

NOM de l'École :

Ecole Polytechnique de l'université de Nantes

RAPPORT INTERMÉDIAIRE V3

Réalisé Par :

Mohamed JAOUADA

Ghassen JRAD

Mehdi BEN SALHA

Classe et Promotion : 4^{ème} Année Informatique

Conception et implémentation d'une plateforme de gestion de formulaires compatible avec Tobii Pro Lab

Tuteurs de l'entreprise : Monsieur Matthieu PERREIRA DA SILVA et Monsieur
François BOUFFARD

Tuteur de l'école : Monsieur Olivier AUBERT

Entreprise commanditaire : L'Experience Lab (XP-LAB) de la Halle 6 Ouest



Date de rédaction : 29/04/2025

Sommaire

1. Présentation de l'entreprise et du sujet.....	7
1.1. Présentation de l'entreprise.....	7
1.2. Présentation du sujet.....	7
1.3. Le brief.....	8
1.3.1. Organisation du projet et répartition des rôles durant celui-ci :.....	8
1.3.2. L'hexamètre quintilien du projet (QQOQQP) :.....	10
2. Expression des besoins et planning.....	11
2.1. Périmètre du projet.....	11
2.2. Besoins fonctionnels.....	12
2.2.1. Description des acteurs / utilisateurs.....	12
2.2.2. Liste des besoins.....	13
2.3. Besoins non fonctionnels.....	14
2.4. Évolution potentielle des besoins.....	15
2.5. Les risques.....	16
2.5.1. Liste des risques du projet.....	16
2.5.2. Politique de gestion des risques.....	18
2.6. Plannings.....	18
2.6.1. Planning réalisé.....	18
3. Travail réalisé.....	22
3.1. Première et deuxième itération.....	22
3.1.1. L'architecture choisie pour la plateforme à implémenter.....	22
3.1.2. La conception UML.....	23
3.1.2.1. Diagramme de cas d'utilisation.....	23
3.1.2.2. Diagramme de classes.....	24
3.1.3. Les scénarios d'utilisation.....	25
3.1.4. Prototype IHM.....	32
3.1.4.1. L'interface du participant.....	32
3.1.4.2. L'interface de l'expérimentateur.....	33
3.1.5. Choix des technologies et librairies à utiliser.....	36
3.1.5.1. Front-end : React avec la librairie Form.js.....	36
3.1.5.2. Back-end : Express.js.....	37
3.1.5.3. Base de Données : SQLite.....	37
3.2. Troisième et quatrième itération.....	38
3.2.1. Implémentation.....	38
3.2.1.1. La base de données.....	38
3.2.1.2. Le back-end.....	40

3.2.1.3. Le front-end.....	42
3.2.2. L'intégration avec Tobii Pro Lab.....	53
4. Conclusion.....	55
5. Bibliographie scientifique et technologique.....	56
5.1. Bibliographie scientifique.....	56
5.2. Bibliographie technologique.....	58
5.3. Références bibliographiques complètes.....	60

Liste des tableaux

Tableau 1 : Liste hiérarchisée des besoins fonctionnels selon le critère MoSCoW.....	13
Tableau 2 : Liste des risques identifiés avec leurs gravités, probabilités d'occurrence et criticités.....	17
Tableau 3 : Tableau « probabilité-conséquences».....	17
Tableau 4 : Les itérations réalisées.....	18
Tableau 5 : Description textuelle du scénario d'utilisation : Créer un formulaire.....	25
Tableau 6 : Description textuelle du scénario d'utilisation : Modifier un formulaire.....	27
Tableau 7 : Description textuelle du scénario d'utilisation : Supprimer un formulaire.....	27
Tableau 8 : Description textuelle du scénario d'utilisation : Dupliquer un formulaire.....	28
Tableau 9 : Description textuelle du scénario d'utilisation : Consulter un formulaire.....	28
Tableau 10 : Description textuelle du scénario d'utilisation : Consulter les réponses des participant à un formulaire.....	29
Tableau 11 : Description textuelle du scénario d'utilisation : Remplir un formulaire.....	30
Tableau 12 : Description textuelle du scénario d'utilisation : Définir un id participant par défaut	31

Table des figures

Figure 1 : hexamètre quintilien du projet.....	10
Figure 2 : Le modèle du domaine du projet.....	11
Figure 3 : L'architecture logicielle simplifiée de la solution implémentée.....	23
Figure 4 : Diagramme de cas d'utilisation de la solution.....	24
Figure 5 : Diagramme de classes de la solution.....	25
Figure 6 : Le prototype de l'interface du participant.....	32
Figure 7 : Le prototype de l'interface de l'expérimentateur (la page d'accueil).....	33
Figure 8 : Le prototype de l'interface de l'expérimentateur (la page de consultation d'un formulaire).....	34
Figure 9 : Le prototype de l'interface de l'expérimentateur (la page des réponses d'un formulaire).....	34
Figure 10 : Le prototype de l'interface de l'expérimentateur (la page de création de formulaires avant l'ajout de widgets).....	35
Figure 11 : Le prototype de l'interface de l'expérimentateur (la page de création de formulaires après l'ajout de widgets).....	35
Figure 12 : Le prototype de l'interface de l'expérimentateur (la page de création de formulaires : explication de la mise en page, les conteneurs et l'affichage conditionnel).....	36
Figure 13 : Diagramme entité-relation de la base de données de la plateforme.....	39
Figure 14 : Page d'accueil de la plateforme.....	43
Figure 15 : Page de création de formulaires.....	44
Figure 16 : Le reste des composants proposés dans la page de création de formulaires.....	45
Figure 17 : Page de création de formulaires (un composant sélectionné).....	45

Figure 18 : Page de modification de formulaires.....	46
Figure 19 : Pop-up affiché lorsqu'on essaye de modifier un formulaire déjà rempli par un participant.....	46
Figure 20 : Page de consultation de formulaires, vue de l'expérimentateur, (page 1 du formulaire).....	48
Figure 21 : Page de consultation de formulaires, vue de l'expérimentateur, (page 2 du formulaire).....	48
Figure 22 : Page de consultation de formulaires, vue du participant, (avec option de navigation).....	49
Figure 23 : Page de consultation de formulaires, vue du participant, (sans option de navigation).....	50
Figure 24 : Page de consultation des réponses à un formulaire.....	50
Figure 25 : Page de consultation des réponses à un formulaire (tableau des réponses téléchargé).....	51
Figure 26 : Page d'accueil contenant un formulaire dupliqué.....	52
Figure 27 : Page d'accueil, l'alerte affichée suite au choix de supprimer un formulaire.....	52
Figure 28 : Page d'édition des scénarios d'expérimentation sur Tobii Pro Lab.....	53
Figure 29 : Page de l'ajout des participants et lancement du scénario sur Tobii Pro Lab.....	54
Figure 30 : Page des enregistrement des passages des participants sur Tobii Pro Lab.....	54

Introduction générale

Dans cette nouvelle version du rapport, nous avons apporté les modifications demandées lors de la soutenance de la phase 3A, notamment en remplaçant les diagrammes de séquence par des descriptions textuelles décrivant les différents scénarios d'utilisation pour clarifier leur déroulement. Suite à une réunion avec le client, nous avons mis à jour la liste des besoins fonctionnels afin de prioriser les fonctionnalités les plus importantes et garantir leur réalisation lors de la quatrième itération.

La partie "Planning réalisé" contient un tableau récapitulant les tâches effectuées lors des itérations 3 et 4 durant la phase 3B du projet. Ce tableau inclut les dates de début et de fin de chaque tâche, le temps passé sur chaque tâche, le membre de l'équipe responsable, ainsi qu'un commentaire détaillant ce qui a été accompli.

Le reste du rapport est consacré à présenter le travail effectué. Nous commençons par un rappel des activités réalisées durant la phase 3A, notamment l'architecture logicielle de la plateforme, la conception UML, le prototype IHM, ainsi que le choix des technologies et bibliothèques utilisées.

Ensuite, nous abordons ce qui a été réalisé durant la phase 3B, incluant les troisième et quatrième itérations pendant lesquelles la plateforme a été implémentée. Cette partie couvre la structure de la base de données, la partie back-end de la plateforme (y compris les routes API), la partie front-end avec la présentation des différentes pages de la plateforme, leurs fonctionnements et les fonctionnalités qu'elles offrent aux utilisateurs. Nous discutons également de l'intégration de notre plateforme avec l'outil Tobii Pro Lab [\[1\]](#).

Enfin, vous trouverez une bibliographie scientifique et technologique regroupant des références à des ouvrages et sites internet expliquant certaines notions abordées dans le rapport.

1. Présentation de l'entreprise et du sujet

1.1. Présentation de l'entreprise

La **Halle 6 Ouest**, située au cœur du Quartier de la Création sur l'île de Nantes, est un espace unique dédié à l'innovation et à la recherche. Elle regroupe divers acteurs tels que des chercheurs, enseignants, étudiants, entrepreneurs, artistes, et ingénieurs. Ce lieu ambitionne de répondre aux enjeux sociétaux de demain en proposant une approche collaborative et interdisciplinaire.

La Halle 6 Ouest abrite trois laboratoires innovants, offrant un accès à des technologies de pointe pour des domaines aussi variés que le prototypage, l'évaluation de l'expérience utilisateur, ou encore la fabrication industrielle. Ces laboratoires soutiennent à la fois la recherche, l'enseignement et l'entrepreneuriat, avec des startups installées sur place, favorisant ainsi les synergies entre les différents acteurs.

En outre, la Halle 6 Ouest se distingue par son engagement dans le développement de pédagogies alternatives. Elle accueille des formations de pointe comme le Master Cultures Numériques et plusieurs licences professionnelles, offrant aux étudiants des conditions d'apprentissage privilégiées. De plus, cet espace est un hub pour des événements, des collaborations avec le monde socio-économique, et la promotion de projets innovants à travers divers événements locaux et internationaux.

Créée en Septembre 2019, l'**Experience Lab (XP-LAB)** de la Halle 6 Ouest est une plateforme dédiée à l'évaluation de l'expérience utilisateur et à l'étude des comportements. Il s'agit d'un **UserLab** qui permet aux chercheurs et aux organisations (publiques ou privées) de mieux comprendre et anticiper les comportements des usagers, clients ou utilisateurs de services, produits ou espaces.

Cette plateforme se distingue par ses expertises scientifiques en méthodologies expérimentales et en analyse de données quantitatives (oculométrie, électrophysiologie, etc.) et qualitatives (interviews). L'Expérience Lab est aussi équipé d'une technologie de pointe, incluant des oculomètres, des capteurs physiologiques, des équipements de réalité virtuelle (VR), et des systèmes de capture de mouvement. Ces outils permettent de mener des recherches poussées sur les interactions humain/machine, l'amélioration de la qualité des interfaces, ou encore l'aide au diagnostic médical.

1.2. Présentation du sujet

Lors des évaluations de l'expérience utilisateur et des études comportementales, les expérimentateurs du XP-LAB ont besoin de recueillir, entre autres, des données auprès des participants en leur demandant de remplir des formulaires pendant l'expérience. Actuellement, le XP-LAB dispose de trois façons de présenter des formulaires à ses participants, toutes imparfaites:

- **Des formulaires en ligne via l'outil SphinxOnline.** Ces formulaires peuvent être présentés sur une tablette séparée et dans ce cas leur synchronisation avec le reste de l'expérimentation repose sur le bon suivi des instructions par le participant. Ils

peuvent également être affichés sous forme de pages web intégrée à certaines expérimentations, mais il est alors impossible d'utiliser le même formulaire pour poser différentes questions à différents moments de l'expérimentation.

- **Des formulaires factices sous forme d'image.** Ils sont parfois utilisés dans des expérimentations d'oculométrie nécessitant des réponses très simples (appui sur un bouton). Cette solution est cependant très limitée et ne propose aucun retour à l'utilisateur concernant les réponses saisies.
- **Des formulaires proposés par OpenSesame [2].** Ces formulaires sont plus intéressants que les deux solutions précédentes mais possèdent un certain nombre de limitations. Par exemple : nécessité de déclarer les formulaires à l'aide d'un langage de script dédié (ou de python), lenteur lorsque de nombreux éléments sont affichés, limitations en matière de mise en page, de validation et de personnalisation graphique.

Ce projet a pour objectif de simplifier la création de formulaires plus complexes, adaptés aux besoins spécifiques des chercheurs. La solution proposée offrira des fonctionnalités principalement liées à l'édition graphique, la mise en page intuitive, et la validation des réponses, tout en étant adaptée à l'environnement spécifique de l'XP-LAB. L'objectif est d'optimiser le processus de collecte de données, en le rendant plus efficace et convivial pour les chercheurs ainsi que pour les participants.

1.3. Le brief

Dans cette partie on abordera quelques éléments essentiels du brief notamment l'organisation du projet et la répartition des rôles durant celui-ci ainsi que l'hexamètre quintilien qui résumerait tout ce qui a été dit avec le client.

1.3.1. Organisation du projet et répartition des rôles durant celui-ci :

La partie MOA (client) : XP-LAB de Nantes Université

Les membres de l'MOA :

- **Francois Bouffard**
 - RÔLE: Responsable opérationnel du UserLab
 - MAIL: Francois.Bouffard@univ-nantes.fr
 - MOBILE : 0253008004
- **Matthieu Perreira Da Silva**
 - RÔLE: Responsable scientifique du UserLab
 - MAIL : Matthieu.Perreiradasilva@univ-nantes.fr
 - MOBILE : 0240683060

Les engagements de l'MOA :

- La spécification des besoins fonctionnels et non fonctionnels.
- Mettre à disposition le matériel et les logiciels nécessitant une licence.
- Répondre aux questions des membres de la MOE.
- Fournir des exemples de formulaires et d'expériences réalisées.

La partie MOE : Équipe Projet transversal

Les membres de l'MOE :

o **Olivier AUBERT**

- RÔLE: Tuteur pédagogique
- MAIL : Olivier.Aubert@univ-nantes.fr
- MOBILE : 0240683146

o **Ghassen JRAD**

- RÔLE: Responsable livrable / Développeur
- MAIL : ghassen.jrad@etu.univ-nantes.fr
- MOBILE: 0753164281

o **Mehdi BEN SALHA**

- RÔLE: Responsable technique / Développeur
- MAIL : mehdi.ben-salha@etu.univ-nantes.fr
- MOBILE : 0771411903

o **Mohamed JAOUADA**

- RÔLE: Chef de projet / Développeur
- MAIL : mohamed.jaouada@etu.univ-nantes.fr
- MOBILE : 0620081262

Les engagements de l'MOE :

- Proposition des idées d'implémentation, des étapes, et de l'architecture technique de la solution.
- La gestion du projet (assurer la bonne conduite du projet).
- Envoi des livrables dans les délais.

1.3.2. L'hexamètre quintilien du projet (QQOQQP) :

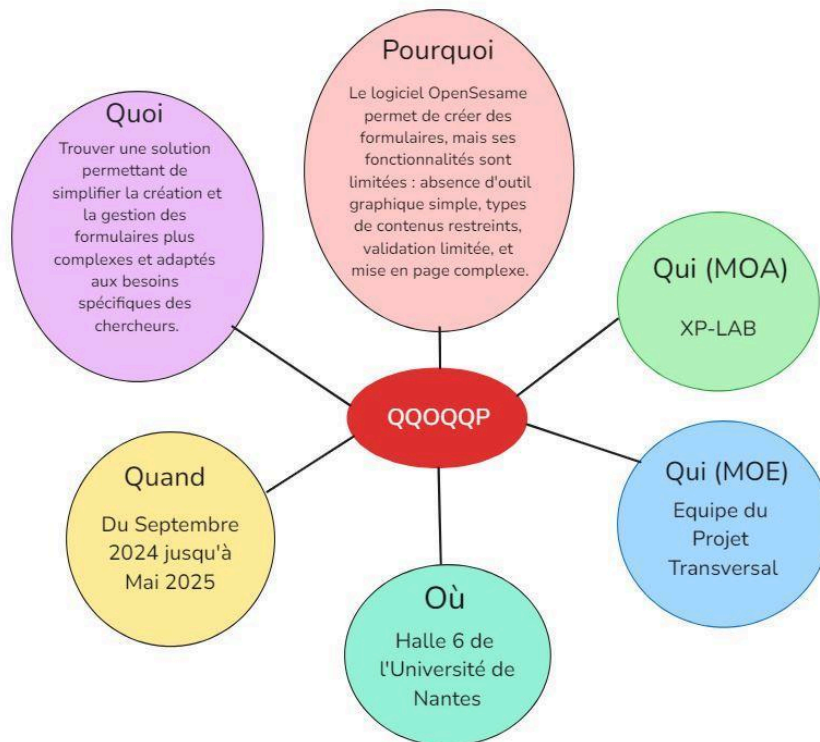


Figure 1 : hexamètre quintilien du projet

2. Expression des besoins et planning

2.1. Périmètre du projet

Afin de représenter les concepts significatifs du monde réel qui sont en relation avec le projet ainsi que son périmètre, on a élaboré le modèle du domaine présenté par la figure 2 ci-dessous.

