

---

# Rapport de Micro-Projet Application Web - FixNow

---

ARIANE ZHANG  
XUE YANG

UE PC3R  
MASTER INFORMATIQUE

*Tuteur(s) université :*  
ROMAIN DEMANGEON

avril 2024

## Table des matières

<b>1</b>	<b>Sujet du Projet</b>	<b>2</b>
1.1	Fournisseurs de services (réparateurs) . . . . .	2
1.2	Demandeurs (clients) . . . . .	2
<b>2</b>	<b>Fonctionnalités principales</b>	<b>2</b>
<b>3</b>	<b>Schéma global du système</b>	<b>3</b>
<b>4</b>	<b>API Web sélectionnée</b>	<b>4</b>
<b>5</b>	<b>Cas d'utilisation</b>	<b>4</b>
<b>6</b>	<b>Conception de la Base de Données</b>	<b>5</b>
<b>7</b>	<b>Description du client</b>	<b>6</b>
7.1	Page d'Accueil . . . . .	6
7.2	Page de Connexion . . . . .	7
7.3	Page d'Inscription . . . . .	8
7.4	Centre Utilisateur . . . . .	8
7.5	Soumission d'Évaluation . . . . .	10
<b>8</b>	<b>Description du Serveur</b>	<b>10</b>
8.1	UserServlet . . . . .	11
8.2	InscriptionServlet . . . . .	11
8.3	ProfileServlet . . . . .	11
8.4	RepairRequestServlet . . . . .	11
8.5	ReviewServlet . . . . .	11
<b>9</b>	<b>Interactions Client-Serveur</b>	<b>12</b>
9.1	Requêtes et Réponses HTTP . . . . .	12
9.1.1	Connexion de l'utilisateur . . . . .	12
9.1.2	Inscription de l'utilisateur . . . . .	12
9.1.3	Téléchargement d'avatar . . . . .	12
9.1.4	Publication et gestion des demandes de réparation . . . . .	12
9.1.5	Gestion des profils utilisateur . . . . .	13
<b>10</b>	<b>État d'Avancement du Projet</b>	<b>14</b>
10.1	Fonctionnalités Réalisées . . . . .	14
10.2	Fonctionnalités Non Réalisées . . . . .	15

# 1 Sujet du Projet

**FixNow** est une application web de service de réparation d'urgence, visant à connecter efficacement les demandeurs et les fournisseurs de services pour résoudre rapidement divers besoins de réparation d'urgence. Le site prend en charge deux types d'utilisateurs : les fournisseurs de services (réparateurs) et les demandeurs (clients).

Lien de GitLab : [https://stl.algo-prog.info/21305448/pc3r\\_projet\\_web.git](https://stl.algo-prog.info/21305448/pc3r_projet_web.git)

## 1.1 Fournisseurs de services (réparateurs)

Les réparateurs peuvent s'inscrire sur le site et créer un profil détaillé, comprenant un nom d'utilisateur, un mot de passe, une adresse e-mail, une photo de profil, ainsi qu'une série d'informations détaillées telles qu'une biographie, les types de réparations qu'ils maîtrisent, les années d'expérience et les certifications professionnelles obtenues. Ils peuvent également utiliser l'API Google Maps pour définir et mettre à jour leur emplacement géographique, ce qui aide non seulement les clients à trouver des fournisseurs de services à proximité, mais permet aussi aux réparateurs de filtrer les demandes de réparation en fonction de leur position. Les fournisseurs peuvent aussi gérer leurs disponibilités et consulter les avis et évaluations des clients à leur sujet.

## 1.2 Demandeurs (clients)

Les clients peuvent s'inscrire sur le site et gérer leur compte personnel, y compris la mise à jour des informations de contact et de localisation. En cas de besoin de services de réparation d'urgence, les clients peuvent publier une demande de réparation détaillant le type de machine, une description précise du problème et le délai souhaité pour l'intervention. Via la plateforme, les clients peuvent consulter les profils détaillés des fournisseurs de services et choisir un réparateur approprié en fonction de la localisation géographique, des disponibilités et des évaluations. Après la réparation, les clients peuvent évaluer le fournisseur de services, aidant ainsi les autres utilisateurs à faire des choix informés.

# 2 Fonctionnalités principales

1. **Utilisateur : Inscription et Connexion** : Authentifier les informations d'inscription et de connexion des utilisateurs
2. **Gestion du profil personnel** : Les fournisseurs de services peuvent télécharger et éditer leurs informations personnelles, y compris les qualifications professionnelles, la gamme de services, les horaires de travail et les coordonnées.
3. **Publication et gestion des demandes de réparation** : Les demandeurs peuvent décrire en détail leurs besoins de réparation (y compris le type de machine, la description du problème, le délai souhaité pour l'intervention, etc.) et les publier sur la plateforme pour que les fournisseurs de services les consultent.
4. **Fonctionnalité de géolocalisation** :
  - Les fournisseurs de services et les demandeurs peuvent utiliser l'API Google Maps pour définir et mettre à jour leur position, optimisant ainsi l'efficacité de la correspondance entre les deux parties.

