

# RAPPORT D'ACTIVITÉS

Réalisation d'une application de candidature  
basée sur le Framework Xataface

SIO1 2021-2022



# TABLE DES MATIÈRES

<b>LEXIQUE</b>	<b>1</b>
<b>ABBREVIATIONS</b>	<b>1</b>
<b>REMERCIEMENTS</b>	<b>2</b>
<b>INTRODUCTION</b>	<b>2</b>
<b>PREMIÈRE PARTIE: PRÉSENTATION DU CONTEXTE</b>	<b>3</b>
I. Présentation de LDNR	3
II. Missions principales	4
III. Xataface	4
<b>DEUXIÈME PARTIE: APPLICATION DE PRÊT POUR LA MONTÉE EN COMPÉTENCE SUR XATAFACE</b>	<b>5</b>
I. Présentation de la mission Prêt	5
II. Conception de la base de données Prêt	5
A. Modèle conceptuel de données	5
B. Modèle logique de données	6
C. Création de la BdD et installation de Xataface	7
D. Configuration de Xataface avec le fichier conf.ini	7
1. Section [_database]	8
2. Section [_tables]	9
3. Section [_auth]	9
E. Relations entre les tables	9
F. Modification de l’affichage des tables et des formulaires grâce aux fichiers fields.ini et valuelists.ini	10
1. Création et gestion de l’affichage de la vue ‘Prêt en cours’	10
2. Modification de l’affichage de la vue ‘Prêt en cours’ grâce à fields.ini	11
G. Gestion des droits: rôles et permissions	13
<b>TROISIÈME PARTIE: APPLICATION CANDIDATURE</b>	<b>15</b>
I. CONTEXTE	15
II. PRÉOPÉRATOIRE: MCD, MLD, UML, GANTT/PERT, MODÉLISATION BDD	15
A. MCD/MLD	15
B. Maquettes	15
C. Diagramme de cas d’utilisation	16
D. Diagramme prédictif de GANTT	16
III. RÉALISATION DE L’APPLICATION CANDIDAT	17
A. Création de la page d’accueil index.html	17
B. Le formulaire d’inscription	17
1. Signup.html	17
2. Inscription.php	17
3. Mdpcheck.js	18

---

C. Permissions	18
IV. Conclusion	19
<b>QUATRIÈME PARTIE: Annexes</b>	<b>20</b>
Annexe 1: script de création de la Bdd prêt	20
Annexe 2: protocole d'installation de Xataface	21
Suite Annexe 2: protocole d'installation de Xataface	22
Annexe 3: fichier relationships.ini de la table lend	23
Annexe 4: requête pour la génération de la vue 'prêt en cours'	24
Annexe 5: MCD/MLD de l'application candidature	25
Annexe 6: Maquettes accueil et inscription	26
Annexe 7: db_functions.php de l'application candidature	27
Annexe 8: functions.inc.php de l'application candidature	28
Annexe 9: Application Delegate de l'application candidature	29
Annexe 10: index.html de l'application candidature	30
Annexe 11: Signup.html de l'application candidature	31
Annexe 12: Inscription.php de l'application candidature	32

## TABLE DES FIGURES

Figure 1: Chiffres clés de LDNR pour l'année 2021-2022. Source: <a href="http://ldnr.fr">ldnr.fr</a>	3
Figure 2: MCD de l'application de prêt.	5
Figure 3: MLD de l'application de prêt.	7
Figure 4: Fichier conf.ini de l'application de prêt.	8
Figure 5: Menu de navigation de l'application de prêt.	9
Figure 6: Fichier relationships.ini de la table formatio	9
Figure 7: Relation entre les tables formation et session.	10
Figure 8: Fichier fields.ini de la vue 'Prêt en cours'.	11
Figure 9: Fichier valuelists.ini de la vue 'Prêt en cours'.	12
Figure 10: Table 'users' de l'application de prêt.	13
Figure 11: Fonctions du function.inc.php.	13
Figure 12: Classe ApplicationDelegate.	14
Figure 13: Diagramme de cas d'utilisation de l'application candidature.	16
Figure 14: Diagramme prédictif de GANTT de l'application candidature.	16
Figure 15: mdpcheck.js de l'application candidature.	18
Figure 16: Extrait du tableau de synthèse des compétences du B1 du BTS SIO	19

## TABLE DES ILLUSTRATIONS

Tableau 1: Liste non exhaustive des formations et certifications proposées par LDNR.	3
Tableau 2: Entités du MCD de l'application de prêt.	5
Tableau 3: Associations du MCD de l'application de prêt.	6

## LEXIQUE

**Framework:** un framework est un ensemble cohérent d'outils et de composants logiciels structurels qui sert à créer les fondations ainsi que les grandes lignes de tout ou partie d'un logiciel, c'est-à-dire une architecture.