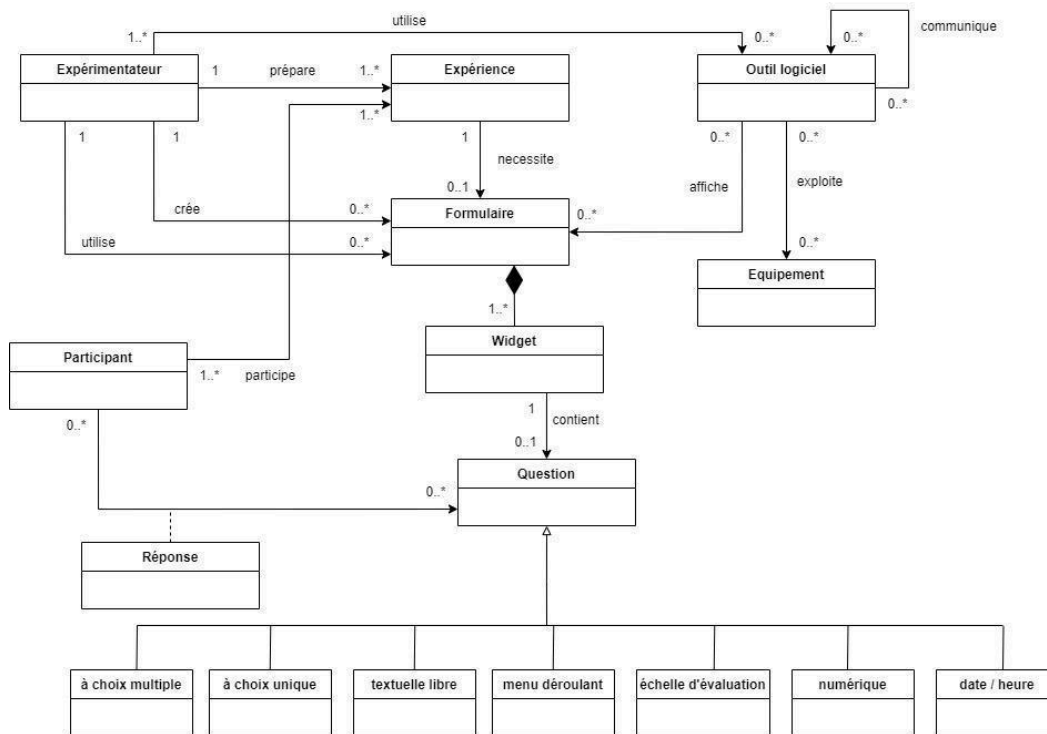


Figure 2 : Le modèle du domaine du projet

Définitions et Vocabulaire :

- **Expérimentateur** : Une personne qui se livre à des expériences scientifiques pour vérifier ou préciser des hypothèses. Dans notre projet, les expérimentateurs concernés mènent des recherches sur les interactions humain/machine, l'amélioration de la qualité des interfaces, ou encore l'aide au diagnostic médical.
- **Expérience** : Fait de provoquer un phénomène dans l'intention de l'étudier, de l'observer, de contrôler une hypothèse. Synonymes : épreuve, essai, expérimentation.
- **Participant** : La personne qui va participer à une expérience, son comportement sera observé et étudié par l'expérimentateur.
- **Formulaire** : Avant, durant ou suite à l'achèvement de l'expérience, le participant peut être amené à remplir un formulaire. Ses réponses vont être récupérées par l'expérimentateur et vont lui permettre d'acquérir plus d'informations sur son avis et ses pensées.
L'expérimentateur doit pouvoir créer et mettre en page des formulaires puis les utiliser lors de ses expériences.
- **Widget** : Les formulaires sont en réalité composés de widgets que l'expérimentateur peut ajouter un par un pour constituer le formulaire.

- **Question** : Un widget peut contenir une question que l'expérimentateur souhaite ajouter au formulaire. Une question peut être de type spécifique, notamment : à choix multiple (checkbox), à choix unique (radio), textuelle libre, menu déroulant, échelle d'évaluation, numérique ou date/heure.
- **Outil logiciel** : Les expérimentateurs utilisent divers outils logiciels exploitant les équipements, pouvant potentiellement afficher les formulaires et communiquer entre eux.
- **Équipement** : Dans certaines expériences on peut utiliser différents équipements incluant des oculomètres, des capteurs physiologiques, des équipements de réalité virtuelle (VR), et des systèmes de capture de mouvement.

2.2. Besoins fonctionnels

2.2.1. Description des acteurs / utilisateurs

Le projet vise à simplifier la création et la gestion de formulaires complexes pour les expériences menées au XP-LAB. La solution proposée offrira des fonctionnalités principalement liées à l'édition graphique, la mise en page avancée, et la validation des réponses, tout en étant adaptée à l'environnement spécifique de l'XP-LAB notamment en évitant toute interférence avec le fonctionnement des équipements, tels que ceux utilisés pour l'eye-tracking [3], durant l'expérience. L'objectif est d'optimiser le processus de collecte de données, en le rendant plus efficace et convivial pour les chercheurs ainsi que pour les participants.

Les deux principaux utilisateurs de la solution sont :

- **L'expérimentateur:**

L'expérimentateur conçoit les étapes de l'expérience qui peuvent inclure le remplissage d'un formulaire.

L'expérimentateur peut alors créer, personnaliser et mettre en page des formulaires destinés aux participants. En ajoutant des widgets (questions avec les champs de réponse, images) un par un, il peut structurer les formulaires selon les besoins de l'expérience, intégrant divers types de questions (choix multiples, texte libre, etc.).

L'expérimentateur peut aussi récupérer les réponses des participants aux formulaires, permettant ainsi d'obtenir des données qualitatives sur leurs réactions, opinions et ressentis.

- **Le participant:**

Le participant prend part à l'expérience et répond aux questions du formulaire s'il y en a.

Il donne des réponses différentes selon les questions posées.

Il interagit avec les formulaires en utilisant une interface simple, répondant aux questions qui leur sont posées pendant ou après l'expérience.

2.2.2. Liste des besoins

Remarque : À la suite d'une réunion tenue après l'achèvement de l'itération 3, la liste des besoins fonctionnels du client a été mise à jour afin de mieux prioriser les fonctionnalités les plus importantes pour ce dernier et de garantir leur réalisation lors de l'itération suivante. Ci-après figure la liste hiérarchisée des besoins fonctionnels mise à jour.

Tableau 1 : Liste hiérarchisée des besoins fonctionnels selon le critère MoSCoW

Must-haves	Should-haves
<ul style="list-style-type: none">• La création d'un formulaire par l'expérimentateur.• L'intégration d'un formulaire dans un scénario d'expérimentation.• Enregistrer le formulaire rempli par le participant (en sauvegardant son identité).• La suppression d'un formulaire par un expérimentateur.• La modification d'un formulaire par un expérimentateur.• La possibilité d'avoir des formulaires multipages et de pouvoir facilement naviguer entre les différentes pages d'un formulaire.• La possibilité de télécharger les réponses des participants à un formulaire donné au format CSV.• Lorsque le participant clique sur le bouton Submit, la fenêtre de la plateforme doit se fermer, et il doit être redirigé vers Tobii (vérifier que Tobii détecte bien que le participant a terminé la tâche de remplissage du formulaire).• Donner un moyen, dans l'interface d'administration des formulaires, de préciser quel est l'ID d'utilisateur par défaut si aucun ID d'utilisateur n'est précisé dans l'url d'accès à un formulaire.	<ul style="list-style-type: none">• La possibilité qu'un participant puisse remplir le formulaire d'une manière morcelée.• Sauvegarder de manière régulière ce qui a été saisi par le participant lors du remplissage du formulaire.• La validation des informations saisies par les participants dans les formulaires.• Permettre la duplication complète d'un formulaire.• Mise en page CSS des différentes pages de la plateforme.

Could-haves	Won't-haves
<ul style="list-style-type: none"> ● Ajouter la possibilité d'enregistrer des composants et de les réutiliser dans différents formulaires. ● L'authentification des expérimentateurs (dans le cas où la solution n'est plus hébergée localement). ● Nous avons proposé au client les fonctionnalités suivantes, que nous avons jugées utiles pour le futur : <ul style="list-style-type: none"> ● La possibilité d'écouter les questions du formulaire à haute voix pour les personnes ne pouvant pas lire. ● La possibilité de créer des formulaires dans différentes langues pour accueillir une diversité de participants. 	<ul style="list-style-type: none"> ● La visualisation des réponses des participants sous forme de graphiques (peut faire l'objet d'un autre sujet ptrans). ● On a aussi proposé cette fonctionnalité au client : <ul style="list-style-type: none"> ● La possibilité de créer d'une manière collaborative les formulaires en temps réel par les expérimentateurs.

2.3. Besoins non fonctionnels

- **Compatibilité :**
 - Le développement et les tests du logiciel devront se faire sous l'environnement Windows.
 - Utilisation de logiciels tels que OpenSesame pour l'intégration des formulaires aux expérimentations.
 - Compatibilité avec certains dispositifs utilisés par l'XP-Lab, comme l'oculométrie.
- **Performance :**
 - Chargement des formulaires dans un temps raisonnable.
- **Capacité :**
 - Une capacité mémoire permettant le stockage des données saisis par les participants.
- **Extensibilité (Scalabilité) :**
 - Au fil du temps, le nombre croissant d'expériences et de formulaires remplis entraînera un volume de données de plus en plus important à stocker, d'où le besoin de s'y adapter.
- **Aptitude à la maintenance :**
 - Respect des conventions de codage afin de livrer un code lisible et propre.

- **Ergonomie :**
 - La solution proposée doit être intuitive, tant pour les chercheurs lors de la création des formulaires que pour les participants lors de leur remplissage.

2.4. Évolution potentielle des besoins

- Pour faciliter le développement de futures fonctionnalités après l'achèvement du projet, il est essentiel de fournir une documentation claire avec le code source livré, permettant aux développeurs ultérieurs de naviguer facilement dans l'architecture existante et de procéder à des ajouts sans difficultés.
- Le client peut envisager l'intégration de **la plateforme web** dans l'outil openSource **OpenSesame** via un plugin développé spécifiquement pour cette configuration. Ce plugin permettra d'accéder aux fonctionnalités de cette plateforme directement depuis OpenSesame, offrant ainsi une interaction fluide entre les deux systèmes.
- Le client a exprimé son souhait que la plateforme intègre une fonctionnalité permettant de **visualiser** les réponses des participants sous forme de **graphiques**, afin de faciliter l'identification des liens de corrélation entre les réponses des participants en fonction de différentes variables (l'âge ou le genre du participant par exemple).
- Une autre évolution possible serait d'ajouter **la fonctionnalité d'authentification**, ce qui permettrait de sécuriser l'accès à la plateforme et aux données qu'elle contient, en plus de permettre de séparer les formulaires créés par chaque expérimentateur et les réponses associées. Ainsi, chaque expérimentateur aurait principalement accès uniquement aux formulaires qu'il a créés (on peut aussi envisager une fonctionnalité de transfert de formulaires d'un expérimentateur à un autre). La fonctionnalité d'authentification nécessiterait d'ajouter un attribut **mot de passe** et un attribut **email** (qui servira d'identifiant unique, plus pratique que l'ID) à la table Expérimentateur. Il faudrait également ajouter la possibilité de **créer un compte** expérimentateur, ainsi que d'autres fonctionnalités pour gérer ces comptes (modification des attributs, suppression, etc.).
- On pourrait aussi ajouter d'**autres paramètres** pour les formulaires (en plus de celui de navigation entre les pages d'un formulaire, que nous aborderons plus tard dans le rapport), notamment :
 - un paramètre permettant de **changer le thème visuel du formulaire** (par exemple ?theme=dark et ?theme=light),
 - un paramètre permettant d'afficher une **barre de progression**, utile lorsque les formulaires sont très longs,
 - ou encore un paramètre permettant de **choisir la langue** dans laquelle les questions du formulaire sont rédigées. »
- La librairie **Form.js** [4] que nous avons utilisée pour l'implémentation de la plateforme offre la possibilité de **créer d'autres composants** [5] en plus de ceux déjà proposés par la librairie. Par exemple, on peut développer un composant input range,

ou encore un composant permettant aux utilisateurs de téléverser un fichier. Il est également possible de créer des composants qui permettent de visualiser des données directement dans le formulaire, par exemple sous forme de graphiques, de courbes ou d'autres représentations visuelles adaptées au contexte du formulaire. Enfin, il est possible d'**étendre le panneau de propriétés** avec des entrées personnalisées, afin de configurer encore plus finement le comportement des différents composants.

2.5. Les risques

2.5.1. Liste des risques du projet

la gravité des conséquences		la probabilité d'occurrence		Criticité= impact*probabilité	
Mineur	1	Improbable	1	Modéré	
Majeur	2	Peu probable	2	Critique	
Grave	3	Probable	3	Très Critique	
Catastrophique	4	Très Probable	4		

Tableau 2 : Liste des risques identifiés avec leurs gravités, probabilités d'occurrence et criticités

ID	Nature du Risque	Description	Gravité	Probabilité	Criticité
R1	Technique	Intégration avec Tobii Pro Lab : Difficultés à intégrer les nouvelles fonctionnalités avec la plateforme existante.	Grave	Probable	Modéré
R2	Planification	Retards de développement : Dépassement des délais en raison de problèmes techniques ou de changements dans les exigences.	Grave	Probable	Très Critique
R3	Humains	Absence de compétences : Manque de compétences spécifiques nécessaires pour le développement de certaines fonctionnalités.	Grave	Probable	Très Critique
R4	Technique	Documentation Insuffisante : Risque que la documentation des outils et bibliothèques utilisées ne soit pas suffisamment détaillée, rendant difficile le travail de l'équipe.	Majeur	Peu Probable	Critique
R5	Planification	Problèmes d'Ordonnancement : Risque que de petites tâches prennent plus de temps que prévu, entraînant des retards mineurs dans le calendrier global du projet.	Mineur	Peu Probable	Modéré

Tableau 3 : Tableau « probabilité-conséquences»

	Improbable	Peu probable	Probable	Très Probable
Catastrophique				
Grave			R1,R2,R3	
Majeur		R4		
Mineur		R5		

2.5.2. Politique de gestion des risques

À la fin de chaque phase du projet, nous procéderons à une révision et une actualisation du registre des risques. Cela nous permettra d'identifier et d'évaluer continuellement les menaces, qu'elles soient déjà présentes ou émergentes. Lors de ces révisions, nous veillerons à gérer au moins deux risques majeurs et deux risques modérés dans chaque rapport. Pour ce faire, nous ferons le point sur les risques déjà identifiés et intégrerons de nouveaux risques si besoin. Ensuite, nous élaborerons un plan d'action détaillant des stratégies spécifiques pour chaque risque identifié. Enfin, nous mettrons en place un suivi régulier pour évaluer l'efficacité des mesures que nous aurons prises, afin d'assurer une gestion proactive des risques tout au long du projet.

Pour traiter les risques déjà identifiés dans le projet, les solutions suivantes ont été définies :

- **R1** : Des sessions de tests d'intégration seront organisées dès les premières phases pour identifier et anticiper les éventuels problèmes d'intégrabilité .
- **R2** : Un calendrier de développement détaillé avec des jalons intermédiaires sera établi pour suivre l'avancement et identifier rapidement les retards.
- **R3** : Investir dans la compréhension approfondie de la documentation existante afin de pallier les lacunes en compétences spécifiques. En complément, solliciter le retour d'experts externes pour orienter l'équipe et apporter des clarifications nécessaires.
- **R5** : Intégrer des marges de sécurité dans le calendrier pour les tâches courtes et ajuster le planning régulièrement afin de prévenir les conflits d'ordonnancement. Utiliser un outil de gestion de projet pour surveiller l'avancement des tâches et anticiper les éventuels retards, permettant ainsi une réévaluation continue des priorités.

2.6. Plannings

2.6.1. Planning réalisé

Les deux itérations trois et quatre ont été réalisées durant la phase 3B du projet. Le tableau ci-dessous récapitule, pour chacune d'elles, les tâches effectuées, les dates de début et de fin, le temps passé sur chaque tâche, le membre de l'équipe ayant réalisé chaque tâche, ainsi qu'un commentaire détaillant ce qui a été effectué.

Vous trouverez le planning réalisé des deux itérations 1 et 2, qui ont été déjà effectués durant la phase 3A du projet, dans l'ancienne version du rapport.