- Intégration de la carte pour afficher la position en temps réel des fournisseurs de services, aidant les demandeurs à trouver les services les plus proches.
5. **Fonctionnalité de recherche et de filtrage** : Les utilisateurs peuvent rechercher et filtrer les fournisseurs de services selon divers critères, tels que le type de réparation, la localisation géographique, les évaluations des fournisseurs de services, afin de trouver rapidement le fournisseur de services approprié.
  6. **Système d'évaluation** :
    - Les demandeurs peuvent évaluer et noter le travail des fournisseurs de services après la réparation, fournissant ainsi des commentaires utiles pour aider d'autres utilisateurs à choisir leurs prestataires.
    - Les fournisseurs de services peuvent consulter et répondre aux évaluations, gérant ainsi leur réputation.

### 3 Schéma global du système

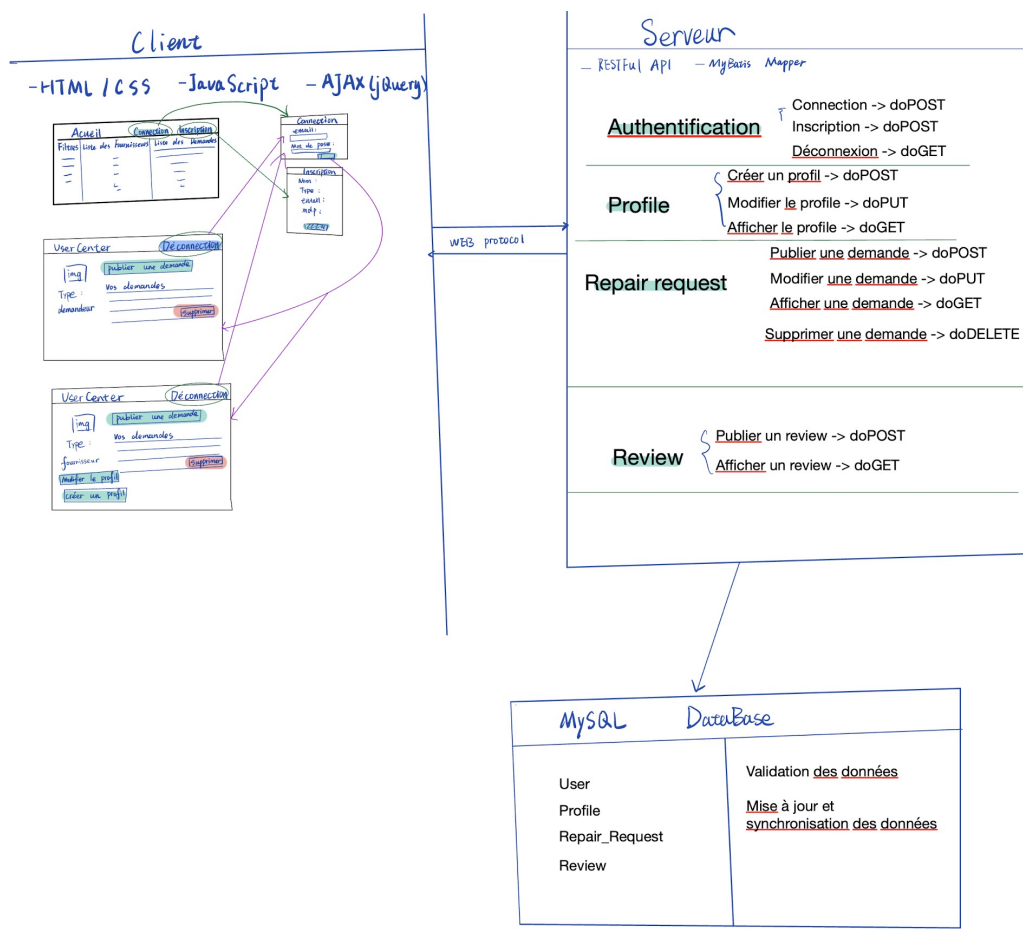


FIGURE 1 – Schéma global du système

4 API Web sélectionnée

Google Maps API :

- Permet aux fournisseurs de services de marquer et mettre à jour leur emplacement.
- Facilite la recherche de prestataires proches pour les demandeurs.
- Aide les demandeurs à évaluer la distance et la facilité d'accès.