**Front-end:** aussi appelé développement web frontal, correspond aux productions HTML, CSS et JavaScript d'une page internet ou d'une application qu'un utilisateur peut voir et avec lesquelles il peut interagir directement.

**Back-end:** en informatique, un back-end est un terme désignant un étage de sortie d'un logiciel devant produire un résultat. On l'oppose au front-end qui est similaire à la partie visible de l'iceberg.

**Open Source:** un logiciel Open Source est un logiciel informatique qui est publié sous une licence dans laquelle le titulaire du droit accorde aux utilisateurs le droit d'utiliser, d'étudier, de modifier et de distribuer le logiciel et son code source à quiconque et à n'importe quelle fin.

**Base de données:** une base de données permet de stocker et de retrouver des données structurées, semi-structurées ou des données brutes ou de l'information, souvent en rapport avec un thème ou une activité; celles-ci peuvent être de natures différentes et plus ou moins reliées entre elles.

## ABBREVIATIONS

**LDNR:** La Distance Nous Rapproche.

**BdD:** base de données.

**MCD:** modèle conceptuel de données.

**MLD:** modèle logique de données.

---

## REMERCIEMENTS

Tout d'abord je tiens à remercier Nina Büchner et Emile Marco pour leur accueil, pour leur bienveillance et pour la confiance qu'ils m'ont accordée en me permettant de participer à ce projet.

Un grand merci aussi à Dylan Turnier qui a eu la gentillesse et la patience nécessaire pour m'aider à prendre mes marques et intégrer la famille LDNR.

Je tiens également à remercier mon collègue de stage Thomas Landes, qui a su m'épauler et me rassurer lorsque j'en avais besoin.

Enfin, un immense merci à Jean-François Ramiara grâce à qui j'ai pu obtenir un entretien avec Emile et Nina qui a débouché sur l'obtention du stage chez LDNR.

## INTRODUCTION

C'est dans le cadre de ma première année de BTS SIO, option SLAM, que j'ai eu le plaisir d'effectuer un stage de 7 semaines chez LDNR du 16 mai 2022 au 30 juin 2022. Nina Büchner et Emile Marco, les fondateurs, nous ont chargé, Thomas Landes et moi-même, de refaire l'application de candidature du centre, ainsi que l'application de gestion des candidatures.

# PREMIÈRE PARTIE: PRÉSENTATION DU CONTEXTE

## I. Présentation de LDNR

LDNR - La Distance Nous Rapproche - est un centre de formation pour adultes situé au cœur de la zone d'activité de Labège qui les prépare à divers métiers du numérique à travers les cours dispensés. Présent depuis 12 ans, LDNR a été cofondé par Nina BÜCHNER et Emile MARCO.

Parmi les différentes formations et certifications numériques proposées par LDNR, en salle ou à distance, on peut notamment retrouver:

Formations	Certifications
Bureautique	Développeur.se Web et Web Mobile
Programmation	JAVA J2E
Sécurité et I.A.	Technicien.ne Supérieur Système et Réseaux
Gestion de projet	
Communication et R.H.	

Tableau 1: Liste non exhaustive des formations et certifications proposées par LDNR.

La figure ci-dessous présente les chiffres clés réalisés par LDNR au cours de l'année 2021-2022:



Figure 1: Chiffres clés de LDNR pour l'année 2021-2022. Source: [ldnr.fr](https://ldnr.fr)

## II. Missions principales

LDNR étant un organisme de formation spécialisé dans le numérique, les candidats aux différentes formations doivent avoir la possibilité de postuler en ligne. Notre mission, à Thomas et à moi-même, est de proposer une nouvelle application de candidature basée sur le framework Xataface, qui soit plus ergonomique et intuitive pour les utilisateurs. Alors que Thomas a davantage travaillé sur l'application d'administration des candidatures, j'ai pour ma part fait la partie front-end à destination des candidats.