Tableau 4 : Les itérations réalisées

Itération	Tâche	Effectuée par	Temps passé	Dates de début et de fin	Commentaires
3	Essayer d'intégrer la librairie formily dans l'application.	Mehdi	4h	28/02/2025	-Due au manque de documentation, on n'a pas pu utiliser la librairie formily dans notre application.
3	Création de la base de données sur Postgresql	Mohamed, Ghassen	3h	28/02/2025	

3	Implémentation de la fonctionnalité : Création d'un formulaire par l'expérimentateur	Mohamed	8h	Frontend : 28/02/2025 (17h à 22h) Backend : 03/03/2025 (15h15 à 18h15)	Une bonne partie du temps a été consacrée à comprendre comment utiliser la librairie formjs dans l'application. - Il manque quelques éléments de style sur cette page malgré qu'on a importer tout les feuilles de style css indiqués dans la documentation.
3	Implémentation de la page d'accueil	Mohamed	30min	03/03/2025	-Il manque le style css de la page
3	Implémentation de la fonctionnalité : Consulter un formulaire déjà créé par l'expérimentateur	Mohamed	1h30	03/03/2025	
3	Implémentation de la fonctionnalité : La modification d'un formulaire par l'expérimentateur	Ghassen	1h	04/03/2025	-Il faut juste enlever la possibilité de modifier un formulaire une fois un participant y a répondu.
3	Adapter la page de consultation de formulaire au rôle de l'utilisateur et Accessibilité depuis Tobii pro lab.	Mohamed	2h	04/03/2025	-Si l'URL utilisée pour accéder à cette page se termine par l'ID du participant, il sera possible de soumettre le formulaire. Sinon, d'autres éléments utiles pour l'expérimentateur seront affichés, sans possibilité de soumission.
3	Implémentation de la fonctionnalité : Enregistrer le formulaire rempli par le participant	Mohamed	3h	06/03/2025	-Il a fallu faire le mapping entre les valeurs key et id de chaque component avant d'envoyer les réponses au backend.
3	Implémenter la page qui affiche le tableau des réponses à un formulaire spécifique	Ghassen	2h	06/03/2025 à	
3	Trouver une solution au manque des feuilles de style css dans la page de création de formulaires.	Mohamed	2h	06/03/2025	
3	Implémentation de la fonctionnalité : Sauvegarder d'une manière régulière les réponses à un formulaire lors de son remplissage (auto-save).	Mohamed	1h40	07/03/2025	
3	Implémentation de la fonctionnalité : La suppression d'un formulaire	Ghassen	1h	18/03/2025	

3	Implémentation de la fonctionnalité : La possibilité d'enregistrer des composants du formulaire et les réutiliser dans différents formulaires	Mehdi	2h	21/03/2025	
4	Réunion de la fin de l'itération 3	Toute l'équipe	1h	26/03/2025	
4	Rédaction du compte rendu de la réunion	Rédaction : Mohamed / Relecture : Ghassen , Mehdi	1h	26/03/2025	
4	Changer le SGBD utilisé de PostgreSQL à SQLite .	Ghassen	3h	10/04/2025	Modification des routes déjà existants
4	Ajouter la possibilité de lancer simultanément la partie frontend et backend avec une seule commande 'npm start'.	Ghassen	1h	14/04/2025	
4	Lorsque le participant clique sur le bouton Submit , la fenêtre de la plateforme doit se fermer, et il doit être redirigé vers Tobii (vérifier que Tobii détecte bien que le participant a terminé la tâche de remplissage du formulaire).	Mohamed	2h	15/04/2025	J'ai essayé window.close() : ne marche pas puisque la fenêtre n'a pas été ouverte par window.open() et j'ai essayé exec('taskkill /FI "WINDOWTITLE eq React App*" /F') ne marche pas : accès refusé
4	Ajouter la possibilité de créer des formulaires multipages .	Mohamed	2h15	15/04/2025	On utilise le composant "separator" pour séparer les pages du formulaire
4	Prévoir un moyen de navigation entre les différentes pages du formulaire une fois celui-ci créé.	Mohamed	5h	15/04/2025	Après avoir rempli une page du formulaire, lorsque le participant navigue vers la page suivante puis revient à la page déjà remplie, il doit retrouver les données qu'il a déjà saisies. Il faut donc les charger à partir de la base de données. Pour cela, il a été nécessaire de modifier la structure de la base de données afin d'éviter les réponses dupliquées pour un même individu, et d'adapter les APIs en conséquence et veiller à ce que la fonction auto-save ne se déclenche pas avant le chargement des données.
4	Le tableau des réponses à un formulaire doit être organisé	Mehdi	1H	15/04/2025	Ajout d'un fichier css

	de façon à ce qu'une colonne représente une question et une ligne corresponde à un participant.				
4	Ajouter la possibilité de télécharger les réponses au format CSV .	Mehdi	1H	15/04/2025	
4	Permettre la duplication complète d'un formulaire (prévoir la génération de nouveaux identifiants pour les composants dupliqués ainsi que pour le formulaire dupliqué).	Ghassen	2h	17/04/2025	
4	Dans la page de création de formulaires, prévoir la possibilité que l'utilisateur aie besoin de modifier l'id du formulaire (et donc si il le modifie, vérifier que l'id n'a pas été déjà utilisé pour un autre formulaire existant).	Mehdi	1h	17/04/2025	
4	Donner un moyen, dans la page d'accueil, de préciser quel est l'ID d'utilisateur par défaut si aucun ID d'utilisateur n'est précisé dans l'url d'accès à un formulaire.	Ghassen	1h	20/04/2025	
4	Rédaction du rapport et du fichier .md contenant le guide utilisateur	Mohamed	7h30	29/04/2025 - 30/04/2025	
4	Correction du code et de certains bugs signalés par le client et l'encadrant.	Ghassen & Mehdi	6h	28/04/2025 - 30/04/2025	

3. Travail réalisé

3.1. Première et deuxième itération

Dans cette partie vous trouverez un rappel de ce qui a été effectué avant la soutenance qui a eu lieu au mois de février (donc la **phase 3A**), qui englobe le choix de l'architecture logicielle de la plateforme à implémenter, sa conception UML, les scénarios d'utilisation, un prototype IHM et enfin le choix des technologies et des bibliothèques à utiliser pour l'implémentation.

3.1.1. L'architecture choisie pour la plateforme à implémenter

La solution qu'on a implémentée est **une plateforme web** hébergée localement sur une machine dans laquelle **Tobii Pro Lab** est installé (elle est donc accessible via un URL local). Le processus commence par la création des formulaires par l'expérimentateur sur la plateforme, chaque formulaire est associé à un URL spécifique de la forme suivante :

`http://localhost:3000/form-viewer/ID du form`

Et chaque page d'un formulaire possède un URL spécifique de la forme suivante :

`http://localhost:3000/form-viewer/ID du form/numéro de la page`

Dans cette plateforme un participant est identifié par un id unique, d'où lorsqu'on veut qu'un participant remplit une page d'un formulaire spécifique, on accède à la page du formulaire à remplir via un URL de la forme suivante :

`http://localhost:3000/form-viewer/ID du form/numéro de la page/id du participant`

Remarque : Dans certains cas, l'expérimentateur peut souhaiter que le participant accède directement à une seule page d'un formulaire multipages sans pouvoir passer aux autres pages. Pour cela, un paramètre "navigation" a été ajouté à l'URL. Lorsqu'il est inclus, deux boutons, "Page suivante" et "Page précédente", sont affichés pour permettre au participant de naviguer entre les pages du formulaire. Si ce paramètre n'est pas inclus dans l'URL, ces deux boutons ne sont pas visibles.

`http://localhost:3000/form-viewer/ID du form/numéro de la page/id du participant?navigation=True`

Les formulaires créés sont ensuite accessibles depuis Tobii via l'URL associée à chaque formulaire créé, permettant ainsi à l'expérimentateur de réaliser son scénario d'expérimentation sur Tobii Pro Lab, sans perturber son utilisation des différents équipements tel que l'eye tracker, tout en recueillant les réponses des participants à des questions qu'il souhaite leur poser à des moments spécifiques en basculant à notre plateforme, puis revenir à Tobii Pro Lab pour poursuivre le scénario.

Les expérimentateurs peuvent accéder aux réponses des participants via la plateforme, où tous les formulaires ainsi que les réponses associées sont enregistrés, ils ont aussi la possibilité de télécharger les réponses sous forme d'un fichier CSV.

La principale limite de cette solution est qu'elle est hébergée localement : l'accès aux formulaires et aux résultats est restreint à l'ordinateur sur lequel elle est déployée, ce qui peut poser problème aux expérimentateurs souhaitant accéder aux données à distance. Nous avons envisagé la possibilité d'héberger cette solution sur une URL publique dédiée. Cependant, après notre échange avec le client, nous avons conclu que cela poserait des problèmes tels que les coûts d'hébergement sur le web et le besoin d'une connexion internet supplémentaire, sans apporter d'avantages notables. D'où la décision de l'héberger localement.

Ci-après est un schéma représentant l'architecture logicielle simplifiée de la solution implémentée.

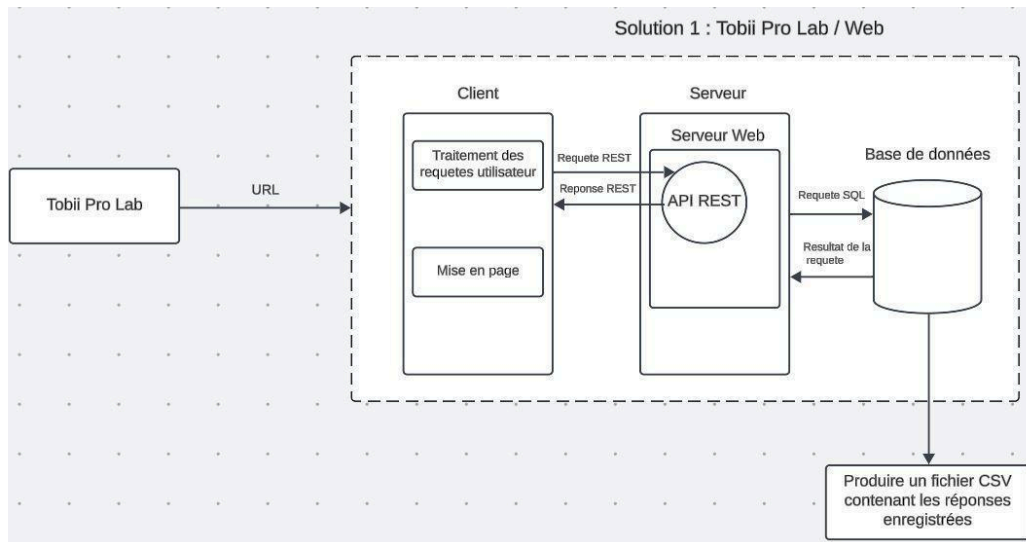


Figure 3 : L'architecture logicielle simplifiée de la solution implémentée

3.1.2. La conception UML

Suite au choix de la solution à implémenter on a passé à la conception UML de celle-ci.

3.1.2.1. Diagramme de cas d'utilisation

Afin de représenter de manière synthétique les différents acteurs de notre solution et les fonctionnalités qu'ils peuvent effectuer, nous avons élaboré le diagramme de cas d'utilisation suivant.

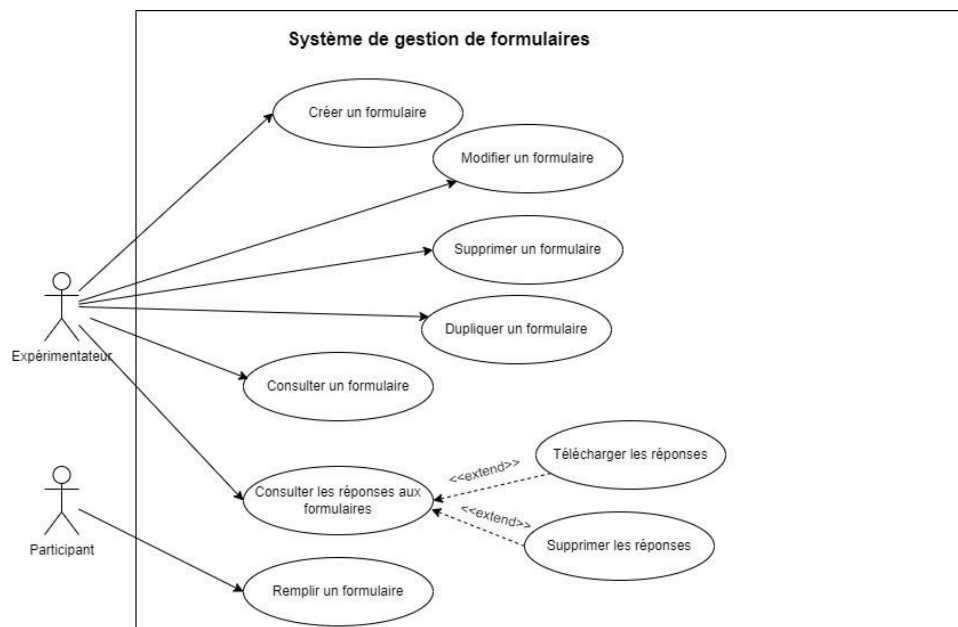


Figure 4 : Diagramme de cas d'utilisation de la solution

3.1.2.2. Diagramme de classes

On a pu déduire un diagramme de classes à partir du modèle de domaine en adaptant ce dernier aux aspects techniques qu'on doit respecter lors du développement de l'application et en prenant en compte la structure de la base de données sous-jacente.

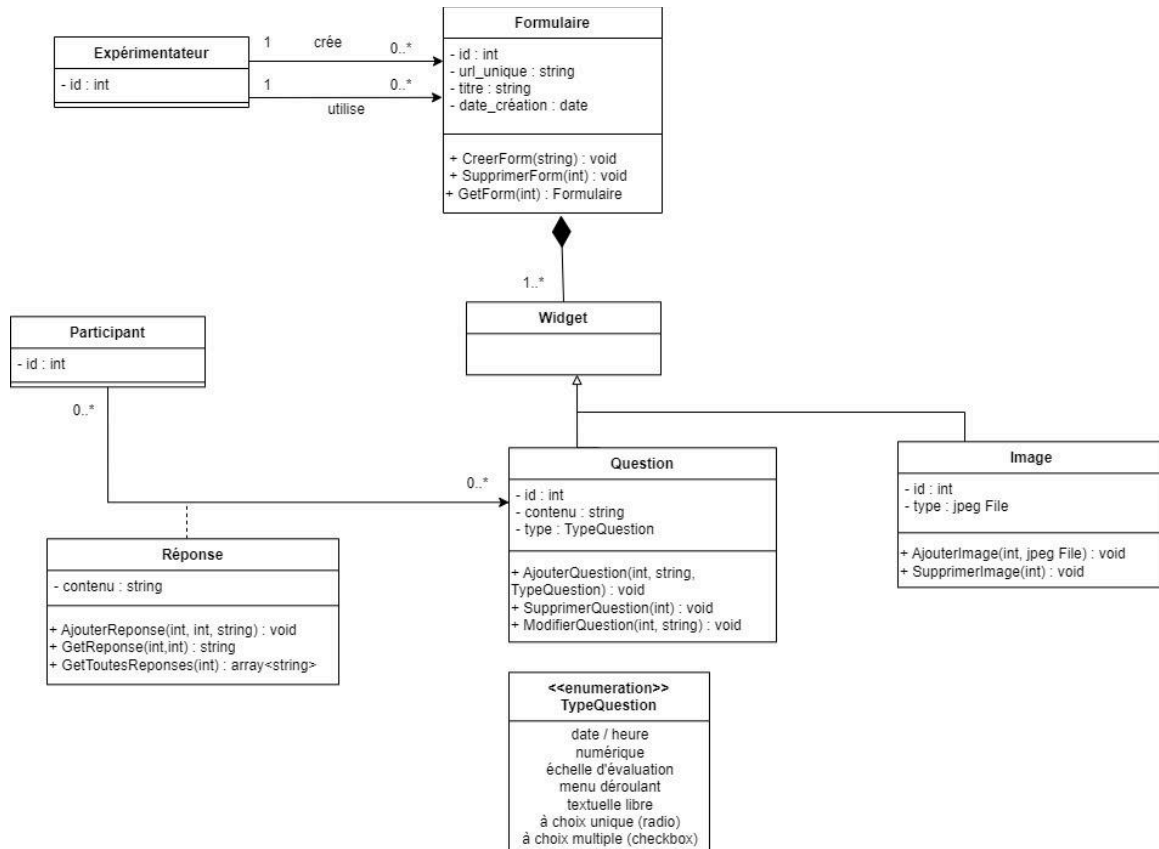


Figure 5 : Diagramme de classes de la solution

Remarques :

- Aucun autre attribut n'a été ajouté aux classes Expérimentateur et Participant, car nous n'envisageons pas, pour le moment, d'inclure des fonctionnalités d'authentification ou de création de comptes pour ces deux rôles.

3.1.3. Les scénarios d'utilisation

Suite aux retours que nous avons reçus concernant les diagrammes de séquence réalisés lors de l'itération 2 — notamment le fait qu'ils sont trop complexes et rendent difficile la compréhension du déroulement des différents cas d'utilisation — nous avons décidé de les remplacer par des descriptions textuelles des différents scénarios d'utilisation. Vous trouverez ci-après l'ensemble des scénarios d'utilisation de notre plateforme.

