l'angle de la caméra de l'environnement soit le même que celui de l'écran.
- Permettre l'ajout de nouveaux avatars. Cela implique d'en créer avec le logiciel libre MakeHuman et de l'importer dans l'application. Cela peut être un peu compliqué pour un novice.
- Intégrer un outil de traitement des données (graphiques, tableaux etc.) récoltées directement dans l'application via un nouveau menu pour l'expérimentateur.

7 Conclusion

Pour conclure sur le projet Emolyse, nous avons été heureux de pouvoir effectuer ce travail en collaboration avec Anna Tcherkassof et Damien Dupré. En effet, la partie relation client nous a vraiment confronté aux réalités du monde du travail. De plus, effectuer ce projet sous forme de stage dans les locaux de Domus nous a mis dans des conditions de travail optimales.

L'application est totalement terminée et validée par la partie cliente. Une copie est disponible en téléchargement libre sur [github](#) avec une documentation d'installation pour être accessible à tous. Dès le début du projet, l'application était destinée à être open source. Une version de démonstration va très prochainement être mise en ligne dans le but d'être présentée lors de nos futurs entretiens professionnels.

L'apprentissage d'une nouvelle technologie, telle que ThreeJS utilisée dans le projet Emolyse, est toujours quelque chose de très plaisant. Le projet ayant été terminé dans les délais, ceci est également une grande satisfaction pour nous.

Webographie

Information sur le webGL

- Le fonctionnement de WebGL :
<http://www.scriptol.fr/programmation/webgl.php>
- Tutoriel d'introduction :
https://developer.mozilla.org/fr/docs/Web/WebGL/Commencer_avec_WebGL
- Les cartes graphiques compatibles :
<https://www.khronos.org/webgl/wiki/BlacklistsAndWhitelists>

La bibliothèque ThreeJS

- Site officiel de la librairie :
<http://threejs.org/>
- Tutoriels d'introduction à ThreeJS :
 - <http://markable.in/file/b06227b4-f52d-11e2-a9c9-984be164924a/>
 - <http://www.alsacreations.com/tuto/lire/1572-webgl-3d-three-canvas-threejs.html>
 - <http://lsc.univ-evry.fr/~didier/home/lib/exe/fetch.php?media=cours:ig:threejs.pdf>
- Livre numérique d'apprentissage de ThreeJS :
<https://books.google.fr/books?id=GvpzBgAAQBAJ>

Outils de modélisation 3D

- Modèle 3D libres de droits :
<http://archive3d.net/>
- Outil de conception de plans 3D
https://fr.wanaplan.com/try_plan
- Aide pour le logiciel blender
<http://wiki.blender.org/>
- Création d'animation sous blender
 - <http://www.kadrmascconcepts.com/blog/2012/01/24/from-blender-to-threefab-exporting-three-js-morph-animations/>
 - <https://devmatrix.wordpress.com/2013/02/27/creating-skeletal-animation-in-blender-and-exporting-it-to-three-js/>

Outils d'aide au développement frontend

- Documentation jQuery
<http://jquery.developpeur-web2.com/documentation.php>
- Tri jQuery