Dans un premier temps, afin de monter en compétences sur Xataface, nous avons réalisé une petite application de prêt de PC LDNR.

Dans un second temps, nous avons travaillé sur l'application permettant aux utilisateurs de candidater aux formations proposées par le centre.

## III. Xataface

Xataface est un framework web open source écrit en PHP, développé par Steve Hannah, dont la première version date de 2005. La version 2.0 date de 2014 et la 3.0, qui est la version que nous allons couvrir, est disponible depuis octobre 2021.

Xataface permet de générer automatiquement une interface graphique pour interagir avec la base de données (BdD) sans avoir à écrire des requêtes SQL. C'est par l'intermédiaire de formulaires, listes et menus qu'il est possible de d'effectuer les actions **CRUD** sur la BdD et les tables qui la composent:

- **Create** pour créer
- **Read** pour lire
- **Update** pour mettre à jour
- **Delete** pour supprimer.

Contrairement à d'autres frameworks qui nécessitent de longuement coder pour avoir une application fonctionnelle, Xataface peut être mis en place en littéralement 4 lignes de PHP. Cette particularité rend cette solution aussi bien adaptée aux développeurs web, administrateurs de base de données, qu'aux personnes aux profils moins techniques.

De plus Xataface n'étant pas très connu, les attaques malveillantes visant les applications basées sur ce framework sont peu courantes, ce qui en fait un avantage non négligeable au vu de la cyberguerre actuelle.

Xataface jouit d'une communauté active de contributeurs sur GitHub, avec Steve Hannah lui-même qui répond aux éventuelles questions que peuvent avoir les utilisateurs du framework.



## DEUXIÈME PARTIE: APPLICATION DE PRÊT POUR LA MONTÉE EN COMPÉTENCE SUR XATAFACE

### I. Présentation de la mission Prêt

Afin de monter en compétences sur Xataface, dans un premier temps nous avons réalisé une petite application de prêt de matériel à partir d'un modèle fourni. Le but était de permettre un affichage de la Bdd associée au prêt de matériel et de pouvoir effectuer les différentes actions **CRUD** vues plus haut.

### II. Conception de la base de données Prêt

#### A. Modèle conceptuel de données

A l'aide de Looping, nous avons réalisé le MCD présenté ci-dessous. Les clés primaires des différentes entités sont soulignées.

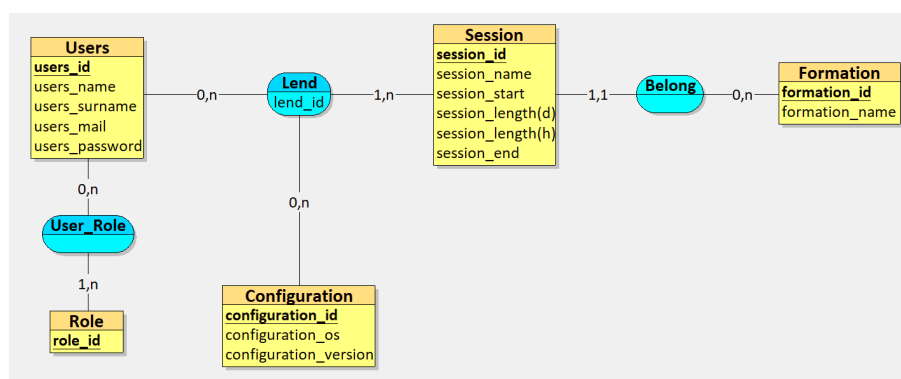


Figure 2: MCD de l'application de prêt.

Les 5 entités figurant sur le MCD sont les suivantes:

Nom de l'entité	Description
Users	Regroupe les utilisateurs
Session	Période sur laquelle se déroule la formation
Formation	Formation suivie par l'utilisateur
Configuration	Configuration du PC emprunté par l'utilisateur
Role	Rôle de l'utilisateur au sein de LDNR

Tableau 2: Entités du MCD de l'application de prêt.