Tableau 5 : Description textuelle du scénario d'utilisation : Créer un formulaire

Créer un formulaire	
Acteur principal : Expérimentateur	
Objectif : Permettre à l'expérimentateur de créer un nouveau formulaire et de l'enregistrer dans notre plateforme.	
Scénario nominal	<ol style="list-style-type: none"> 1. L'acteur se rend à la page d'accueil. 2. Le système affiche la page d'accueil. 3. L'acteur choisit l'option "Créer un nouveau formulaire". 4. Le Système le redirige vers la page de création de formulaires. 5. L'acteur saisit le titre qu'il veut associer à son nouveau formulaire dans le champ dédié. 6. L'acteur construit son nouveau formulaire en glissant un par un les composants présents dans la partie gauche de la page vers le centre de la page. 7. Le Système affiche les composants ajoutés au formulaire au centre de la page (ce qui donne un prévu de ce à quoi le formulaire va ressembler). 8. Après avoir ajouté et paramétré les différents composants du formulaire, l'acteur clique sur le bouton "Enregistrer". 9. Le Système enregistre le nouveau formulaire puis redirige l'acteur vers la page d'accueil.
Alternatives	<p>A1. L'acteur peut modifier la position et l'espace occupé par un composant :</p> <ol style="list-style-type: none"> 8. L'acteur glisse le composant concerné dans l'emplacement qu'il souhaite qu'il occupe. 9. Le système change la position et l'espace occupé par le composant. <p>A2. L'acteur peut choisir un comportement spécifique de l'un de ces composants:</p> <ol style="list-style-type: none"> 8. L'acteur sélectionne le composant concerné 9. Le Système affiche dans la partie droite de l'écran les paramètres spécifiques au composant sélectionné. 10. L'acteur modifie les paramètres. <p>A3. L'acteur peut supprimer un composant qu'il a ajouté au formulaire en cours de création :</p> <ol style="list-style-type: none"> 8. L'acteur sélectionne le composant à supprimer 9. Le Système encadre le composant sélectionné et affiche à côté de ce composant une icône contenant une corbeille. 10. L'acteur clique sur l'icône. 11. Le Système supprime le composant du formulaire. <p>A4. L'acteur peut séparer le formulaire en plusieurs pages :</p>

	<p>8. L'acteur glisse le composant nommé "Separator" dans le formulaire dans l'emplacement où il souhaite que la séparation des deux pages soit effectuée.</p> <p>9. Le Système affiche le "Separator" dans le formulaire en cours de création sous forme d'une ligne horizontale séparant les composants des différentes pages.</p>
--	--

Tableau 6 : Description textuelle du scénario d'utilisation : Modifier un formulaire

Modifier un formulaire	
Acteur principal : Expérimentateur	
Objectif : Permettre à l'expérimentateur de modifier un formulaire déjà créé.	
Scénario nominal	<ol style="list-style-type: none"> 1. L'acteur se rend à la page d'accueil. 2. Le système affiche la page d'accueil. 3. L'acteur choisit l'option "Modifier" correspondante au formulaire qu'il souhaite modifier. 4. Le Système le redirige vers la page de modification de formulaires. 5. Le Système charge le formulaire à modifier ainsi que son titre. 6. L'acteur saisit le nouveau titre qu'il veut associer à son formulaire dans le champ dédié (dans le cas où il souhaite le modifier). 7. L'acteur modifie le formulaire en ajoutant, supprimant ou paramétrant des composantes. 8. Le Système affiche les modifications appliquées au formulaire au centre de la page (ce qui donne un prévu de ce à quoi le formulaire va ressembler). 9. Après avoir modifié le formulaire, l'acteur clique sur le bouton "Mettre à jour". 10. Le Système enregistre les modifications du formulaire puis redirige l'acteur vers la page d'accueil.
Alternatives	<p>A1. On ne peut pas modifier un formulaire s'il a été déjà rempli par un participant :</p> <p>4. Le Système affiche un pop-up contenant le texte "Ce formulaire contient déjà des réponses et ne peut pas être modifié".</p> <p>A2. Après avoir modifié le formulaire, l'acteur clique sur le bouton "Retour à l'accueil" :</p> <p>10. Un pop-up s'affiche, informant l'acteur qu'il n'a pas enregistré ses modifications et que, s'il retourne à la page d'accueil sans les enregistrer, celles-ci seront perdues.</p>

--	--

Tableau 7 : Description textuelle du scénario d'utilisation : Supprimer un formulaire

Supprimer un formulaire	
Acteur principal : Expérimentateur	
Objectif : Permettre à l'expérimentateur de supprimer un formulaire qu'il a déjà créé	
Scénario nominal	<ol style="list-style-type: none"> 1. L'acteur se rend à la page d'accueil. 2. Le système affiche la page d'accueil. 3. L'acteur choisit l'option "Supprimer" correspondante au formulaire qu'il souhaite supprimer. 4. Le Système affiche un pop-up contenant le texte "Êtes-vous sûr de vouloir supprimer ce formulaire ? Toutes les réponses associées seront perdues. Cette action est irréversible" et deux boutons "Confirmer" et "Annuler". 5. L'acteur clique sur le bouton "Confirmer". 6. Le Système supprime le formulaire, puis affiche un pop-up contenant le texte "Formulaire et réponses supprimés !". 7. L'acteur clique sur le bouton "Ok". 8. Le Système cache le pop-up.
Alternatives	<p>A1. L'acteur a changé d'avis et ne veut plus supprimer le formulaire :</p> <ol style="list-style-type: none"> 5. L'acteur clique sur le bouton "Annuler". 6. Le Système cache le pop-up.

Tableau 8 : Description textuelle du scénario d'utilisation : Dupliquer un formulaire

Dupliquer un formulaire	
Acteur principal : Expérimentateur	
Objectif : Permettre à l'expérimentateur de dupliquer un formulaire qu'il a déjà créé	
Scénario nominal	<ol style="list-style-type: none"> 1. L'acteur se rend à la page d'accueil. 2. Le système affiche la page d'accueil. 3. L'acteur choisit l'option "Dupliquer" correspondante au formulaire qu'il souhaite dupliquer. 4. Le Système duplique le formulaire (en lui générant un nouveau id et en dupliquant tous ses composants) puis affiche un pop-up contenant le texte "Formulaire dupliqué avec succès ! Nouveau Formulaire ID: <id du nouveau

	<p>formulaire>" et un bouton "Ok".</p> <p>5. L'acteur clique sur le bouton "Ok".</p> <p>6. Le Système cache le pop-up.</p>
Alternatives	

Tableau 9 : Description textuelle du scénario d'utilisation : Consulter un formulaire

Consulter un formulaire	
Acteur principal : Expérimentateur	
Objectif : Permettre à l'expérimentateur de consulter un formulaire qu'il a créé	
Scénario nominal	<ol style="list-style-type: none"> 1. L'acteur se rend à la page d'accueil. 2. Le système affiche la page d'accueil. 3. L'acteur choisit l'option "Voir" correspondante au formulaire qu'il souhaite consulter. 4. Le Système le redirige vers la page de consultation de formulaires (vue d'expérimentateur) ayant un URL de la forme : http://localhost:3000/form-viewer/ID du form/numéro de la page?navigation=True 5. L'acteur peut consulter le formulaire, naviguer entre les différentes pages du formulaire en cliquant sur le bouton "Page suivante" et "Page précédente".
Alternatives	<p>A1. L'acteur souhaite intégrer un formulaire dans un scénario d'expérimentation de Tobii Pro Lab :</p> <p>5. L'acteur fait une copie de l'URL indiqué dans la page de consultation du formulaire qu'il souhaite intégrer qui est de la forme : http://localhost:3000/form-viewer/ID du form/numéro de la page/id du participant?navigation=True.</p> <p>6. L'acteur colle cet URL dans le champ de texte correspondant sur Tobii, en remplaçant /id du participant par l'id du participant qu'il souhaite qu'il remplisse le formulaire (vous trouverez une explication plus détaillée dans la partie intégration avec Tobii Pro Lab du rapport).</p>

Tableau 10 : Description textuelle du scénario d'utilisation : Consulter les réponses des participant à un formulaire

Consulter les réponses des participant à un formulaire
--

Acteur principal : Expérimentateur	
Objectif : Permettre à l'expérimentateur de consulter la liste des réponses des participants ayant rempli un formulaire qu'il a créé.	
Scénario nominal	<ol style="list-style-type: none"> 1. L'acteur se rend à la page d'accueil. 2. Le système affiche la page d'accueil. 3. L'acteur choisit l'option "Voir Réponses" correspondante au formulaire qu'il souhaite consulter les réponses qui lui sont associées. 4. Le Système le redirige vers la page de consultation de réponses où il trouve un tableau qui contient les réponses des participants ayant rempli le formulaire concerné, tel que chaque colonne représente une question du formulaire et chaque ligne représente les réponses d'un participant donné (chaque participant est identifié par son id).
Alternatives	<p>A1. L'acteur souhaite télécharger sous format CSV le tableau des réponses :</p> <ol style="list-style-type: none"> 5. L'acteur clique sur le bouton "Exporter en CSV". 6. Le Système lance le téléchargement du tableau des réponses sous format CSV. <p>A2. L'acteur souhaite supprimer toutes les réponses affichées dans le tableau :</p> <ol style="list-style-type: none"> 5. L'acteur clique sur le bouton "Supprimer toutes les réponses". 6. Le Système affiche un pop-up contenant le texte "Êtes-vous sûr de vouloir supprimer toutes les réponses de ce formulaire ?" et deux boutons "Confirmer" et "Annuler". 7. L'acteur choisit l'option "Confirmer". 9. Le Système affiche un pop-up contenant le texte "Toutes les réponses ont été supprimées." et un bouton "Ok". 10. L'acteur clique sur "Ok". 11. Le Système cache le pop-up.

Tableau 11 : Description textuelle du scénario d'utilisation : Remplir un formulaire

Remplir un formulaire
Acteur principal : Participant
Objectif : Permettre à un participant de remplir un formulaire.
Pré-condition : L'expérimentateur a ajouté le remplissage du formulaire sur notre plateforme par le participant comme étape du scénario d'expérimentation en saisissant

<p>l'URL du formulaire à remplir (de la forme http://localhost:3000/form-viewer/ID du form/numéro de la page/id du participant?navigation=True) dans le champ dédié dans Tobii Pro Lab et en choisissant la durée au bout de laquelle le participant sera redirigé du formulaire au stimuli suivant.</p>	
Scénario nominal	<ol style="list-style-type: none"> 1. A un moment donné durant l'expérience, Tobii Pro Lab ouvre le navigateur et redirige le participant vers le formulaire correspondant à l'URL saisit par l'expérimentateur. 2. L'acteur (le participant) saisit ses réponses dans les champs du formulaire (et sélectionne des choix s'il s'agit de select lists, checkboxes ou radios...). 3. Le système enregistre les réponses de l'acteur à fur et à mesure du remplissage du formulaire, plus précisément à chaque fois qu'il remplit un des champs du formulaire grâce à l'événement 'onchange'. Cela garantira qu'aucune réponse sera perdue dans le cas d'une panne ou un imprévu qui survient sur le système. 4. L'acteur clique sur le bouton "Submit" (s'il est présent dans le formulaire qu'il est en train de remplir). 5. Le Système affiche un pop-up contenant le texte "Votre formulaire a été soumis avec succès." et un bouton "Ok". 6. L'acteur clique sur le bouton "Ok" 7. Le Système redirige l'acteur vers une page contenant le texte "Merci pour votre réponse !". 8. Dès que le temps consacré au remplissage du formulaire s'écoule ou si l'acteur clique sur la touche "F10", Tobii redirige l'acteur vers le stimuli suivant.
Alternatives	

Tableau 12 : Description textuelle du scénario d'utilisation : Définir un id participant par défaut

Définir un id participant par défaut	
Acteur principal : Expérimentateur	
<p>Objectif : Permettre à l'expérimentateur de définir un id participant par défaut qui sera ajouté dynamiquement à l'URL d'accès d'un formulaire si ce dernier contient le symbole clé @ au lieu de l'id du participant (l'utilité de cette fonctionnalité sera expliquée dans la partie implémentation ici).</p>	
Scénario nominal	<ol style="list-style-type: none"> 1. L'acteur se rend à la page d'accueil.

	<ol style="list-style-type: none"> 2. Le système affiche la page d'accueil. 3. L'acteur saisit l'id du participant par défaut dans le champ de texte dédié, puis clique sur "Sauvegarder". 4. Le Système affiche un pop-up contenant le texte "ID participant par défaut enregistré !" et un bouton "Ok". 5. Dans Tobii Pro Lab, l'acteur saisit l'URL du formulaire qu'il souhaite que les participants remplissent mais cette fois en mettant le symbole @ au lieu de l'id participant (donc un URL de la forme : http://localhost:3000/form-viewer/ID du form/numéro de la page/@?navigation=True). 6. L'acteur lance le scénario d'expérimentation sur Tobii Pro Lab 7. A un moment donné durant l'expérience, Tobii Pro Lab ouvre le navigateur et redirige le participant vers le formulaire correspondant à l'URL saisit par l'acteur. 8. Le système remplace dynamiquement le symbole @ présent dans l'URL par l'id du participant par défaut (donc un URL de la forme : http://localhost:3000/form-viewer/ID du form/numéro de la page/@ va être dynamiquement changé en http://localhost:3000/form-viewer/ID du form/numéro de la page/id du participant par défaut). 9. Le participant remplit le formulaire affiché.
Alternatives	

3.1.4. Prototype IHM

Afin d'assurer une vision commune de l'interface homme-machine de la solution à implémenter, tant entre les membres de l'équipe qu'avec le client, nous avons conçu un prototype IHM à l'aide de l'outil Figma.

3.1.4.1. L'interface du participant

En ce qui concerne l'interface du participant, on a réalisé le prototype suivant.

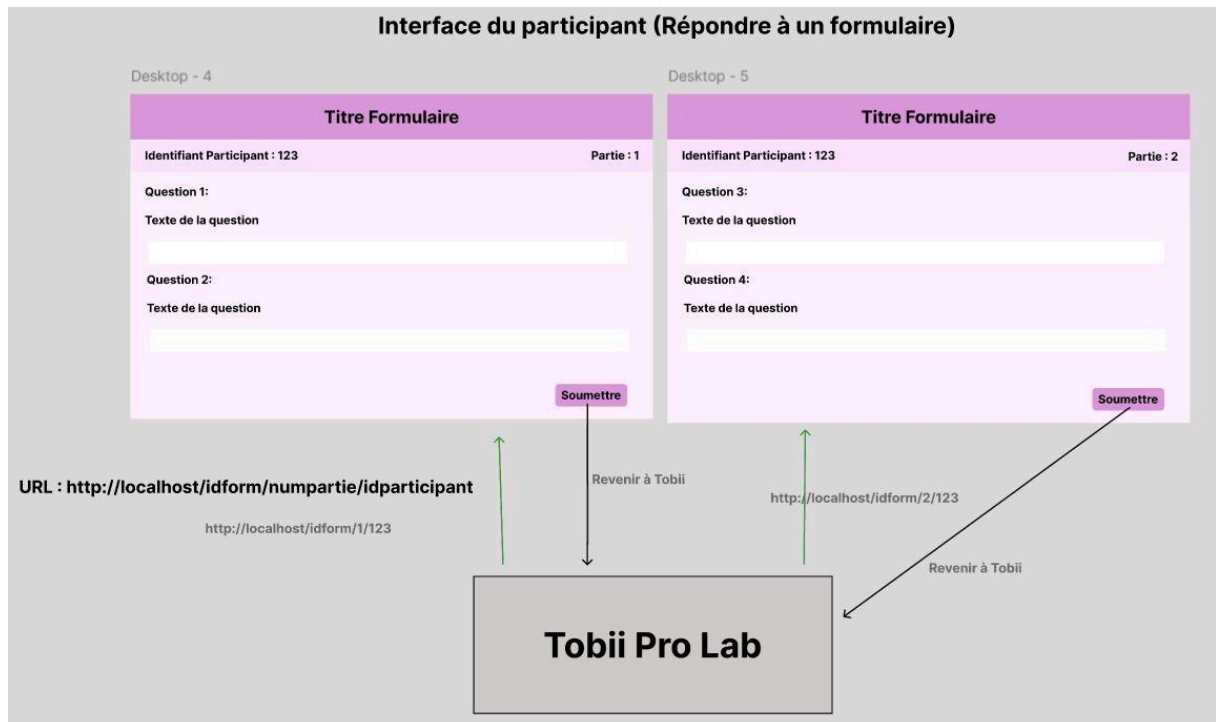


Figure 6 : Le prototype de l'interface du participant

Ce prototype illustre le processus par lequel le participant naviguera entre l'outil Tobii Pro Lab, où se déroulera l'expérience, et notre plateforme, sur laquelle il devra remplir tout ou une partie d'un formulaire. Une fois sa réponse soumise, il sera redirigé vers Tobii pour poursuivre l'expérience.