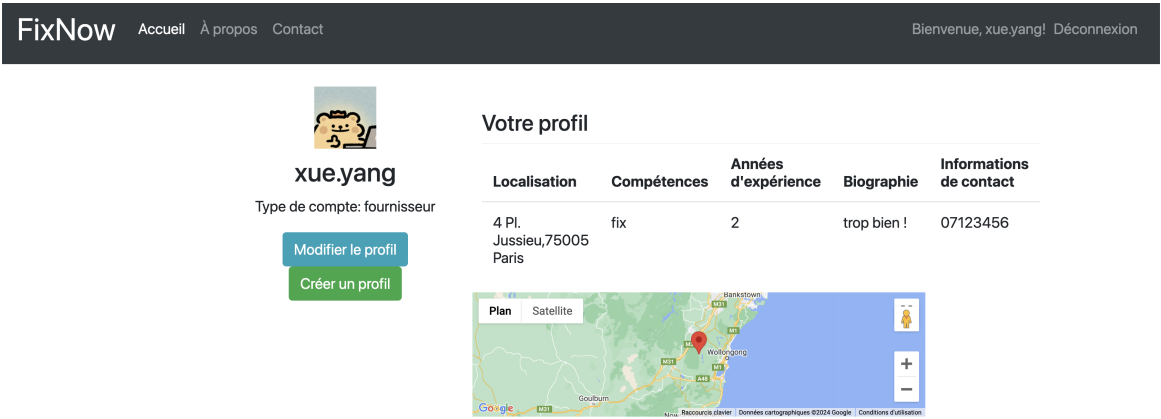


FIGURE 2 – userCenter-Modifier le profil

5 Cas d'utilisation

- (a) Alice souhaite publier une demande de réparation urgente. Elle visite la page d'accueil du site, clique sur le bouton "Connexion" et s'authentifie en entrant son adresse e-mail et son mot de passe. Après s'être connectée, Alice est automatiquement redirigée vers son espace personnel. Elle clique sur le bouton "Publier une demande" et remplit les informations spécifiques à sa demande : Type de machine, Description, Heure préférée.
- (b) Bob, un expert en réparation, se connecte à son compte et est redirigé vers son espace personnel. Dans son espace personnel, il clique sur le bouton "Créer un profil" et saisit sa Localisation, Compétences, Années d'expérience, Biographie et Informations de contact.
- (c) Alice utilise la barre de recherche sur la page d'accueil pour entrer des mots-clés liés au type de réparation. Elle utilise ensuite les filtres pour trouver des fournisseurs de services en fonction de la localisation géographique et des évaluations des utilisateurs. Après avoir consulté les profils détaillés des fournisseurs de services, elle décide de contacter un réparateur bien noté et proche de chez elle.
- (d) Bob consulte les nouvelles demandes de réparation publiées sur la page d'accueil et en trouve une qui correspond à ses disponibilités et compétences. Il accepte la demande en cliquant sur le bouton "Accepter la demande" et utilise la fonctionnalité de messagerie interne pour contacter Carole, la demandeuse, afin de discuter des détails de la réparation.
- (e) Une fois la réparation terminée, Carole accède à la section "Mes demandes de réparation" de son espace personnel, trouve la demande correspondante et clique sur le

bouton "Évaluer" pour donner son avis détaillé et une note sur le travail de Bob. Ces évaluations aident les autres utilisateurs à choisir leurs fournisseurs de services en se basant sur des expériences précédentes.

- (f) Bob se connecte régulièrement au site pour vérifier les nouvelles évaluations dans la section "Mes évaluations" de son espace personnel. Il met à jour ses disponibilités dans son espace personnel afin que les demandeurs sachent quand ils peuvent réserver ses services.

## 6 Conception de la Base de Données

La conception de la base de données pour notre application web FixNow comprend plusieurs tables essentielles qui permettent de gérer les utilisateurs, leurs emplacements, les profils des fournisseurs de services, les demandes de réparation et les évaluations des services. Voici une description détaillée de chaque table et de ses colonnes principales :

USER					
userId	email	password	userName	userType	avatar
1	xue.yang@etu.sorbonne-universite.fr	123	xue.yang	demandeur	/FixNow_war_explored/img/img.png
2	ariane.zhang@etu.sorbonne-universite.fr	1234	ariane.zhang	fournisseur	/FixNow_war_explored/img/four1.jpg

(a) Structure de la table user

Repair_Request					
requestId	userId	machineType	description	preferredTime	status
1	1	M1	en panne	demandeur	2024-05-26 20:00:00

(b) Structure de la table Repair\_Request

Profile						
profileId	userId	location	skills	experienceYears	bio	contactInfo
1	2	4 Pl. Jussieu, 75005 Paris	fix	5	trop bien!	07123456

(a) Structure de la table Profile

Review					
reviewId	requestId	reviewerId	revieweeId	rating	comment
1	1	1	2	5	Ponctuel et efficace