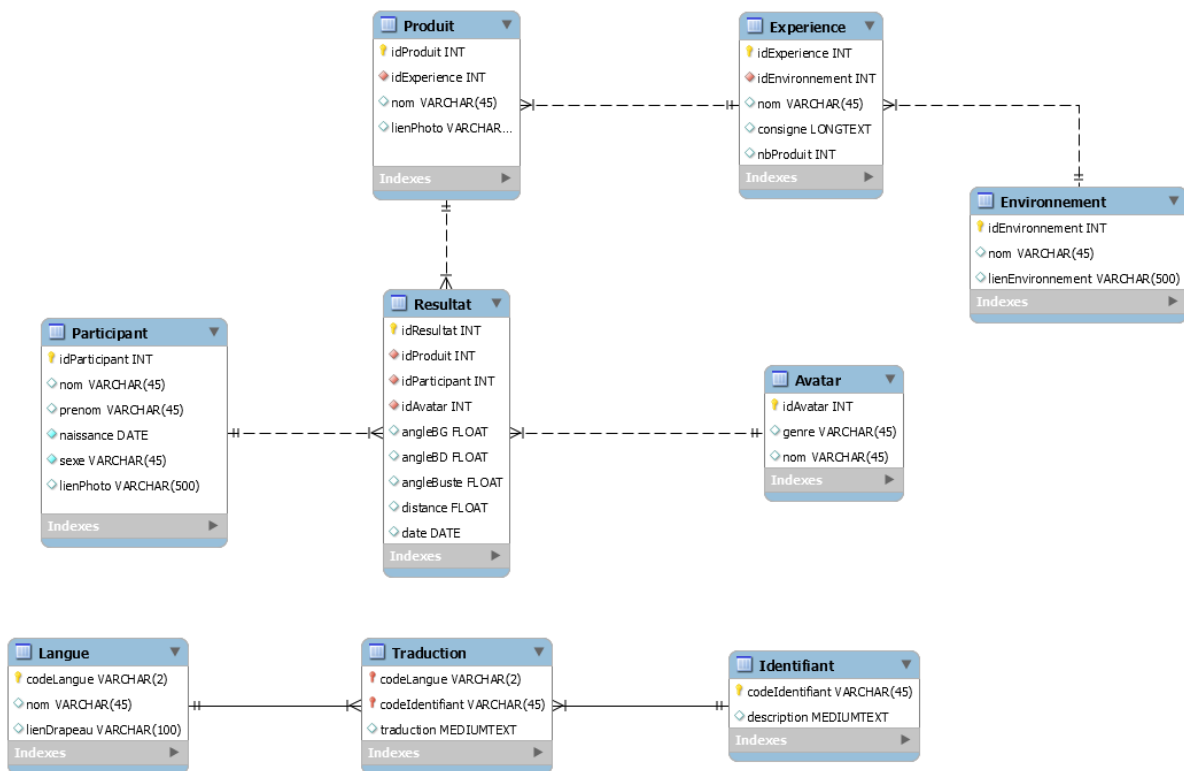
- <https://jqueryui.com/sortable/>
 - <http://stygianvision.net/updates/jquery-sortable-remove-item/>
- Enregistrer une image via Drag & Drop
 - <http://www.grafikart.fr/tutoriels/jquery/upload-drop-172>
 - <http://www.maximechailou.com/simple-upload-en-drag-and-drop-avec-html5-jquery-php/>
- Icône de chargement animée
<http://projects.lukehaas.me/css-loaders/>