Comme on peut le constater, chaque partie du formulaire est associée à une URL au format suivant : **http://localhost/idform/numpartie/idparticipant**. Cette URL sera intégrée par l'expérimentateur dans Tobii en tant que "web stimulus" et constituera une étape du scénario expérimental.

3.1.4.2. L'interface de l'expérimentateur

Pour l'interface de l'expérimentateur, qui aura accès à différentes fonctionnalités on a élaboré les prototypes suivants :



Figure 7 : Le prototype de l'interface de l'expérimentateur (la page d'accueil)

Il s'agit de la première page qui va s'afficher lorsque l'expérimentateur visitera notre plateforme. Elle présentera la liste des formulaires créés précédemment, avec pour chacun d'eux des informations telles que le nombre de réponses, la date de création, la date de la dernière réponse, ainsi que trois boutons : l'un pour consulter le formulaire, un autre pour visualiser la liste des réponses, et un dernier pour le supprimer.

Cette page comporte également un bouton permettant d'accéder à la page de création de formulaires.

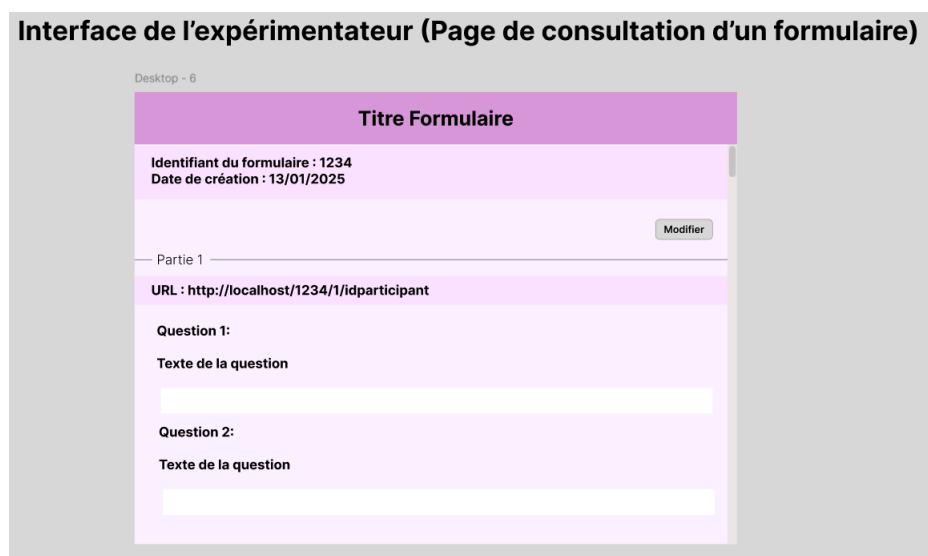


Figure 8 : Le prototype de l'interface de l'expérimentateur (la page de consultation d'un formulaire)

Sur cette page, l'expérimentateur peut consulter en détail un formulaire déjà créé, visualiser les widgets qui le composent, l'identifiant du formulaire, la date de création, ainsi que l'URL de chaque partie du formulaire. Une option "Modifier" permet de passer en mode édition pour apporter des modifications au formulaire.

Interface de l'expérimentateur (Page des réponses d'un formulaire)

Desktop - 2

Formulaire 1					
Export					
#	1. Participan...	2. Quel âge ...	3. Vous êtes :	4. A quelle fréquence utilis...	5. Quelles sont vos habitudes de réservation :
1	1	54	Je ne sais pas	Quelques fois par an	AirBnB ; Booking.com ; Autre
2	19	33	Une femme	Quelques fois par an	AirBnB ; Booking.com ; Réservation en direct (site ...
3	1	24	Une femme	Quelques fois par an	AirBnB ; Booking.com
4	2	20	Un homme	Moins d'une fois par an	Hotels.com ; Réservation en direct (site web ou par ...
5	3	64	Un homme	Une à quelques fois par mois	AirBnB ; Booking.com ; Réservation en direct (site ...
6	4	32	Un homme	Quelques fois par an	AirBnB ; Booking.com ; Autre
7	5	67	Un homme	Quelques fois par an	Booking.com
8	6	33	Un homme	Une à quelques fois par mois	Booking.com ; Hotels.com ; Réservation en direct (...)

Figure 9 : Le prototype de l'interface de l'expérimentateur (la page des réponses d'un formulaire)

Cette page affiche les réponses fournies par différents participants à un formulaire spécifique, avec la possibilité de télécharger ces réponses au format CSV.

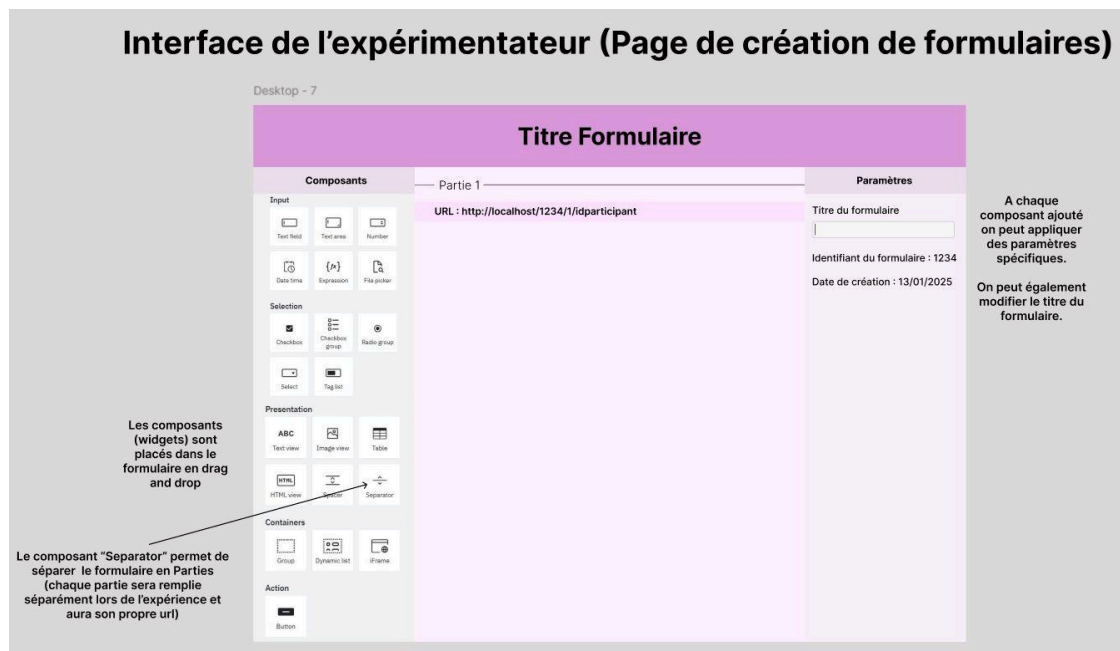


Figure 10 : Le prototype de l'interface de l'expérimentateur (la page de création de formulaires avant l'ajout de widgets)

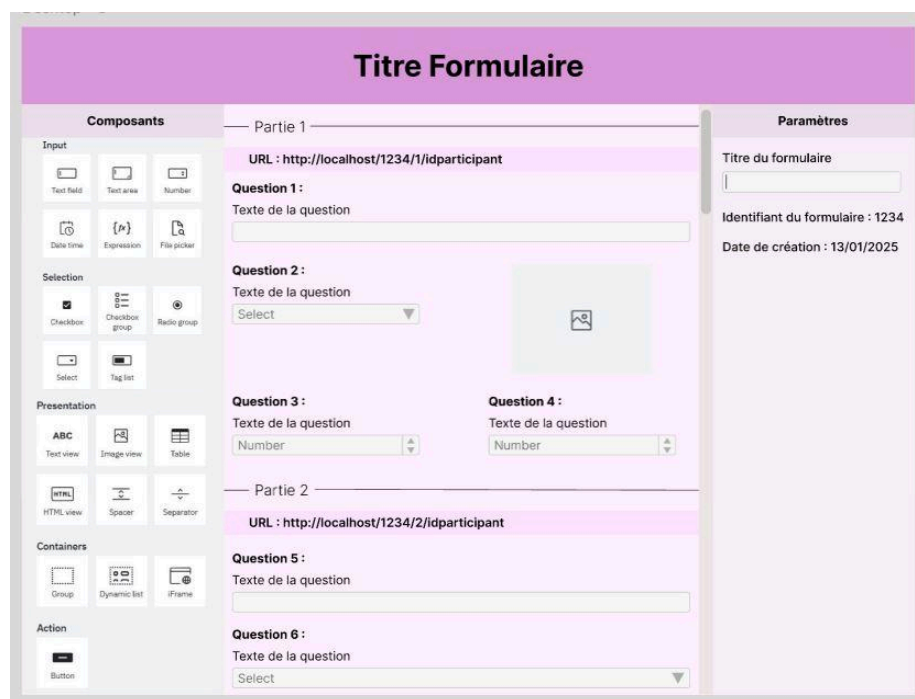


Figure 11 : Le prototype de l'interface de l'expérimentateur (la page de création de formulaires après l'ajout de widgets)

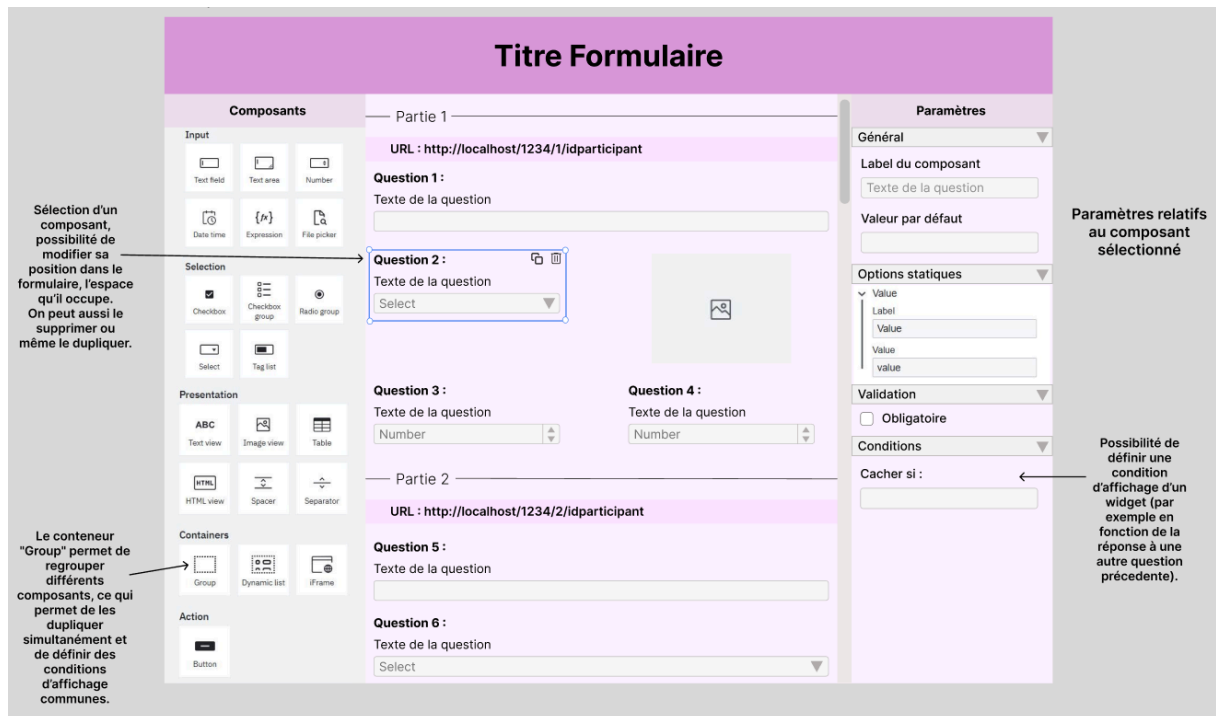


Figure 12 : Le prototype de l'interface de l'expérimentateur (la page de création de formulaires : explication de la mise en page, les conteneurs et l'affichage conditionnel)

La page de création de formulaires permet de concevoir un formulaire en ajoutant les widgets (ou composants) nécessaires. Chaque type de widget peut être configuré avec des paramètres spécifiques permettant de modifier son comportement, tels qu'un affichage conditionnel, des règles de validation, ou l'obligation de réponse.

Il est également possible de réorganiser les widgets en ajustant leur position et l'espace qu'ils occupent, offrant ainsi une mise en page avancée. Par ailleurs, les widgets peuvent être supprimés, dupliqués, ou regroupés dans un conteneur.

Ce prototype est accessible sur Figma via le lien suivant :

<https://www.figma.com/design/4fZpU4zN9yKRe98M5uin9S/Ptrans?node-id=0-1&t=QtYWlcv6ET301BpD-1>

3.1.5. Choix des technologies et bibliothèques à utiliser

3.1.5.1. Front-end : React avec la bibliothèque Form.js

Pour la partie front-end de l'application, nous avons opté pour **React**, une bibliothèque JavaScript très utilisée pour la création d'interfaces utilisateur interactives. Ce choix repose sur plusieurs avantages:

- **Modularité et composants réutilisables** : React permet une architecture en **composants**, facilitant la maintenance et l'évolution de l'application.

- **Écosystème riche** : Il existe de nombreuses bibliothèques compatibles, notamment **Ant Design** et **Form.js**, qui nous aideront à structurer et styliser l'interface.

Pour la gestion et la construction des formulaires, nous avons choisi la bibliothèque **Form.js**, qui est une bibliothèque JavaScript avancée et légère utilisant JSON pour l'entrée de données. Elle génère des formulaires dynamiques centrés sur l'utilisateur.

Après avoir hésité entre cette bibliothèque et une autre concurrente appelée Formily, nous avons finalement opté pour Form.js pour les raisons suivantes :

- **Facilité d'apprentissage et de prise en main** : Contrairement à d'autres bibliothèques, Form.js est facile à utiliser grâce à la disponibilité de documentations et de guides sur Internet.
- **Gratuité** : Bien qu'il existe d'autres bibliothèques plus performantes que Form.js, comme SurveyJS, la plupart sont payantes.
- **Mise en page avancée** : Form.js offre la possibilité de modifier la position des widgets sur la page du formulaire ainsi que l'espace qu'ils occupent, offrant ainsi une grande liberté pour la mise en page.
- **Diversité des widgets proposés et possibilité d'extension** : Inclut des champs de texte, d'images, de dates/heure, des checkbox, des boutons radio, des listes déroulantes, et bien plus encore. Même si certains widgets nécessaires ne sont pas inclus, il est possible de les définir et de les rendre disponibles pour l'utilisateur final.
- **Logique conditionnelle et validation** : Il est possible de personnaliser le comportement du formulaire en ajoutant, par exemple, des conditions d'affichage pour certains widgets et en validant les entrées de l'utilisateur.
- **Construction simple de formulaires** : Grâce à son système de "Drag and Drop", Form.js rend la création de formulaires plus simple et intuitive pour les utilisateurs n'ayant pas nécessairement de connaissances en informatique.

3.1.5.2. Back-end : Express.js

Le back-end sera développé avec **Express.js** [6], un framework minimaliste pour Node.js permettant de gérer efficacement les requêtes API.

Nous avons choisi Express.js pour plusieurs raisons :

- **Léger et rapide** : Express.js est un framework simple à mettre en place et performant.
- **Facilité d'intégration avec SQLite** : Grâce au module sqlite3, Express permet une interaction fluide avec la base de données.

3.1.5.3. Base de Données : SQLite

Pour stocker les formulaires et les réponses des participants, nous avons initialement choisi **PostgreSQL** [7] pour sa robustesse et commencé l'implémentation de la plateforme avec ce SGBD lors de la troisième itération. Cependant, il s'est avéré que cette solution ajoutait une couche de complexité au moment du déploiement, car elle nécessitait l'installation de toute

l'infrastructure de la base de données sous-jacente. Après discussion avec notre encadrant et le client, nous avons donc décidé de basculer vers **SQLite** [8], qui présente plusieurs avantages, notamment :

- **Léger et autonome** : SQLite fonctionne sans serveur distinct, ce qui le rend facile à déployer et à intégrer dans des applications sans nécessiter une infrastructure complexe.
- **Fiabilité** : SQLite est reconnu pour sa fiabilité et sa robustesse, offrant une solution stable pour le stockage de données critiques.
- **Facilité d'intégration** : En tant que base de données embarquée, SQLite s'intègre facilement dans des applications existantes sans nécessiter de configuration lourde.

3.2. Troisième et quatrième itération

Dans cette partie, vous trouverez tout ce qui a été réalisé après la soutenance qui a eu lieu au mois de février (correspondant à la **phase 3B**).

Durant ces deux itérations, nous nous sommes principalement concentrés sur l'implémentation de la plateforme, son intégration avec Tobii Pro Lab, ainsi que sur la documentation.