Les associations pour ce modèle, au nombre de 3, sont présentées ci-dessous:

Nom de l'association	Description
User_Role	Certains utilisateurs peuvent avoir plusieurs rôles (0,n), et chaque rôle est associé à au moins un utilisateur (1,n)
Lend	Cette association ternaire possède une propriété qui est l'identifiant du prêt et permet de faire le lien entre les utilisateurs, la configuration du PC emprunté et de la session suivie. Certains utilisateurs ont plusieurs prêts en cours (0,n), une session peut être associée à plusieurs prêts (1,n) et certaines configurations peuvent être associées à plusieurs prêts (0,n)
Belong	une session correspond à une formation (1,1), certaines formations peuvent avoir plusieurs sessions (0,n)

Tableau 3: Associations du MCD de l'application de prêt.

## B. Modèle logique de données

Grâce à Looping, il est possible de passer du MCD au MLD. L'association 'Belong' a disparu, laissant place à une clé étrangère dans la table Session, et permet de faire le lien entre les tables Session et Formation.

Lend s'est transformé en une table à part entière, dont la clé primaire est la concaténation des clés étrangères des tables Session, Users et Configuration.

User\_Role est également devenu une table, dont la clé primaire est aussi la concaténation des clés étrangères des tables Users et Role. Le MLD ainsi généré est présenté ci-dessous:

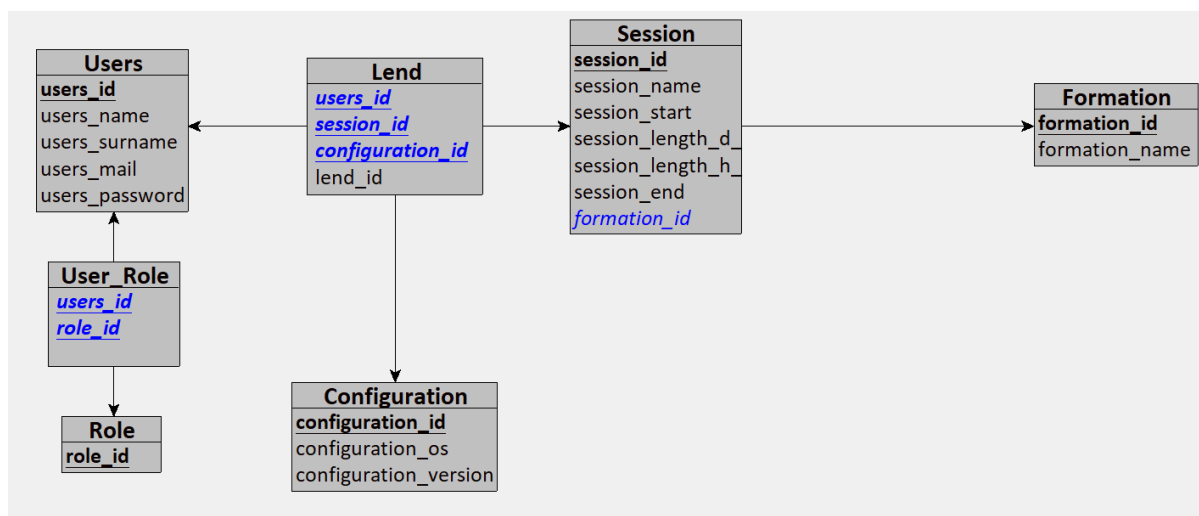


Figure 3: MLD de l'application de prêt.

### C. Création de la BdD et installation de Xataface

Le script de création de la BdD se trouve en annexe 1. Afin d'éviter les problèmes de contraintes liés aux clés étrangères au moment de la création des tables dans phpMyAdmin, nous les avons "désactivées" et avons ajouté dans leur intitulé "fk" pour *foreign key* en préfixe. En effet, il nous a été demandé de mettre en place les relations entre les tables via Xataface, et non phpMyAdmin.