Sites d'entraide généralistes

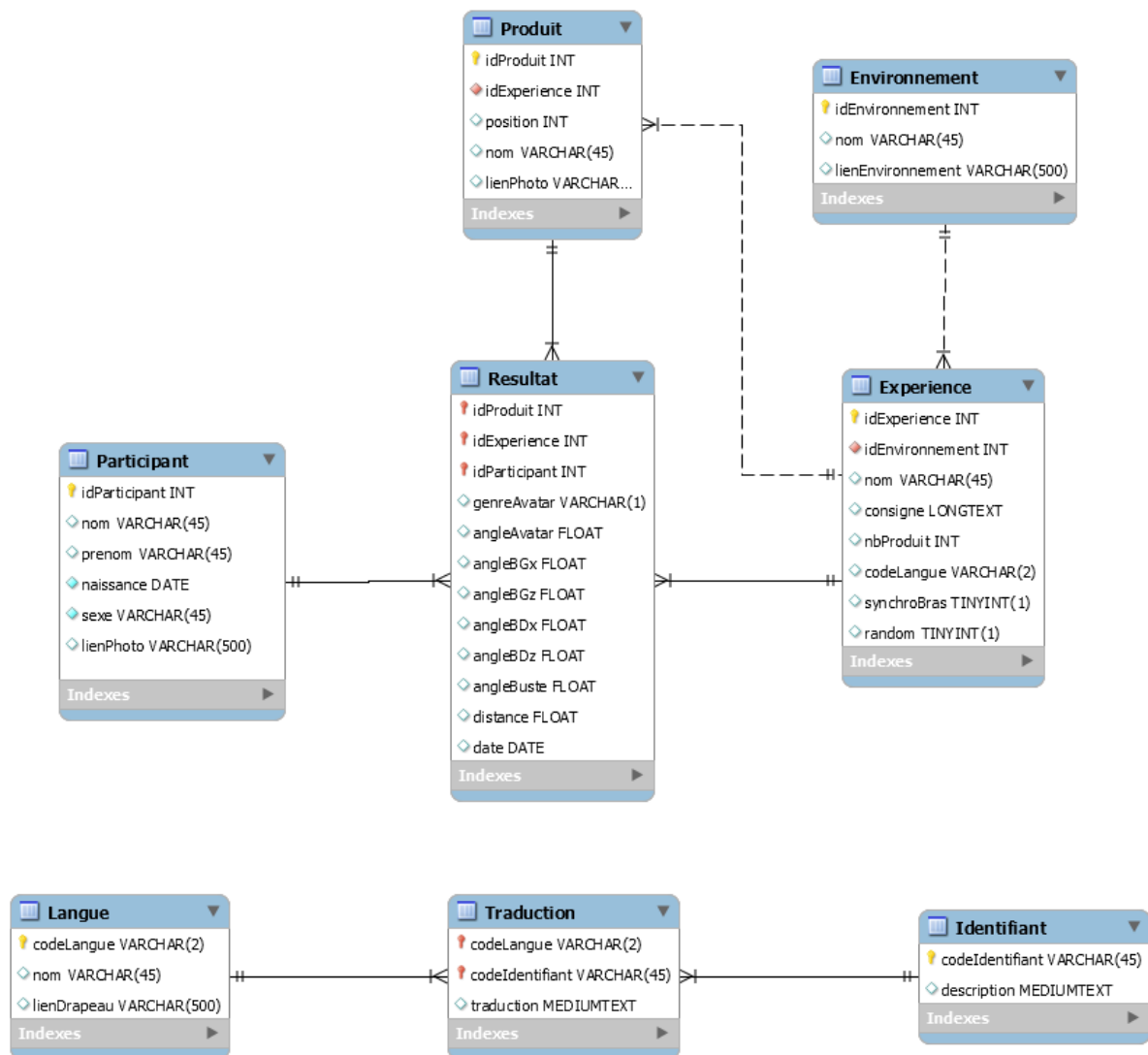
- <http://stackoverflow.com/>
- <http://www.commentcamarche.net/>
- <http://jsfiddle.net/>
- <http://openclassrooms.com/>

Annexes

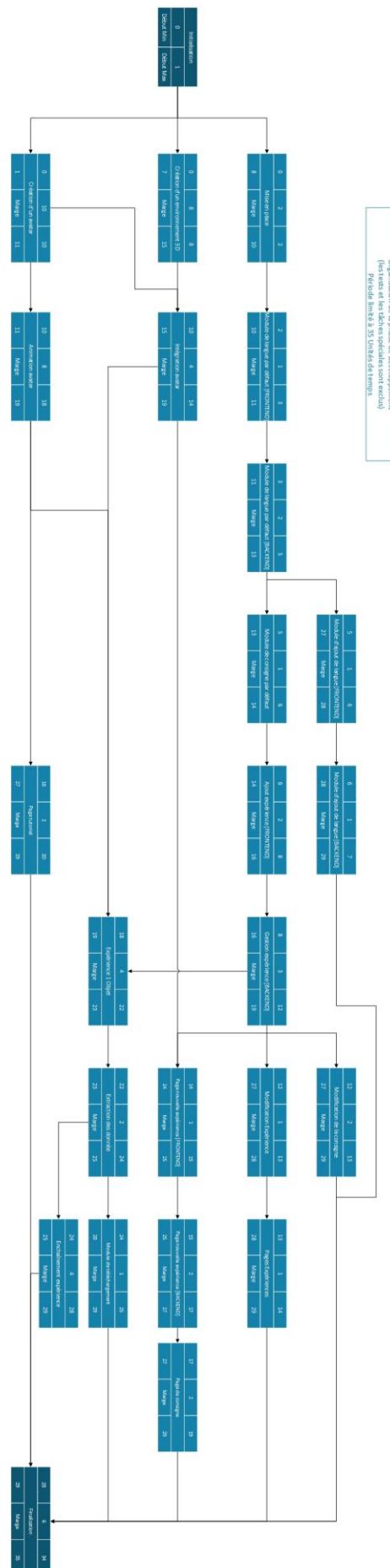
Annexe 1 : Base de données Conception



Annexe 2 : Base de données Développement



Annexe 3 : Diagramme de Pert



Annexe 4 : Cas d'utilisation