(b) Structure de la table Review

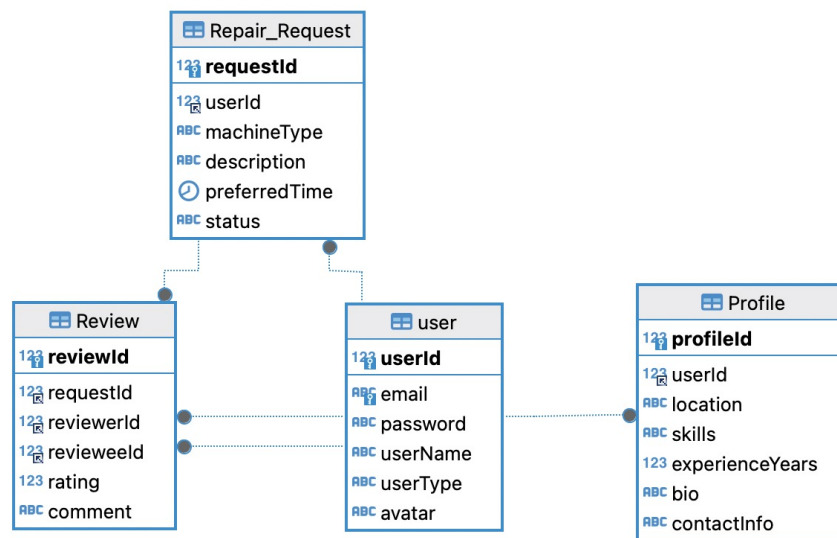


FIGURE 5 – Structure de la database FixNow

Les relations entre ces tables permettent de gérer efficacement les interactions entre les utilisateurs, les services fournis, et les évaluations reçues, assurant ainsi une expérience utilisateur fluide et structurée.

## 7 Description du client

Le client comprend les pages principales suivantes, chacune ayant des fonctionnalités spécifiques et utilisant différentes méthodes pour les réaliser :

### 7.1 Page d'Accueil

La page d'accueil affiche la liste des demandes de réparation et des fournisseurs de services (réparateurs) du site, offrant des fonctions de recherche et de filtrage.

- **Fonctionnalités** : Les utilisateurs peuvent consulter et rechercher des demandes de réparation ainsi que des informations détaillées sur les fournisseurs de services.
- **Méthode de réalisation** : Utilisation de requêtes Ajax pour obtenir les données du serveur et mettre à jour dynamiquement le contenu de la page.
- **Interaction avec le serveur** :
  - **GET /repairRequest** : Récupérer toutes les demandes de réparation publiques et les afficher sur la page d'accueil.
  - **GET /user?type=fournisseur** : Récupérer la liste de tous les fournisseurs de services et afficher leurs informations de base.

```
function fetchRepairRequests() {
  $.ajax({
    url: 'repairRequest',
    type: 'GET',
    success: function(response) {
```

(a) ajax-GET-Liste des Demandes

```
function fetchFournisseurList() {
  $.ajax({
    url: 'profile',
    type: 'GET',
    success: function(response) {
```

(b) ajax-GET-Liste des Fournisseurs

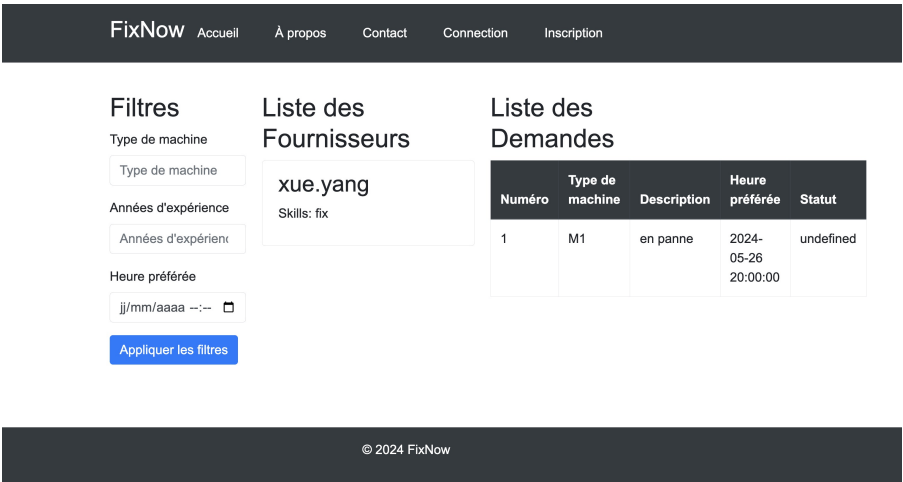


FIGURE 7 – Page d’Accueil

7.2 Page de Connexion

- La page de connexion permet aux utilisateurs de saisir leur nom d'utilisateur et leur mot de passe pour se connecter.
- **Fonctionnalités** : Les utilisateurs peuvent se connecter en entrant leur E-mail et leur mot de passe.
  - **Méthode de réalisation** : Après soumission du formulaire, une requête POST est envoyée au serveur pour vérifier les informations de l'utilisateur.
  - **Interaction avec le serveur** :
    - **POST /connexion** : Soumettre les informations de connexion pour l'authentification de l'utilisateur.

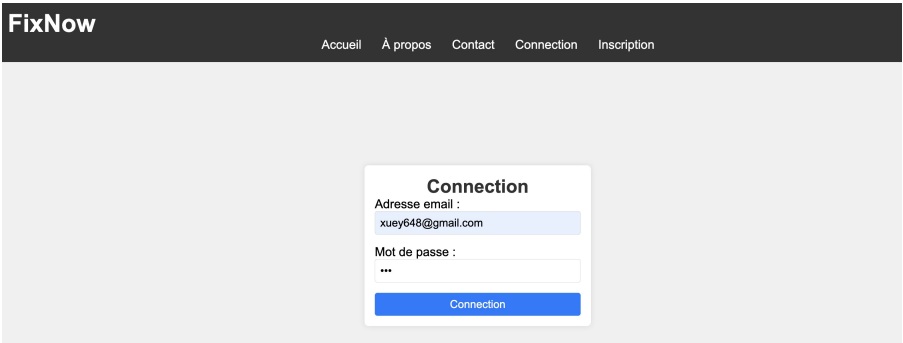


FIGURE 8 – Page de Connexion



### 7.3 Page d'Inscription

Cette page comprend un formulaire d'inscription demandant aux nouveaux utilisateurs de saisir un nom d'utilisateur, un email, un mot de passe, etc.