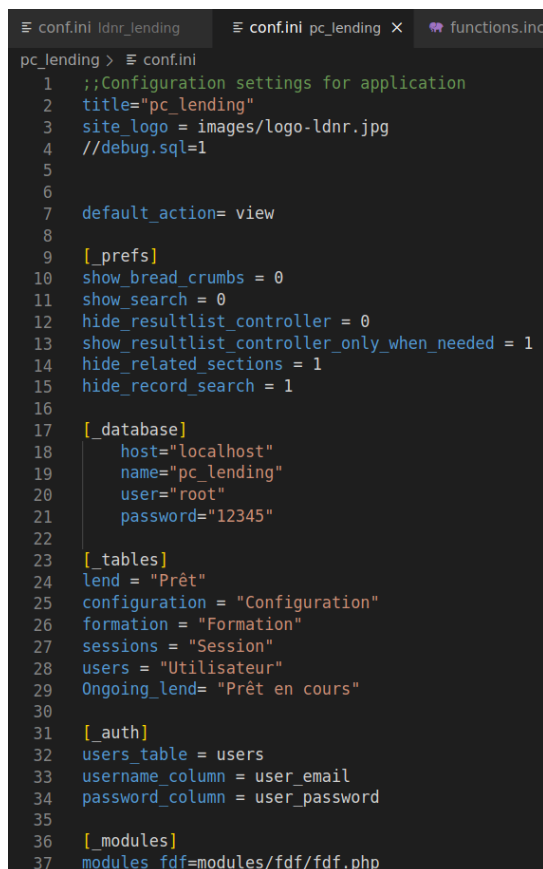
Le protocole à suivre pour lier la BdD ainsi créée et le framework Xataface est présenté en annexe 2.

### D. Configuration de Xataface avec le fichier *conf.ini*

Afin de configurer Xataface, il convient de modifier le fichier *conf.ini* présent dans le répertoire de l'application, situé dans notre cas dans */var/www/html/pc\_lending*. L'édition du *conf.ini* permet entre autres de:

- stocker les informations de connexion à la BdD
- choisir les tables qui vont apparaître dans le menu des tables
- instaurer des règles d'authentification
- utiliser des modules complémentaires
- etc

La capture d'écran ci-dessous est le conf.ini de l'application de prêt:



```

pc_lending > conf.ini
1  ;;Configuration settings for application
2  title="pc_lending"
3  site_logo = images/logo-ldnr.jpg
4  //debug.sql=1
5
6
7  default_action= view
8
9  [_prefs]
10 show_bread_crums = 0
11 show_search = 0
12 hide_resultlist_controller = 0
13 show_resultlist_controller_only_when_needed = 1
14 hide_related_sections = 1
15 hide_record_search = 1
16
17 [_database]
18 host="localhost"
19 name="pc_lending"
20 user="root"
21 password="12345"
22
23 [_tables]
24 lend = "Prêt"
25 configuration = "Configuration"
26 formation = "Formation"
27 sessions = "Session"
28 users = "Utilisateur"
29 Ongoing_lend= "Prêt en cours"
30
31 [_auth]
32 users_table = users
33 username_column = user_email
34 password_column = user_password
35
36 [_modules]
37 modules_fdf=modules/fdf/fdf.php
  
```

Figure 4: Fichier conf.ini de l'application de prêt.

## 1. Section [\_database]

Cette section permet de configurer la relation avec la base de données interrogée. Dans notre cas, le serveur apache est local, l'hôte est donc le localhost, joignable au 127.0.0.1.

Le nom de la BdD est celle que nous avons précédemment définie dans phpMyAdmin au moment de sa création, *pc\_lending*.

Enfin, nous interagissons avec la BdD en tant que root, donc avec toutes les permissions.

## 2. Section [\_tables]

Cette section permet de générer la liste des tables qui seront accessibles depuis le menu navigation de l'application et de leur faire correspondre un nom qui sera affiché dans ce même menu comme on peut le voir ci-dessous:

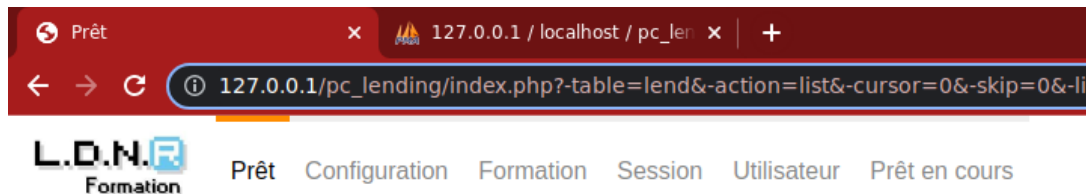


Figure 5: Menu de navigation de l'application de prêt.

## 3. Section [\_auth]

Cette section permet de renseigner le nom exact de la table dans laquelle sont stockées les informations nécessaires à l'authentification, ainsi que les colonnes correspondant au nom d'utilisateur et au mot de passe. Il est également possible de choisir si par exemple l'authentification par adresse mail est autorisée ou non.