3.2.1. Implémentation

3.2.1.1. La base de données

La base de données de la plateforme contient quatre tables :

- Une table **forms**, dans laquelle sont enregistrés les différents formulaires créés, chacun possède un identifiant unique, un titre, une date et une heure de création, une date et une heure de dernière modification, ainsi qu'un schéma JSON contenant sa structure et permettant de le générer.
- Une table **components**, contenant les différents composants (ou widgets) formant chaque formulaire. Un composant possède un identifiant unique, est associé à un formulaire spécifique grâce à une clé étrangère `form_id`, dispose d'un label, d'un type, d'une action (par exemple *submit* dans le cas d'un bouton), d'un `key_name` et d'un layout (indiquant sa mise en page et sa position dans le formulaire).
- Une table **responses**, dans laquelle sont enregistrées les réponses des participants aux différentes questions des formulaires. Une réponse est associée à un identifiant unique, à un formulaire spécifique grâce à une clé étrangère `form_id`, à un composant spécifique grâce à une clé étrangère `component_id`, à un participant spécifique grâce à une clé étrangère `user_id`. Elle possède également une valeur (la réponse saisie ou sélectionnée) et une date et une heure d'enregistrement.

- Une table **settings**, dans laquelle est enregistré l'id du participant par défaut.

Ci-après est le diagramme entité-relation de la base de données de la plateforme.

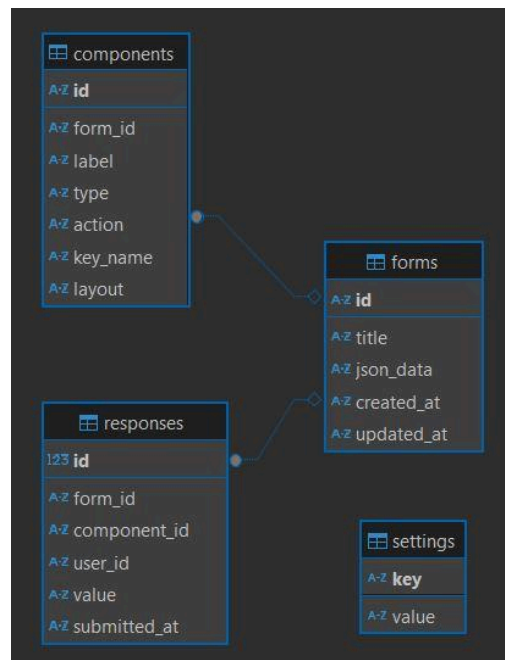


Figure 13 : Diagramme entité-relation de la base de données de la plateforme

Une notion fondamentale introduite par la librairie **form.js**, utilisée pour l'implémentation de la plateforme, et sur laquelle nous nous basons pour sauvegarder les formulaires créés et les générer après leur création, est celle du **schéma JSON du formulaire**.

Ce schéma est une représentation du formulaire, de ses composants, de leur mise en page (ainsi que de leurs identifiants), sous format JSON, selon la structure suivante :

```

{"components":[

{"label":"Textfield","type":"textfield","layout":{"row":"Row_0txg27r","columns":null},"id":"Field_070s0b2","key":"textfield_rocwl"},

{"label":"Number","type":"number","layout":{"row":"Row_046itkl","columns":null},"id":"Field_0g8xpov","key":"number_bhp49i"},

{"label":"Textarea","type":"textarea","layout":{"row":"Row_046itkl","columns":null},"id":"Field_1idcu8b","key":"textarea_sn0qj"},

{"label":"Button","action":"submit","type":"button","layout":{"row":"Row_0dlsesg","columns":null},"id":"Field_1y62q4z"},

{"type":"separator","layout":{"row":"Row_0sbjd66","columns":null},"id":"Field_1y0ijgl"},

{"subtype":"date","dateLabel":"Date","type":"datetime","layout":{"row":"Row_112rraf","columns":null},"id":"Field_0q64yj7","key":"datetime_2gl9m"},


```

```
{
  "label": "Number",
  "type": "number",
  "layout": {
    "row": "Row_Odqxbe4",
    "columns": null
  },
  "id": "Field_0xzw87z",
  "key": "number_a7hz8"
},
{
  "label": "Button",
  "action": "submit",
  "type": "button",
  "layout": {
    "row": "Row_0lckcsx",
    "columns": null
  },
  "id": "Field_16mmdtg"
}],
"type": "default",
"id": "Form_1ptsvm8",
"schemaVersion": 18
}
```

3.2.1.2. Le back-end

Le back-end de notre plateforme a été développé en Node.js en s'appuyant sur le framework Express.js pour la gestion des routes API et la communication avec la base de données. Il constitue la couche intermédiaire entre l'interface utilisateur (front-end) et la base de données SQLite.

- **Organisation du code**

L'implémentation back-end est organisée en plusieurs couches pour assurer la modularité et la maintenabilité :

1. **routes/** : contient tous les points d'entrée HTTP (API REST), organisés par fonctionnalité.
2. **controllers/** : traite la logique métier liée à chaque route.
3. **services/** : effectue les interactions directes avec la base de données SQLite.
4. **database/** : gère la création et l'ouverture de la base bd.db, ainsi que l'initialisation automatique des tables à partir du script bd.sql.

- **Fonctionnalités principales**

- **Gestion des formulaires**

Les formulaires sont stockés dans la table forms avec leur titre, leur structure JSON, et les dates de création/mise à jour. Les routes associées permettent de :

- ◆ Créer un formulaire (POST /api/save-form)
- ◆ Lire un formulaire (GET /api/forms/:id)
- ◆ Modifier un formulaire (PUT /api/forms/:id)
- ◆ Supprimer un formulaire et ses réponses (DELETE /api/forms/:id)

- ◆ Dupliquer un formulaire avec génération de nouveaux IDs (POST /api/forms/:id/duplicate)

→ Réponses des participants

Les réponses sont enregistrées dans la table responses en associant un participant, un formulaire, un composant et une valeur. Le backend offre les opérations suivantes :

- ◆ Sauvegarde automatique à chaque modification (POST /api/save-response)
- ◆ Soumission finale du formulaire (POST /api/submit-form)
- ◆ Consultation des réponses par formulaire (GET /api/forms/:id/responses)
- ◆ Consultation des réponses d'un participant (GET /api/form-responses-participant/:form_id/:user_id)
- ◆ Suppression de toutes les réponses d'un formulaire (DELETE /api/forms/:id/responses)

→ Gestion d'un identifiant utilisateur par défaut

- ◆ Pour permettre une compatibilité avec Tobii Pro Lab même sans spécification explicite de l'ID du participant dans l'URL, nous avons mis en place une solution:
- ◆ Une nouvelle table settings stocke des paires clé/valeur.
- ◆ Le champ defaultUserId y est enregistré et récupérable via :
 - GET /api/default-user-id
 - POST /api/default-user-id
- ◆ Ce mécanisme permet à l'expérimentateur de configurer un ID par défaut depuis la page d'accueil, qui sera automatiquement injecté dans l'URL lorsqu'il est absent ou remplacé par un @.

→ Gestion des erreurs et robustesse

- ◆ Toutes les requêtes critiques du back-end sont encapsulées dans des promesses avec gestion des erreurs. Les transactions SQL sont utilisées pour les opérations atomiques comme la suppression complète d'un

formulaire (supprimer le formulaire, ses composants et les réponses associées).

3.2.1.3. Le front-end

La plateforme que nous avons implémentée est composée de différentes pages, chacune permettant d'effectuer des tâches spécifiques et destinée à un acteur particulier.

- **La page d'accueil**

Il s'agit de la première page qui s'affiche lorsque l'expérimentateur accède à notre plateforme.

Cette page comporte un bouton permettant d'accéder à la page de création de formulaires, afin de créer un nouveau formulaire.

Elle présente aussi un champ de texte dans lequel l'expérimentateur peut saisir l'**id du participant par défaut**. Cet ID sera ajouté dynamiquement à l'URL d'accès d'un formulaire si celui-ci contient le symbole-clé @ à la place de l'ID du participant.

Ainsi, une URL de la forme :

`http://localhost:3000/form-viewer/ID_du_formulaire/numéro_de_la_page/@`

sera dynamiquement remplacée par :

`http://localhost:3000/form-viewer/ID_du_formulaire/numéro_de_la_page/ID_participant_par_défaut`

Cela s'avère utile dans les cas où l'expérimentateur fait passer les participants un par un lors de chaque test. Avant de lancer un test sur une machine, il lui suffit alors de saisir l'ID du participant dans ce champ de texte pour que les formulaires remplis soient automatiquement associés à cet utilisateur. Lorsqu'un nouveau participant se présente, l'expérimentateur n'aura qu'à modifier l'ID par défaut dans ce même champ.

Cette page contient également la liste des formulaires créés précédemment. Pour chaque formulaire on peut voir : son identifiant, son titre, sa date et heure de création, la date et heure de la dernière modification, ainsi que cinq boutons :

- Voir : permet d'accéder à une page de consultation du formulaire.
- Modifier : permet d'accéder à une page pour modifier le formulaire, si cela est possible.
- Voir réponses : permet d'accéder à une page affichant la liste des réponses des participants aux différentes questions du formulaire.
- Dupliquer : Permettant de dupliquer le formulaire.
- Supprimer : Permettant de supprimer le formulaire.

XP-Forms

Créer un nouveau formulaire

ID participant par défaut :

Sauvegarder

Liste des formulaires enregistrés

ID	Titre	Date de création	Dernière mise à jour	Actions
Form_1qbjzz1	Form testing 2	29/04/2025 15:12:14	29/04/2025 15:12:14	<div>Voir</div> <div>Modifier</div> <div>Voir Réponses</div> <div>Dupliquer</div> <div>Supprimer</div>
Form_1ud0tsg	Form demo 3	29/04/2025 15:11:54	29/04/2025 15:11:54	<div>Voir</div> <div>Modifier</div> <div>Voir Réponses</div> <div>Dupliquer</div> <div>Supprimer</div>
Form_1ab5dvu	Form demo 5	29/04/2025 10:52:42	29/04/2025 10:53:31	<div>Voir</div> <div>Modifier</div> <div>Voir Réponses</div> <div>Dupliquer</div> <div>Supprimer</div>

Figure 14 : Page d'accueil de la plateforme

- La page de création de formulaires

La page de création de formulaires permet de construire un formulaire en ajoutant les composants (ou *widgets*) nécessaires. Elle est composée de trois parties :

- À gauche, on trouve les différents composants pouvant être ajoutés au formulaire, en les faisant glisser vers la zone centrale.
- Au centre, se trouve le formulaire en cours de création.
- À droite, sont affichés des paramètres spécifiques permettant de modifier le comportement et les propriétés du composant sélectionné (ou du formulaire en général si aucun composant n'est sélectionné), tels que l'affichage conditionnel, les règles de validation ou encore le caractère obligatoire d'une réponse.

Nous avons utilisé la librairie **Form.js** pour implémenter cette fonctionnalité. Grâce à son système de **Drag and Drop**, elle rend la création de formulaires plus simple et intuitive pour les expérimentateurs. Elle permet également de modifier la position des widgets sur la page du formulaire ainsi que l'espace qu'ils occupent, offrant ainsi une grande liberté en matière de mise en page.

Form.js propose une large panoplie de composants, notamment des champs de texte, d'images, de dates, des cases à cocher (*checkbox*), des boutons radio, des listes déroulantes, et bien plus encore. Même si certains widgets nécessaires ne sont pas inclus par défaut, il est possible de les définir et de les rendre disponibles à l'utilisateur final.

Nous avons également ajouté la possibilité pour l'expérimentateur de saisir un **titre** pour le formulaire, à l'aide d'un champ de texte situé en haut de la page.

Enfin, pour créer un formulaire composé de plusieurs pages (**formulaire multipage**), il suffit d'ajouter le composant appelé "**separator**". Comme son nom l'indique, ce composant permet de séparer les différentes pages d'un même formulaire. Dans l'image ci-après de la page de création de formulaires, le séparateur est représenté par une ligne horizontale grise qui divise visuellement deux pages du formulaire. Lors de la consultation du formulaire, chaque page sera alors affichée séparément.

Figure 15 : Page de création de formulaires

Dans l'image ci-après, on peut voir le reste des composants qui ne sont pas présents dans l'image précédente de la page de création de formulaires.

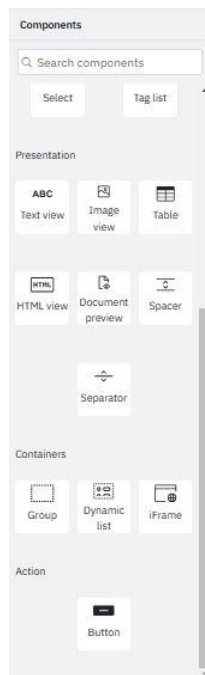


Figure 16 : Le reste des composants proposés dans la page de création de formulaires

Dans l'image suivante, on peut voir que lorsqu'un composant du formulaire est sélectionné, il est possible de le supprimer. À droite, apparaissent les paramètres spécifiques à ce composant, que l'on peut modifier.

Figure 17 : Page de création de formulaires (un composant sélectionné)

- **La page de modification de formulaires**

On accède à cette page lorsqu'on choisit de modifier un formulaire. Elle présente le formulaire sélectionné et offre la possibilité d'apporter toute modification souhaitée à ce dernier.

Remarque : si un formulaire a déjà été rempli par un participant, il devient impossible de le modifier.

La figure ci-après est une capture d'écran de la page de modification de formulaires.

Figure 18 : Page de modification de formulaires

ID	Titre	Date de création
Form_1owu7ng	Form demo 2	29/04/2025 15:26:29
Form_1qbjzz1	Form testing 2	29/04/2025 15:12:14
Form_1ud0tg	Form demo 3	29/04/2025 15:11:54
Form_1ab5dru	Form demo 5	29/04/2025 10:52:42

Figure 19 : Pop-up affiché lorsqu'on essaye de modifier un formulaire déjà rempli par un participant

- **La page de consultation de formulaires (vue de l'expérimentateur)**

Dans cette page, l'expérimentateur peut consulter un formulaire. En haut de la page, des instructions lui indiquent quelle URL saisir dans Tobii Pro Lab pour que le participant soit dirigé vers la page correspondante du formulaire, dans le cadre d'un scénario d'expérimentation (voir l'explication de l'utilisation des URLs pour la navigation entre notre plateforme et Tobii Pro Lab [ici](#)).

On peut également remarquer la présence des boutons "Page suivante" et "Page précédente", permettant de naviguer d'une page à l'autre dans le cas d'un formulaire multipage.

Cette page présente deux vues :

- Une **vue pour l'expérimentateur**, accessible via une URL de la forme :
`http://localhost:3000/form-viewer/ID_du_formulaire/numéro_de_la_page`
- Une **vue pour le participant**, qui est amené à remplir le formulaire, accessible via une URL de la forme :
`http://localhost:3000/form-viewer/ID_du_formulaire/numéro_de_la_page/ID_du_participant`

La figure suivante présente la vue de l'expérimentateur de la page de consultation de formulaires.

Form Viewer

[Retour à l'accueil](#)

ID du Formulaire : Form_1owu7ng

Date de Création : 29/04/2025 15:26:29

Pour intégrer dans un scénario Tobii utilisez : http://localhost:3000/form-viewer/Form_1owu7ng/1/id_participant

Ajoutez @ comme ID participant pour utiliser l'ID utilisateur par défaut.

Ajoutez ?navigation=True à la fin si vous voulez permettre la navigation entre pages.

Page : 1 / 2
[Page suivante](#)

Question 1 : Genre

☐ Homme

☐ Femme

Question 2 : Date de naissance

Question 3 : Prénom

Question 4 : Nom

Question 5 : Raconte une histoire

[Soumettre](#)

Figure 20 : Page de consultation de formulaires, vue de l'expérimentateur, (page 1 du formulaire)

Form Viewer

[Retour à l'accueil](#)

ID du Formulaire : Form_1owu7ng

Date de Création : 29/04/2025 15:26:29

Pour intégrer dans un scénario Tobii utilisez : http://localhost:3000/form-viewer/Form_1owu7ng/2/id_participant

Ajoutez @ comme ID participant pour utiliser l'ID utilisateur par défaut.

Ajoutez ?navigation=True à la fin si vous voulez permettre la navigation entre pages.

[Page précédente](#)
Page : 2 / 2

Question 6 : Age

Question 7 : Profession

[Soumettre](#)

Figure 21 : Page de consultation de formulaires, vue de l'expérimentateur, (page 2 du formulaire)

- **La page de consultation de formulaires (vue du participant)**

Dans cette vue, le participant est amené à remplir le formulaire affiché, ou une page spécifique de celui-ci.

Grâce à la fonctionnalité de **sauvegarde automatique**, les réponses saisies par le participant sont enregistrées au fur et à mesure du remplissage, plus précisément à chaque fois qu'un champ est modifié (événement **onchange**). Cela garantit qu'aucune réponse ne sera perdue en cas de panne ou d'imprévu.