- **Fonctionnalités** : Les utilisateurs peuvent créer un nouveau compte en remplissant le formulaire.
- **Méthode de réalisation** : Après soumission du formulaire, une requête POST est envoyée au serveur pour enregistrer les informations de l'utilisateur.
- **Interaction avec le serveur** :
  - **POST /inscription** : Envoyer les informations d'inscription de l'utilisateur au serveur.

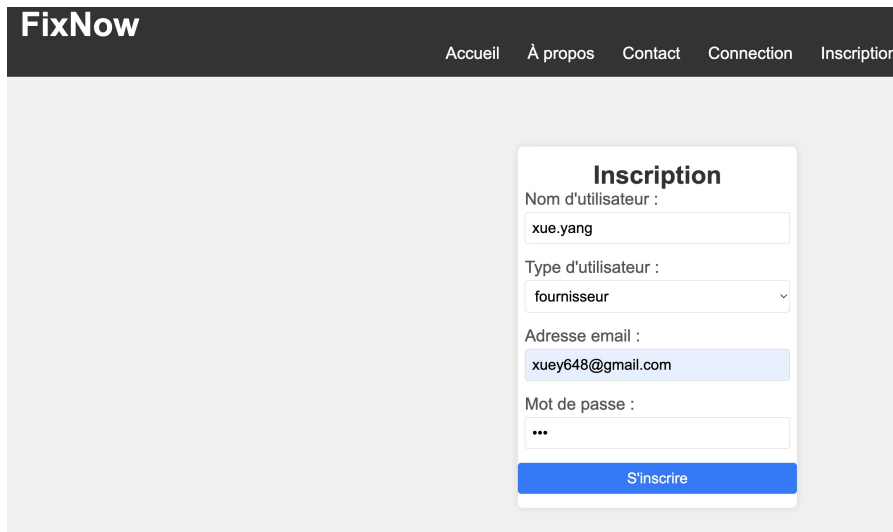


FIGURE 9 – Page d'Inscription

### 7.4 Centre Utilisateur

L'interface principale affichée après la connexion de l'utilisateur, montrant des informations différentes selon le type de compte (demandeur, fournisseur), incluant les informations personnelles, la liste des demandes de réparation, le formulaire de nouvelle demande, et la localisation.

- **Fonctionnalités** : Les utilisateurs peuvent consulter et gérer leurs informations personnelles, publier et gérer des demandes de réparation.
- **Méthode de réalisation** : Utilisation de requêtes Ajax pour mettre à jour dynamiquement les données de la page.
- **Interaction avec le serveur** :
  - **GET /repairRequest ?userId={userId}** : Récupérer la liste des demandes de réparation de l'utilisateur actuel.
  - **POST /repairRequest** : Permettre aux utilisateurs de publier de nouvelles demandes de réparation.
  - **DELET /repairRequest** : Permettre aux utilisateurs de supprimer de demandes de réparation.
  - **POST /profile** : Permettre aux utilisateurs de créer un profil.

- **PUT** /profile : Mettre à jour le profil.
- **GET** /profile?userId={userId} : Récupérer les informations personnelles de l'utilisateur.

```
$.ajax({
  url: 'repairRequest',
  type: 'GET',
  data: { userId: userId },
```

(a) ajax-GET-repairRequest

```
$.ajax({
  url: 'repairRequest',
  type: 'POST',
  contentType: 'application/json',
```

(b) ajax-POST-repairRequest

```
$.ajax({
  url: 'repairRequest',
  type: 'DELETE',
  contentType: 'application/json',
```

(c) ajax-DELETE-repairRequest

```
$.ajax({
  url: 'profile?userId=' + $('#userId').val(),
  type: 'GET',
  contentType: 'application/json',
```

(a) ajax-GET-profile

```
$.ajax({
  url: 'profile',
  type: 'POST',
  contentType: 'application/json',
```

(b) ajax-POST-profile

```
$.ajax({
  url: 'profile',
  type: 'PUT',
  contentType: 'application/json',
  data: JSON.stringify(updatedData),
```