## E. Relations entre les tables

Nous avons vu précédemment que les relations entre les tables n'ont pas été faites via phpMyAdmin car nous avons supprimé les clés étrangères. Pour le moment, nos tables ne sont donc pas reliées entre elles, et il n'est pas possible d'interroger plusieurs tables. Pour remédier à cela, il nous faut créer un fichier *relationships.ini* dans chaque répertoire de table.

Par exemple pour lier les tables formation et session, dans le *relationships.ini* de la table, nous précisons qu'il existe une égalité entre les valeurs de la colonne *formation\_id* de *formation* et *fk\_formation\_id* de la table *session* comme présenté ci-dessous:

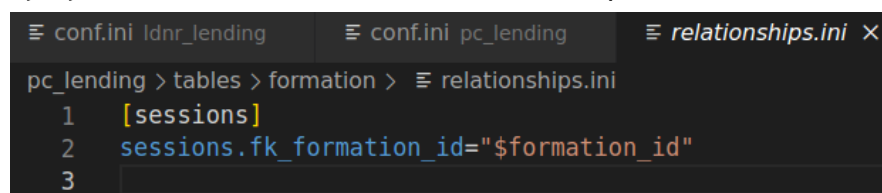


Figure 6: Fichier relationships.ini de la table formation.

L'établissement de cette relation peut être directement visualisée lors de la navigation sur l'application:

Algorithmique – initiation (base de la programmation) #1

Formation

▼ Details List Find

View Sessions

Sort Add New Sessions Record Filter

Found 1 Records in relationship sessions  
Now Showing 1 to 1 (Display 30 records per page)

Session id	Session name	Session start	Session end
4	ALGO_INI_2022-2023	01/09/22	31/05/23

With Selected: Remove Update

Found 1 Records in relationship sessions  
Now Showing 1 to 1 (Display 30 records per page)

Powered by Xataface  
(c) 2005-2022 All rights reserved

Figure 7: Relation entre les tables formation et session.

## F. Modification de l'affichage des tables et des formulaires grâce aux fichiers *fields.ini* et *valuelists.ini*

Les modifications effectuées dans ces fichiers affectent à la fois l'affichage de la table ET le formulaire de saisie de la table, généré par Xataface.

La table *lend*, que nous avons renommée *prêt* est une table à part car elle regroupe beaucoup d'informations venant des autres tables de la BdD de l'application de prêt. En conséquence, son fichier *relationships.ini* comporte les jointures qui la relie aux tables *session*, *configuration* et *users* et est présenté en annexe 3.