Comme on peut le voir dans les deux figures ci-après, lorsqu'on accède à la page via une URL de la forme :

`http://localhost:3000/form-viewer/ID_du_formulaire/numéro_de_la_page/ID_du_participant?navigation=True`

le participant a la possibilité de naviguer entre les pages du formulaire.

En revanche, si l'on retire le paramètre **?navigation=True** de l'URL, la navigation devient impossible. Cela est utile si l'on souhaite que le participant remplisse différentes pages du formulaire à des moments distincts de l'expérimentation.

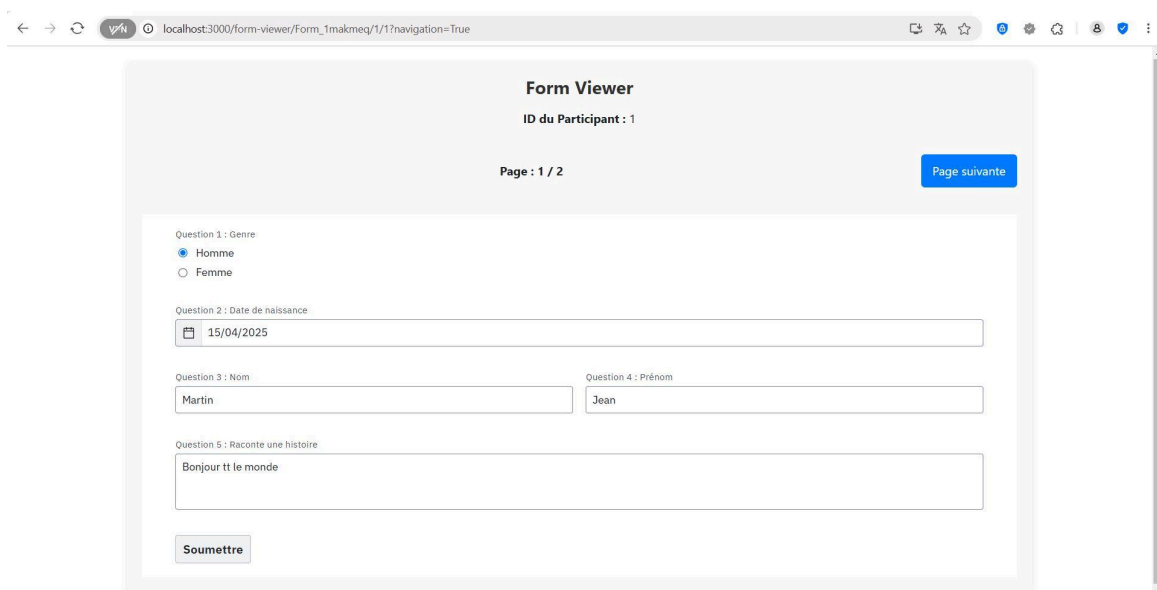


Figure 22 : Page de consultation de formulaires, vue du participant, (avec option de navigation)

Form Viewer

ID du Participant : 1

Question 1 : Genre
☒ Homme
☐ Femme

Question 2 : Date de naissance

Question 3 : Nom

Question 4 : Prénom

Question 5 : Raconte une histoire

Soumettre

Figure 23 : Page de consultation de formulaires, vue du participant, (sans option de navigation)

- **La page de consultation des réponses à un formulaire**

Dans cette page, on trouve un tableau contenant les réponses des participants ayant rempli le formulaire concerné. Chaque colonne représente une question du formulaire et chaque ligne correspond aux réponses d'un participant (identifié par son ID).

On y trouve un bouton permettant de **télécharger le tableau au format CSV**, ainsi qu'un bouton **"Supprimer toutes les réponses"**, permettant d'effacer toutes les réponses présentes dans le tableau.

Vous pouvez voir dans la figure ci-après une capture d'écran de cette page.

Réponses du formulaire

[Retour à l'accueil](#)

[Exporter en CSV](#)

ID Utilisateur		Question 7 : Profession	Question 5 : Raconte une histoire	Question 6 : Age	Question 4 : Prénom	Question 3 : Nom	Question 1 : Genre
1	2025-04-15	Prof	Bonjour tt le monde	50	Jean	Martin	value
2	2025-04-22	étudiante	Hello world	20	Marie	Pierre	value2
3	2025-04-01	Boxeur	Non	40	John	Jones	value
Kevin	2025-04-01	Prof	Hello	55	Kevin	Martin	value

[Supprimer toutes les réponses](#)

Figure 24 : Page de consultation des réponses à un formulaire

Comme vous pouvez le voir dans la figure ci-dessous le bouton “Exporter en CSV” présent dans cette page nous permet de télécharger sous format CSV le tableau des réponses.

Réponses du formulaire

Retour à l'accueil

Exporter en CSV

ID Utilisateur		Question 7 : Profession	Question 5 : Raconte une histoire	Question 6 : Age	Question 4 : Prénom	Question 3 : Nom	Question 1 : Genre
1	2025-04-15	Prof	Bonjour tt le monde	50	Jean	Martin	value
2	2025-04-22	étudiante	Hello world	20	Marie	Pierre	value2
3	2025-04-01	Boxeur	Non	40	John	Jones	value
Kevin	2025-04-01	Prof	Hello	55	Kevin	Martin	value

Supprimer toutes les réponses

Figure 25 : Page de consultation des réponses à un formulaire (tableau des réponses téléchargé)

• La duplication d’un formulaire

Dans la page d’accueil, chaque formulaire créé peut être dupliqué via le bouton “**Dupliquer**” correspondant. Cette action génère une copie du formulaire avec les mêmes composants et la même mise en page, mais attribue un nouvel ID au formulaire dupliqué ainsi qu’à chacun de ses composants.

Les **réponses** des participants **ne sont pas dupliquées** et ne sont donc pas associées au nouveau formulaire.

Cela permet à l’expérimentateur de créer rapidement un nouveau formulaire similaire à un formulaire existant, sans devoir recommencer de zéro. Il lui suffit alors de modifier la copie obtenue.

Vous pouvez voir dans la figure suivante un formulaire dupliqué intitulé “Form demo 2 (copy)”, qui est une copie du formulaire “Form demo 2”.

XP-Forms

Créer un nouveau formulaire

ID participant par défaut :

Kev

Sauvegarder

Liste des formulaires enregistrés

ID	Titre	Date de création	Dernière mise à jour	Actions
Form_0rew5klb	Form demo 2 (copy)	29/04/2025 15:32:33	29/04/2025 15:32:33	<button>Voir</button> <button>Modifier</button> <button>Voir Réponses</button> <button>Dupliquer</button> <button>Supprimer</button>
Form_1owu7ng	Form demo 2	29/04/2025 15:26:29	29/04/2025 15:26:29	<button>Voir</button> <button>Modifier</button> <button>Voir Réponses</button> <button>Dupliquer</button> <button>Supprimer</button>
Form_1qbjzz1	Form testing 2	29/04/2025 15:12:14	29/04/2025 15:12:14	<button>Voir</button> <button>Modifier</button> <button>Voir Réponses</button> <button>Dupliquer</button> <button>Supprimer</button>
Form_1ud0tsg	Form demo 3	29/04/2025 15:11:54	29/04/2025 15:11:54	<button>Voir</button> <button>Modifier</button> <button>Voir Réponses</button> <button>Dupliquer</button> <button>Supprimer</button>
Form_1ab5dvu	Form demo 5	29/04/2025 10:52:42	29/04/2025 10:53:31	<button>Voir</button> <button>Modifier</button> <button>Voir Réponses</button> <button>Dupliquer</button> <button>Supprimer</button>

Figure 26 : Page d'accueil contenant un formulaire dupliqué

- La suppression d'un formulaire

Dans la page d'accueil, chaque formulaire créé peut être supprimé en cliquant sur le bouton **“Supprimer”** correspondant.

Un pop-up s'affiche alors, demandant à l'utilisateur s'il est certain de vouloir supprimer le formulaire, tout en l'informant que les réponses associées seront également supprimées.

Si l'utilisateur confirme son choix, le formulaire, ainsi que toutes ses réponses, seront définitivement supprimés.

La capture d'écran suivante montre l'alerte affichée suite au clic sur **“Supprimer”** pour l'un des formulaires.

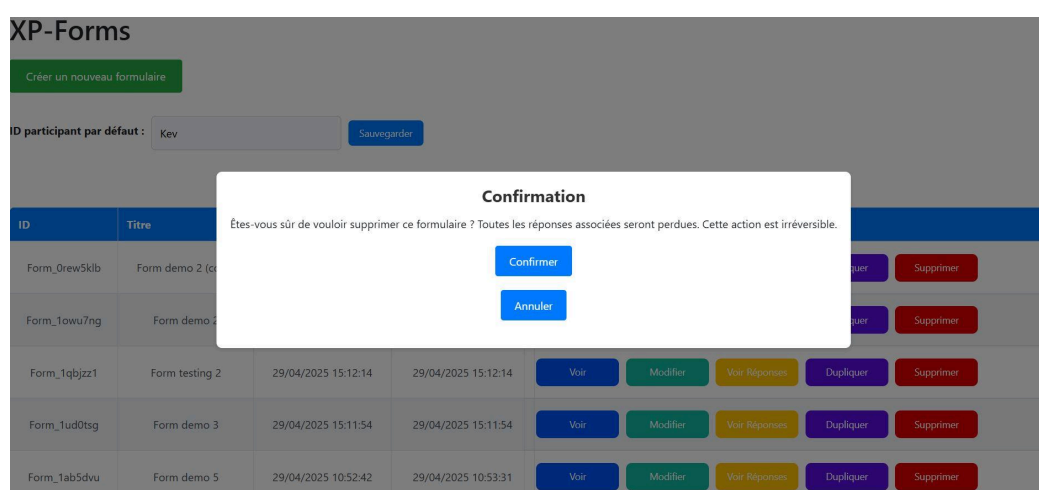


Figure 27 : Page d'accueil, l'alerte affichée suite au choix de supprimer un formulaire

3.2.2. L'intégration avec Tobii Pro Lab

Dans la page de Tobii Pro Lab visible dans la capture d'écran ci-après, l'expérimentateur peut préparer un **scénario d'expérimentation** à faire passer à différents participants.

Ce scénario peut inclure des visites à la plateforme que nous avons développée, afin de recueillir les réponses des participants à certaines questions, par exemple après l'exposition à un stimulus.

Comme illustré dans la figure, l'expérimentateur a préparé un scénario comprenant une première visite à la plateforme, un stimulus sous forme d'image, puis une seconde visite à la plateforme.

À droite de la page, lorsqu'une des visites à notre plateforme est sélectionnée, un champ de texte intitulé **"URL"** apparaît. C'est dans ce champ que l'on saisit l'URL menant à la page du formulaire à remplir à cette étape du scénario.

On remarque que l'URL se termine par l'ID du participant, ici "Kevin". Il aurait également été possible d'utiliser le symbole **@** à la place de l'ID pour accéder au formulaire en tant que **participant par défaut**, comme expliqué précédemment.

On peut aussi remarquer qu'il est possible de définir une **durée limite** à chaque étape, au terme de laquelle Tobii passe automatiquement au stimulus suivant.

On peut également appuyer sur la touche **F10** pour passer manuellement au stimulus suivant sans attendre la fin du compte à rebours.

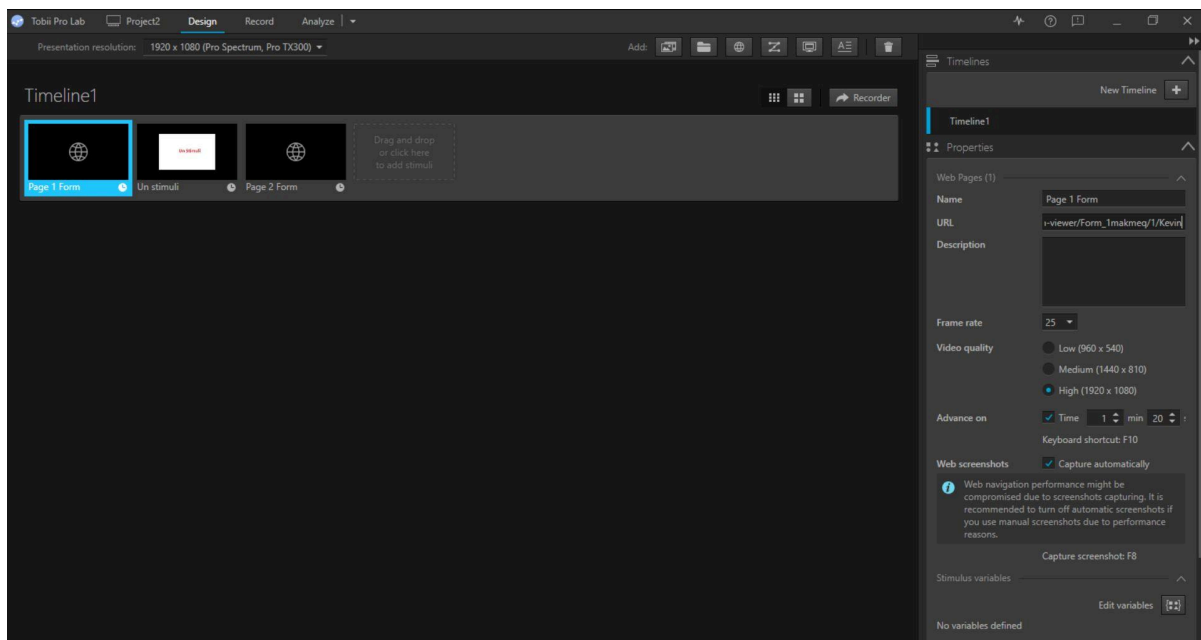


Figure 28 : Page d'édition des scénarios d'expérimentation sur Tobii Pro Lab

Après avoir préparé un scénario d'expérimentation, l'expérimentateur saisit les noms (considérés comme identifiants) des participants, puis clique sur **"Start Recording"** pour lancer l'expérimentation et commencer l'enregistrement de ce qui se passe à l'écran.

Comme le montre la capture d'écran suivante, le participant nommé Kevin a été ajouté.

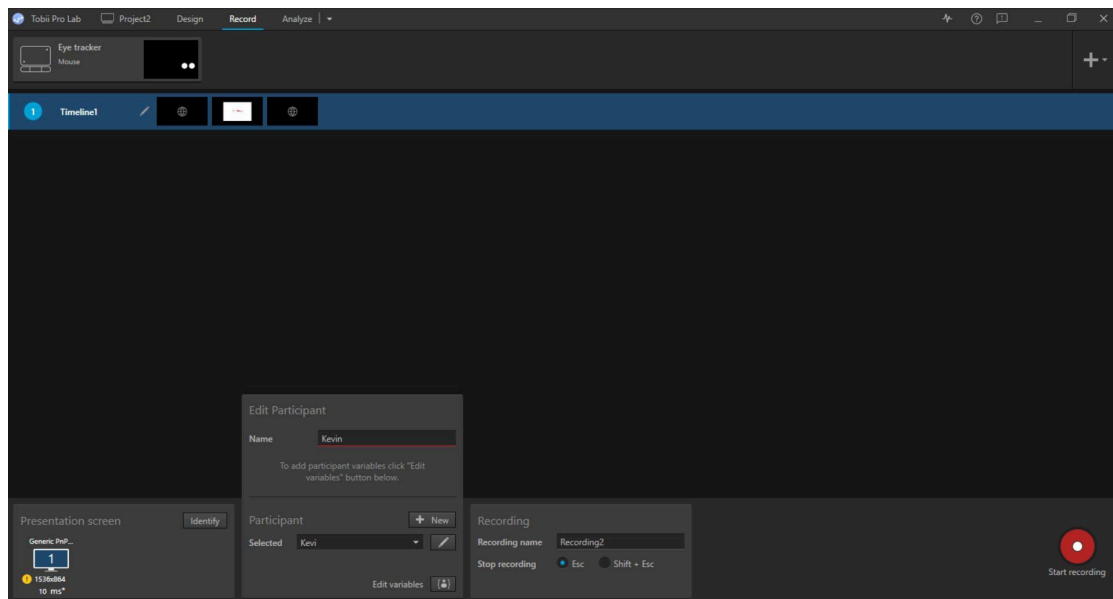


Figure 29 : Page de l'ajout des participants et lancement du scénario sur Tobii Pro Lab

Dans la capture d'écran suivante, on trouve la page de Tobii Pro Lab contenant les enregistrements d'écran (Recordings) effectués lors du passage des participants par le scénario d'expérimentation.