(c) ajax-PUT-profile

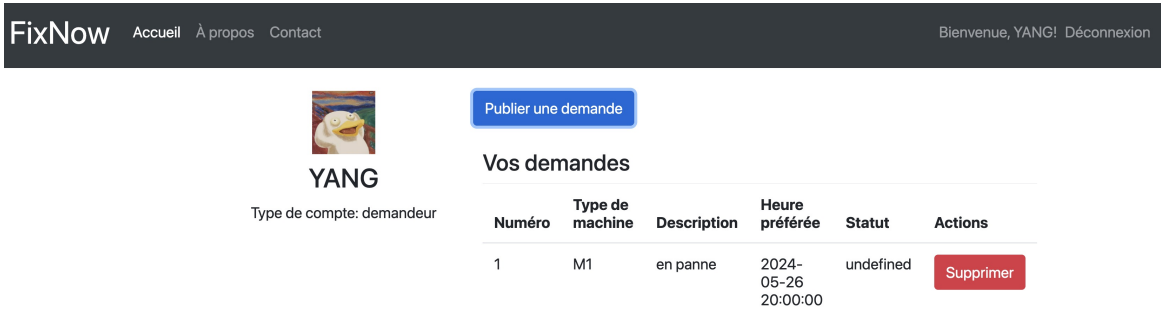


FIGURE 12 – userCenter-Demandeur

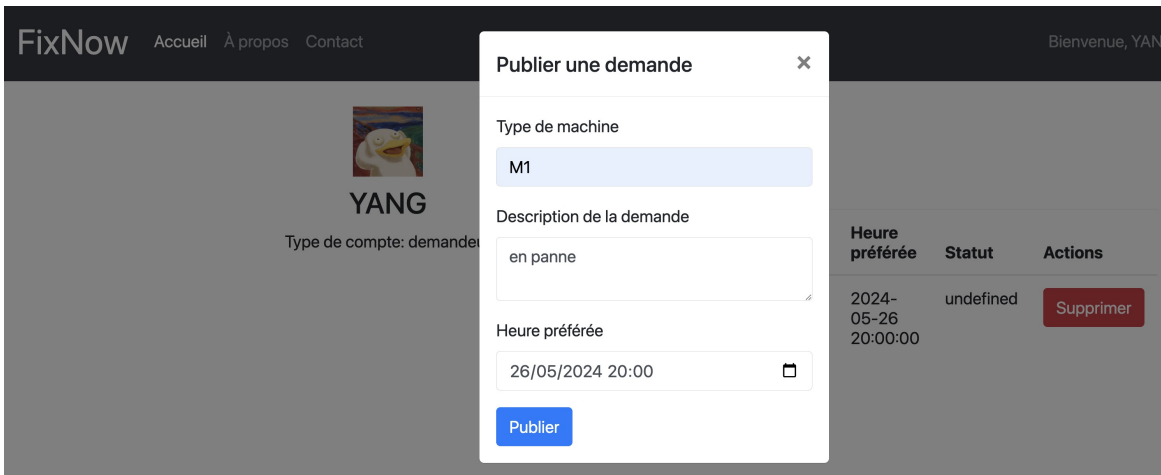


FIGURE 13 – userCenter-publier une demande

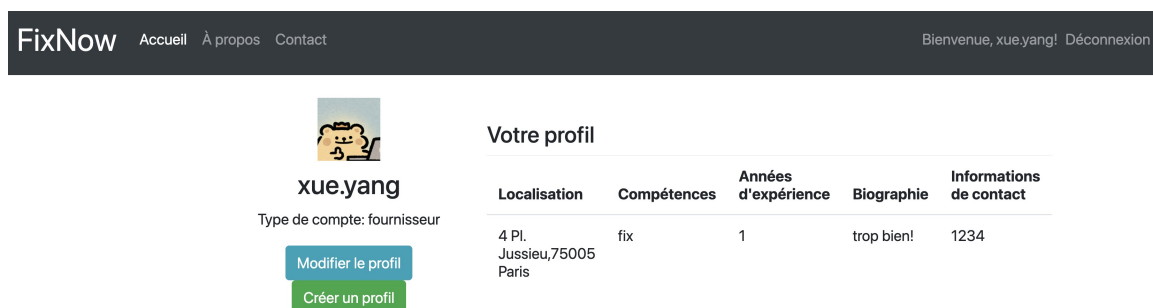


FIGURE 14 – userCenter-Fournisseur

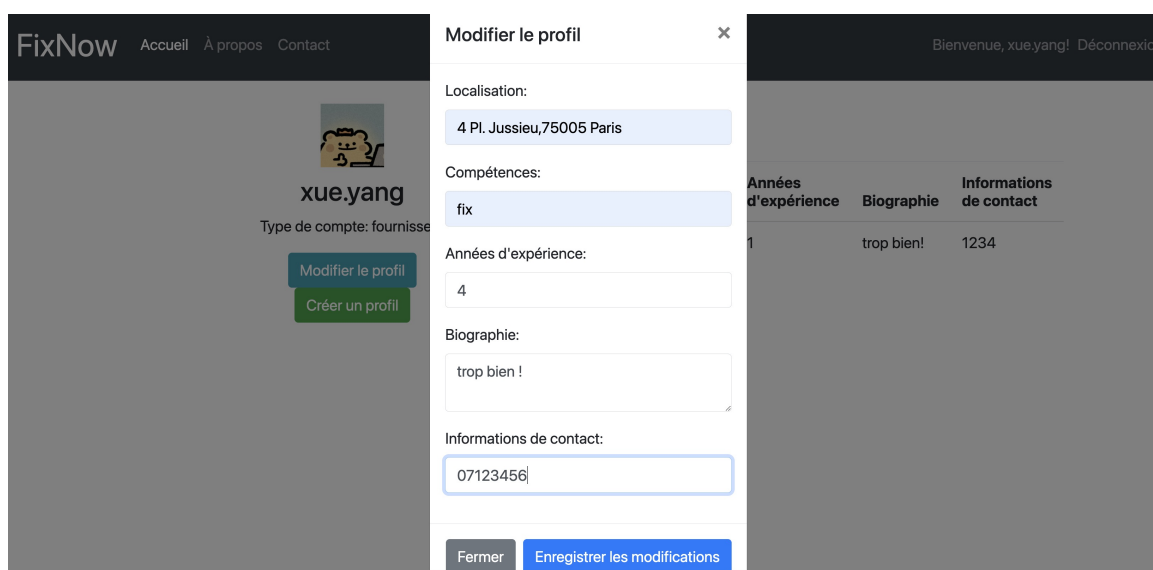


FIGURE 15 – userCenter-Modifier le profil

## 7.5 Soumission d'Évaluation

Permet de soumettre une évaluation et une note pour un service de réparation complété.

- **Fonctionnalités** : Les utilisateurs peuvent évaluer et noter les fournisseurs de services.
- **Méthode de réalisation** : Après soumission du formulaire, une requête POST est envoyée au serveur pour enregistrer l'évaluation.
- **Interaction avec le serveur** :
  - **POST /review** : Soumettre les informations d'évaluation au serveur.

## 8 Description du Serveur

Le serveur utilise la technologie Servlet et le serveur Tomcat pour traiter les requêtes des clients et interagir avec la base de données. Les principales fonctionnalités du serveur comprennent :

## 8.1 UserServlet

**Ressource :** /connection

**Fonctionnalités :**

- **doPost /connection** : Gérer la connexion de l'utilisateur. Si la connexion réussit, l'utilisateur est redirigé vers la page `userCenter.jsp`, sinon, il est renvoyé à la page de connexion avec un message d'erreur.

## 8.2 InscriptionServlet

**Ressource :** /inscription

**Fonctionnalités :**

- **doPost /inscription** : Gérer l'inscription d'un nouvel utilisateur. Si l'inscription réussit, l'utilisateur est redirigé vers la page de connexion, sinon, il est renvoyé à la page d'inscription avec un message d'erreur.

## 8.3 ProfileServlet

**Ressource :** /profile

**Fonctionnalités :**

- **doGet /profile/{userId}** : Récupérer le profil d'un utilisateur spécifié.
- **doPost /profile** : Créer un profil pour un nouvel utilisateur.
- **doPut /profile/{userId}** : Mettre à jour le profil d'un utilisateur spécifié.

## 8.4 RepairRequestServlet

**Ressource :** /repairRequest

**Fonctionnalités :**

- **doGet /repairRequest/{id}** : Récupérer les informations détaillées d'une demande de réparation spécifiée.
- **doGet /repairRequest** : Récupérer la liste de toutes les demandes de réparation, avec support de filtrage et de tri.
- **doPost /repairRequest** : Créer une nouvelle demande de réparation.
- **doPut /repairRequest/{id}** : Mettre à jour les informations d'une demande de réparation.
- **doDelete /repairRequest/{id}** : Annuler ou supprimer une demande de réparation.

## 8.5 ReviewServlet

**Ressource :** /review

**Fonctionnalités :**

- **doGet /review/{id}** : Consulter une évaluation spécifique.
- **doPost /review** : Soumettre une nouvelle évaluation de service.
- **doDelete /review/{id}** : Supprimer une évaluation.

## 9 Interactions Client-Serveur

### 9.1 Requêtes et Réponses HTTP

Voici quelques exemples de requêtes et de réponses HTTP utilisées pour l'interaction entre le client et le serveur :

#### 9.1.1 Connexion de l'utilisateur

- **Type de requête :** POST
- **URL de la requête :** /connection
- **Paramètres de la requête :**  
`email=exampleUser@example.com&password=examplePassword`
- **Réponse :**
  - Succès : Redirection vers `userCenter.jsp`
  - Échec : Redirection vers `connection.jsp` avec un message d'erreur

#### 9.1.2 Inscription de l'utilisateur

- **Type de requête :** POST
- **URL de la requête :** /inscription
- **Paramètres de la requête :**  
`username=newUser&usertype=demandeur&email=newuser@example.com  
&password=userPassword`
- **Réponse :**
  - Succès : Redirection vers `connection.jsp`
  - Échec : Redirection vers `inscription.jsp` avec un message d'erreur

#### 9.1.3 Téléchargement d'avatar

- **Type de requête :** POST
- **URL de la requête :** /userCenter
- **Paramètres de la requête :** FormData avec le fichier d'avatar
- **Réponse :**
  - Succès : Redirection vers `userCenter.jsp` avec l'URL de l'avatar mis à jour
  - Échec : Message d'erreur affiché sur `userCenter.jsp`