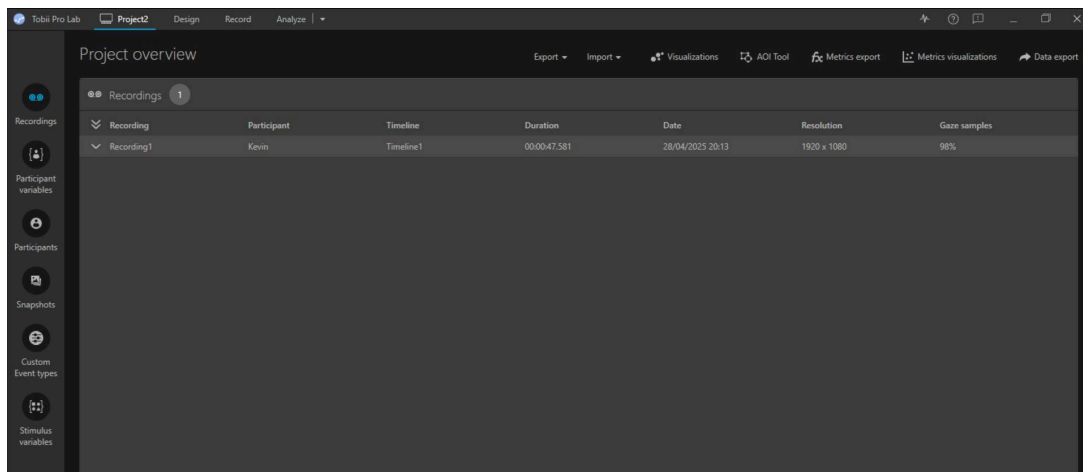


Figure 30 : Page des enregistrement des passages des participants sur Tobii Pro Lab

Ce lien vous mènera vers un exemple d'enregistrement vidéo effectué lors du passage d'un participant par un scénario d'expérimentation.

Dans cette vidéo, vous pouvez voir comment l'on effectue des allers-retours entre Tobii Pro Lab (pour visualiser les stimuli) et notre plateforme (pour remplir les pages du formulaire).

https://drive.google.com/file/d/1ow_aZETP3UjnZ4ykJOKRhXuohza85jnX/view?usp=sharing

4. Conclusion

À l'issue de la phase 3B de notre projet, composée des deux dernières itérations, nous sommes en mesure d'affirmer que nous avons réussi à implémenter une plateforme fonctionnelle répondant parfaitement aux besoins du client, compatible avec les outils qu'il utilise, notamment Tobii Pro Lab. Le code est bien organisé, commenté, et accompagné d'une documentation claire couvrant l'ensemble des aspects de la plateforme, le tout livré dans les délais prévus.

En adoptant une approche Agile, nous avons su planifier efficacement les différentes itérations et répartir les tâches de manière optimale entre les membres de l'équipe.

Nous nous sommes appuyés sur les réalisations des deux premières itérations, notamment la conception UML de la solution retenue, qui nous a permis de traduire les exigences du client en une modélisation technique claire ; la définition de la structure sous-jacente de la base de données à travers le diagramme de classes ; la description textuelle des différents scénarios d'utilisation ; ainsi que le prototype IHM illustrant les interfaces à implémenter et les comportements attendus de la solution.

Nous avons ensuite procédé à l'implémentation de la plateforme. Comme décrit en détail dans ce rapport, nous avons soigneusement présenté ses différents composants, à savoir la base de données, la partie back-end, la partie front-end, ainsi que l'intégration de la plateforme avec Tobii Pro Lab.

Enfin, suite à la soutenance prévue début mai, nous comptons élaborer la recette de la plateforme implémentée, dans laquelle nous validerons son bon fonctionnement en vérifiant que les différentes fonctionnalités produisent les résultats attendus.

5. Bibliographie scientifique et technologique

5.1. Bibliographie scientifique

- **Importance de la recherche expérimentale et des interfaces utilisateur adaptées**

L'expérience utilisateur (UX) joue un rôle fondamental dans la conception d'outils interactifs de collecte de données, en particulier dans le cadre de la recherche expérimentale. Lorsqu'un participant interagit avec un formulaire, plusieurs facteurs influencent la qualité et la fiabilité des données recueillies : la clarté des questions, la fluidité de l'interface et la simplicité de navigation. Une mauvaise conception d'interface peut induire des biais cognitifs, fausser les réponses ou encore provoquer un abandon prématuré du questionnaire. Les études en interaction humain-machine (HCI) montrent que plus une interface est intuitive et bien pensée, plus les participants sont enclins à fournir des réponses précises et cohérentes.

Dans ce contexte, il est essentiel de concevoir des outils de collecte de données qui minimisent la charge cognitive des utilisateurs tout en maximisant leur engagement. L'adoption des principes de l'UX dans la recherche expérimentale contribue à l'amélioration des résultats en assurant une meilleure qualité des réponses obtenues. Des chercheurs ont démontré que la perception d'un outil comme étant "facile à utiliser" influence directement le comportement des participants et leur persévérance dans l'expérience.

- Source : Hassenzahl, M., & Tractinsky, N. (2006). User experience – A research agenda. Behaviour & Information Technology, 25(2), 91-97.

- **Expérience utilisateur et ergonomie des formulaires interactifs**

Les formulaires interactifs sont des outils clés pour la collecte de données en recherche expérimentale, mais leur efficacité dépend fortement de leur conception. Une mauvaise organisation des éléments visuels, une structuration confuse des questions ou une surcharge d'informations peuvent entraîner une fatigue cognitive et influencer les réponses des participants. Les principes d'ergonomie applicables aux formulaires suggèrent qu'un bon questionnaire doit être structuré de manière progressive, avec un enchaînement logique des questions, des instructions claires et une interface épurée.

Des recherches en psychologie cognitive indiquent que l'utilisateur est plus à l'aise lorsqu'il comprend immédiatement la manière dont il doit interagir avec un formulaire. Une étude réalisée par Redish et Chisnell (2004) a montré que des formulaires mal conçus entraînent une baisse significative du taux de réponse et une augmentation des erreurs dans les données collectées. En optimisant la mise en page, la lisibilité et la fluidité des transitions entre les sections, il est possible de rendre l'expérience plus naturelle et intuitive, améliorant ainsi la qualité des résultats.

- Source : Redish, J., & Chisnell, D. (2004). Designing forms that people can use. UXPA Journal.

- **Méthodes et outils de collecte de données expérimentales**

La recherche expérimentale s'appuie sur différentes techniques pour collecter des données auprès des participants. Les formulaires classiques sur papier ont progressivement été remplacés par des formulaires numériques interactifs, qui offrent de nombreux avantages : flexibilité, automatisation des analyses et réduction des erreurs humaines. Cependant, tous les outils ne sont pas adaptés aux exigences de la recherche scientifique. Des solutions comme Qualtrics, Google Forms, et SphinxOnline permettent de concevoir des questionnaires simples, mais manquent souvent de fonctionnalités avancées, comme l'intégration avec des équipements de laboratoire ou la personnalisation graphique des interfaces.

Une étude comparative menée par Funke et Reips (2012) a mis en évidence que les expérimentations réalisées en ligne, via des formulaires numériques, produisent des résultats similaires aux expérimentations en laboratoire si la plateforme est bien optimisée. Toutefois, des biais peuvent survenir, notamment lorsque l'expérience repose sur une interface mal calibrée ou lorsque les participants rencontrent des difficultés techniques. Ainsi, il est crucial d'évaluer les différentes solutions disponibles en fonction des besoins spécifiques de l'expérimentation.

- Source : Funke, J., & Reips, U. D. (2012). Data collection in experimental psychology: A comparison of laboratory and online methods. Behavior Research Methods, 44, 249–271.

- **Oculométrie et analyse comportementale dans la recherche expérimentale**

L'oculométrie (eye-tracking) est une technique largement utilisée en psychologie cognitive et en ergonomie des interfaces pour analyser l'attention des utilisateurs et comprendre comment ils interagissent avec les interfaces numériques. Grâce aux avancées technologiques, des outils comme Tobii Pro Lab permettent aujourd'hui d'obtenir des données précises sur les points d'intérêt visuel et la fluidité de la navigation sur une interface donnée.

Dans le contexte de la recherche expérimentale, l'oculométrie peut être utilisée pour identifier les éléments de l'interface qui attirent le plus l'attention, ainsi que les éventuelles difficultés rencontrées par les utilisateurs. Une étude de Holmqvist et al. (2011) a démontré que les données oculométriques permettent non seulement d'évaluer l'ergonomie des interfaces, mais aussi d'anticiper les erreurs de compréhension ou d'interaction. L'intégration de ces outils dans la conception des formulaires expérimentaux permettrait d'optimiser leur utilisation et d'améliorer l'expérience des participants.

- Source : Holmqvist, K., Nyström, M., Andersson, R., Dewhurst, R., Jarodzka, H., & van de Weijer, J. (2011). Eye Tracking: A Comprehensive Guide to Methods and Measures. Oxford University Press.

- **Sécurité et éthique dans la collecte de données expérimentales**

La collecte de données en recherche scientifique est soumise à des contraintes éthiques et légales afin de garantir la protection des participants et la confidentialité des informations recueillies. Avec l'entrée en vigueur du Règlement Général sur la Protection des Données (RGPD) en Europe, les chercheurs doivent s'assurer que les formulaires utilisés respectent les principes de minimisation des données, d'anonymisation et de sécurité.

Un rapport de la Commission Européenne (2018) souligne l'importance d'informer les participants sur l'usage qui sera fait de leurs données et de leur garantir un droit de retrait à tout moment. Dans le cadre des expérimentations UX, il est également crucial de limiter la collecte aux seules informations nécessaires et de garantir que les réponses ne puissent pas être exploitées à des fins autres que la recherche. Ces enjeux éthiques imposent aux chercheurs et aux développeurs de mettre en place des pratiques rigoureuses pour assurer la conformité de leurs outils.

- Source : European Union. (2018). General Data Protection Regulation (GDPR): Guidelines for Research Data Management.

5.2. Bibliographie technologique

- **Frameworks et technologies backend**

Le choix du framework backend joue un rôle essentiel dans la conception d'une application web. Pour ce projet, plusieurs technologies ont été envisagées : **Flask (Python)** [13], **Express.js** et **NestJS (Node.js)** [14].

- **Flask** est un micro-framework léger et rapide à mettre en place, particulièrement adapté aux petites applications. Il est apprécié pour sa simplicité et sa flexibilité, ce qui le rend idéal pour un projet nécessitant un démarrage rapide avec peu de configuration.
- **Express.js** est une solution minimaliste pour le développement d'API en JavaScript, offrant une grande flexibilité dans la gestion des routes et des middlewares. Il est utilisé dans de nombreuses applications modernes grâce à son intégration avec l'écosystème Node.js.
- **NestJS** repose sur une architecture modulaire et utilise **TypeScript** pour structurer le code de manière plus robuste, ce qui le rend idéal pour des applications de grande envergure et évolutives.

Le choix du framework doit prendre en compte plusieurs critères : la courbe d'apprentissage, la flexibilité, la performance et la compatibilité avec les autres outils de l'écosystème de l'application. Flask, bien que simple, peut nécessiter des ajouts pour répondre à des besoins avancés. Express.js est extrêmement flexible mais demande une gestion rigoureuse du middleware, tandis que NestJS impose une structure stricte qui facilite la scalabilité à long terme.

- **Bases de données et stockage des réponses**

Le projet nécessite un stockage efficace des données issues des formulaires. Deux bases de données principales ont été considérées : **SQLite** et **PostgreSQL**.

- **SQLite** est une base de données légère et sans configuration, idéale pour les tests et les applications locales. Elle stocke les données sous forme de fichiers et ne nécessite pas de serveur dédié, ce qui la rend facile à intégrer dans des projets de petite échelle.
- **PostgreSQL** est une base de données relationnelle robuste, adaptée aux projets de grande envergure. Elle offre des fonctionnalités avancées, comme la gestion des index et des requêtes complexes, qui sont essentielles pour assurer de bonnes performances sur des volumes de données importants.

Le choix entre ces deux bases dépend du volume de données attendu et des besoins en requêtage avancé. **PostgreSQL** offre des fonctionnalités puissantes pour la gestion des relations complexes et des indexations avancées, tandis que **SQLite** se distingue par sa simplicité de mise en place et sa légèreté.

- **Bibliothèques front-end pour la création de formulaires interactifs**

Pour permettre aux expérimentateurs de concevoir des formulaires dynamiques de manière intuitive, plusieurs bibliothèques spécialisées ont été analysées. Parmi elles, **Form-js** et **Formily** [\[15\]](#) se distinguent par leurs capacités de **drag-and-drop**, facilitant la création et l'organisation des champs de saisie.

- **Form-js** est une bibliothèque JavaScript qui permet de créer des formulaires interactifs et réactifs avec une API flexible. Elle est particulièrement utile pour intégrer rapidement des fonctionnalités de saisie de données sans nécessiter un développement complexe.
- **Formily** est une solution plus avancée qui repose sur une architecture déclarative et modulaire. Elle est conçue pour s'intégrer efficacement dans des applications React et permet une grande flexibilité dans la personnalisation des composants.

Ces outils permettent de générer des interfaces utilisateurs ergonomiques et réactives, réduisant ainsi le temps de développement et garantissant une meilleure expérience pour les utilisateurs finaux.

- **OpenSesame et Tobii Pro Lab : Présentation et Intégration**

OpenSesame

OpenSesame est un logiciel open-source conçu pour créer des expériences expérimentales en psychologie cognitive, neurosciences et sciences du comportement. Il permet aux chercheurs de concevoir facilement des protocoles d'expérimentation avec des stimuli, des questionnaires et des interactions spécifiques aux participants.

- L'un des avantages majeurs d'OpenSesame est qu'il offre une interface graphique intuitive tout en permettant une personnalisation avancée grâce au langage **Python**.

- Il prend en charge plusieurs types de dispositifs expérimentaux, y compris l'intégration avec **Tobii Pro Lab** pour l'oculométrie.
- OpenSesame propose une gestion simplifiée des sessions expérimentales, ce qui est particulièrement utile pour les études en laboratoire et les collectes de données en milieu contrôlé.

Dans le cadre de ce projet, OpenSesame peut être utilisé pour présenter les formulaires aux participants et collecter leurs réponses de manière synchronisée avec d'autres mesures expérimentales.

Tobii Pro Lab

Tobii Pro Lab est un logiciel avancé permettant d'analyser les mouvements oculaires des participants dans des expériences UX et en recherche cognitive. Il est utilisé pour comprendre comment un utilisateur interagit avec une interface et où son attention est portée.

- Tobii Pro Lab permet d'enregistrer et d'analyser des **données d'oculométrie**, telles que la durée de fixation sur certains éléments et la trajectoire du regard.
- Il est particulièrement utile dans l'étude des **formulaires interactifs**, car il permet d'évaluer si certaines questions ou sections posent des difficultés aux participants.
- Le logiciel peut être synchronisé avec **OpenSesame**, ce qui permet d'enregistrer des données combinées (réponses au questionnaire + données oculométriques).

L'intégration avec Tobii Pro Lab offre une opportunité unique d'analyser l'ergonomie et la lisibilité des formulaires, ce qui pourrait être exploité dans des expérimentations futures pour améliorer l'expérience utilisateur.

5.3. Références bibliographiques complètes

[1] Tobii Pro Lab, "Tobii Pro Lab - Eye tracking software," Available:

<https://www.tobiipro.com/product-listing/tobii-pro-lab/>.

[2] OpenSesame Documentation, "Open-source software for behavioral experiments,"

Available: <https://osdoc.cogsci.nl/>.

[3] K. Holmqvist, M. Nyström, R. Andersson, R. Dewhurst, H. Jarodzka, and J. van de Weijer, *Eye Tracking: A Comprehensive Guide to Methods and Measures*. Oxford University Press, 2011.

[4] Form-js Documentation, "Form-js: A library for building interactive forms," Available:

<https://form-js.dev/>.

[5] Créer de nouveaux composants Form-js, "Create Custom Form Components", Available:

<https://bpmn.io/blog/posts/2023-custom-form-components>

[6] Express.js Documentation, "Fast, unopinionated, minimalist web framework for Node.js,"

Available: <https://expressjs.com/>.

- [7] PostgreSQL Documentation, "PostgreSQL: The world's most advanced open source database," Available: <https://www.postgresql.org/>.
- [8] SQLite Documentation, "SQLite Home Page," Available: <https://www.sqlite.org/index.html>.
- [9] M. Hassenzahl and N. Tractinsky, "User experience – A research agenda," *Behaviour & Information Technology*, vol. 25, no. 2, pp. 91–97, 2006.
- [10] J. Redish and D. Chisnell, "Designing forms that people can use," *UXPA Journal*, 2004.
- [11] J. Funke and U. D. Reips, "Data collection in experimental psychology: A comparison of laboratory and online methods," *Behavior Research Methods*, vol. 44, pp. 249–271, 2012.
- [12] European Union, *General Data Protection Regulation (GDPR): Guidelines for Research Data Management*, 2018.
- [13] Flask Documentation, "Welcome to Flask documentation," Available: <https://flask.palletsprojects.com/>.
- [14] NestJS Documentation, "A progressive Node.js framework for building efficient, reliable, and scalable server-side applications," Available: <https://nestjs.com/>.
- [15] Formily Documentation, "Formily: Unified Form Solution," Available: <https://formilyjs.org/>.