#### 9.1.4 Publication et gestion des demandes de réparation

- **Type de requête :** POST, GET, PUT, DELETE
- **URL de la requête :** /repairRequest
- **Paramètres de la requête :**
  - POST :

```
{  
  "userId": 123,
```

```
        "machineType": "Réfrigérateur",
        "description": "L'appareil fait un bruit inhabituel.",
        "preferredTime": "2024-05-20T14:00:00Z"
    }

— GET :
    {
        "userId": 123
    }

— PUT :
    {
        "requestId": 5678,
        "userId": 123,
        "machineType": "Réfrigérateur",
        "description": "L'appareil fait un bruit inhabituel.",
        "preferredTime": "2024-05-20T14:00:00Z",
        "status": "open"
    }

— DELETE :
    {
        "requestId": 5678
    }

— Réponse :
    {
        "success": true,
        "message": "Opération réussie"
    }

ou
    {
        "success": false,
        "message": "Erreur lors de l'opération"
    }
```

### 9.1.5 Gestion des profils utilisateur

- **Type de requête** : GET, POST, PUT, DELETE
- **URL de la requête** : /profile
- **Paramètres de la requête** :
  - GET :

```
    {
        "userId": 123
```

```
    }

— POST, PUT :
    {
        "userId": 123,
        "location": "4 Pl. Jussieu,75005 Paris",
        "skills": "Technicien certifié HVAC",
        "experienceYears": 5,
        "bio": "Expérimenté dans les grands projets",
        "contactInfo": "555-0199"
    }

— DELETE :
    {
        "profileId": 5678
    }

— Réponse :
    {
        "success": true,
        "message": "Opération réussie"
    }

ou
    {
        "success": false,
        "message": "Erreur lors de l'opération"
    }
```

Chaque requête est conçue pour retourner un champ **success**, indiquant le succès ou l'échec de l'opération, ainsi qu'un champ **message** fournissant des informations sur l'opération. Dans le cadre de la gestion de la connexion et de l'inscription des utilisateurs, nous avons opté pour une méthode intuitive de redirection multi-pages au lieu de l'utilisation d'Ajax. Cependant, pour améliorer l'expérience utilisateur, nous utilisons Ajax pour récupérer les informations des utilisateurs et des demandes de réparation depuis la base de données sur la page du centre utilisateur et la page d'accueil. Cette combinaison permet d'assurer une navigation fluide et réactive.

## 10 État d'Avancement du Projet

### 10.1 Fonctionnalités Réalisées

Dans le cadre de ce projet, nous avons réussi à implémenter plusieurs fonctionnalités clés pour notre système de gestion des réparations :

### Authentification

- **Connexion (doPOST)** : Les utilisateurs peuvent se connecter à leur compte.
- **Inscription (doPOST)** : Les utilisateurs peuvent créer un nouveau compte.
- **Déconnexion (doGET)** : Les utilisateurs peuvent se déconnecter de leur compte.

### Profile

- **Créer un profil (doPOST)** : Les utilisateurs peuvent créer leur profil.
- **Modifier le profil (doPUT)** : Les utilisateurs peuvent mettre à jour leur profil.
- **Afficher le profil (doGET)** :
  - **Affichage par ID utilisateur dans le centre utilisateur** :
    - **Fournisseurs** : Les fournisseurs peuvent consulter et gérer leur propre profil, y compris les informations telles que leur biographie, les types de réparations qu'ils maîtrisent, leurs années d'expérience et leurs certifications.
  - **Affichage sur la page d'accueil** :
    - **Tous les utilisateurs** : Sur la page d'accueil, tous les utilisateurs, qu'ils soient fournisseurs ou demandeurs, peuvent voir la liste des fournisseurs de services ainsi que la liste des demandes de réparation.

### Repair Request

- **Publier une demande (doPOST)** : Les utilisateurs peuvent soumettre des demandes de réparation.
- **Modifier une demande (doPUT)** : Les utilisateurs peuvent mettre à jour leurs demandes de réparation.
- **Afficher une demande (doGET)** :
  - **Affichage par ID utilisateur dans le centre utilisateur** :
    - **Demandeurs** : Les demandeurs peuvent consulter les détails de leurs propres demandes de réparation, y compris les réponses des fournisseurs.
  - **Affichage sur la page d'accueil** : Tous les utilisateurs peuvent voir la liste complète des demandes de réparation publiées par les demandeurs.
- **Supprimer une demande (doDELETE)** : Les utilisateurs peuvent supprimer leurs demandes de réparation.

## 10.2 Fonctionnalités Non Réalisées

En raison de contraintes de temps, certaines fonctionnalités prévues initialement n'ont pas été implémentées :

**Évaluation et Notation** : La fonctionnalité permettant aux demandeurs d'évaluer et de noter les fournisseurs de services après une réparation n'a pas encore été mise en place.

**Localisation Avancée** : Bien que nous ayons intégré l'API Google Maps pour afficher la localisation des utilisateurs, nous n'avons pas encore implémenté les fonctionnalités logiques avancées. De plus, l'utilisation de cette API implique des coûts, ce qui a limité notre capacité à explorer toutes ses fonctionnalités.