### 1. Création et gestion de l'affichage de la vue 'Prêt en cours'

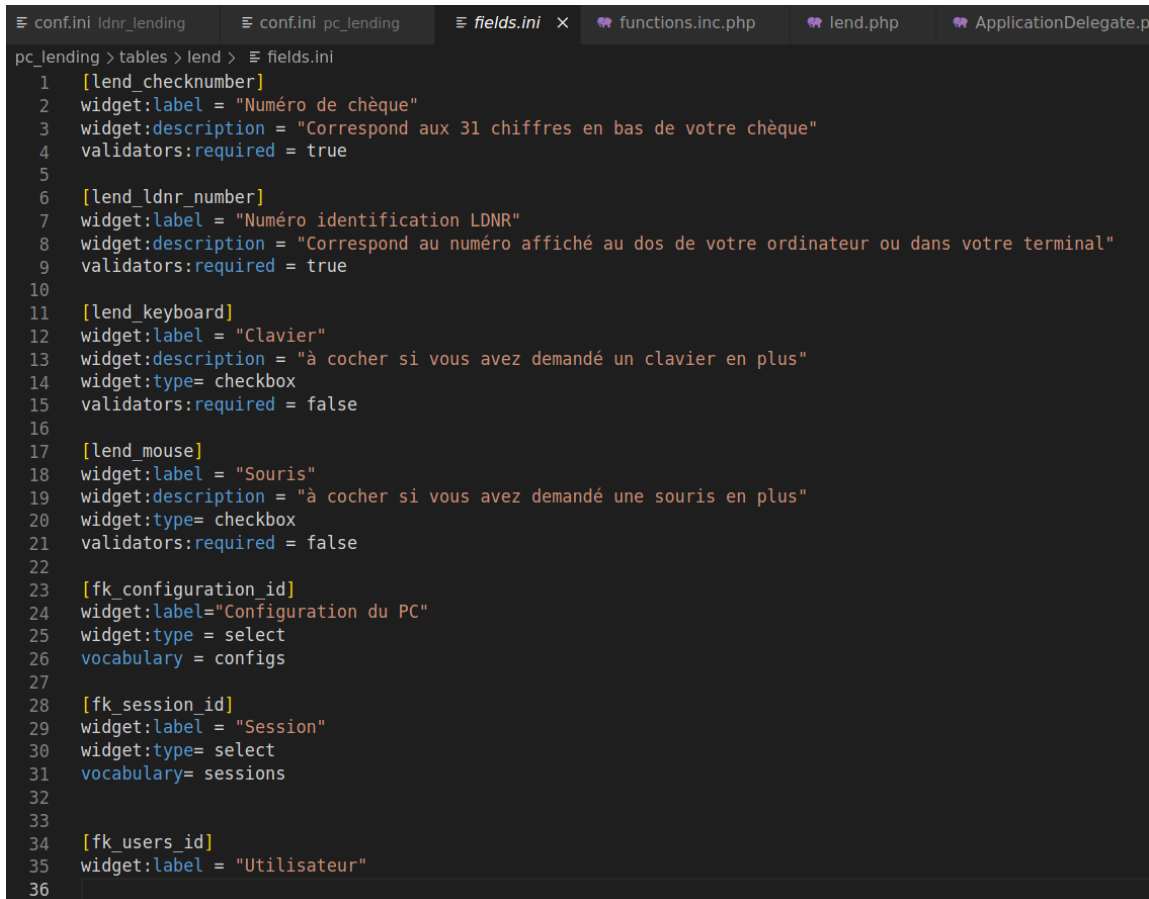
Afin de contourner les problèmes rencontrés lors de l'affichage de la table *lend*, nous avons créé une vue depuis phpMyAdmin, regroupant toutes les informations que nous souhaitons faire apparaître pour le suivi de prêt. La requête permettant de récupérer toutes les informations qui nous intéressent est présentée dans l'annexe 4.

De plus, pour que notre application puisse utiliser la vue générée dans phpMyAdmin, deux actions sont nécessaires:

- il faut **déclarer la vue** dans la section `[_database]` du fichier *conf.ini* de l'application (figure 4).
- dans le fichier *fields.ini* de la vue, il est obligatoire de **déclarer une clé primaire**, faute de quoi Xataface retourne une erreur. Dans la section du champ correspondant, on note `KEY=PRI`.

## 2. Modification de l’affichage de la vue ‘Prêt en cours’ grâce à *fields.ini*

Le fichier *fields.ini* permet d’éditer les métadonnées liées à une table en particulier et d’en modifier le nom des champs affichés par Xataface. La figure 8 ci-dessous est une capture d’écran des modifications faites au fichier *fields.ini* de la vue ‘prêt en cours’ :



```

pc_lending > tables > lend > fields.ini
1  [lend_checknumber]
2  widget:label = "Numéro de chèque"
3  widget:description = "Correspond aux 31 chiffres en bas de votre chèque"
4  validators:required = true
5
6  [lend_ldnr_number]
7  widget:label = "Numéro identification LDNR"
8  widget:description = "Correspond au numéro affiché au dos de votre ordinateur ou dans votre terminal"
9  validators:required = true
10
11 [lend_keyboard]
12 widget:label = "Clavier"
13 widget:description = "à cocher si vous avez demandé un clavier en plus"
14 widget:type= checkbox
15 validators:required = false
16
17 [lend_mouse]
18 widget:label = "Souris"
19 widget:description = "à cocher si vous avez demandé une souris en plus"
20 widget:type= checkbox
21 validators:required = false
22
23 [fk_configuration_id]
24 widget:label="Configuration du PC"
25 widget:type = select
26 vocabulary = configs
27
28 [fk_session_id]
29 widget:label = "Session"
30 widget:type= select
31 vocabulary= sessions
32
33
34 [fk_users_id]
35 widget:label = "Utilisateur"
36

```

Figure 8: Fichier *fields.ini* de la vue ‘Prêt en cours’.

Chaque section du *fields.ini* de ‘lend’ correspond en réalité aux colonnes de la table. Les widgets *label*, *description* et *type* permettent respectivement de :

- changer le nom affiché de la colonne
- ajouter une description au champ
- définir le type d’entrée attendue

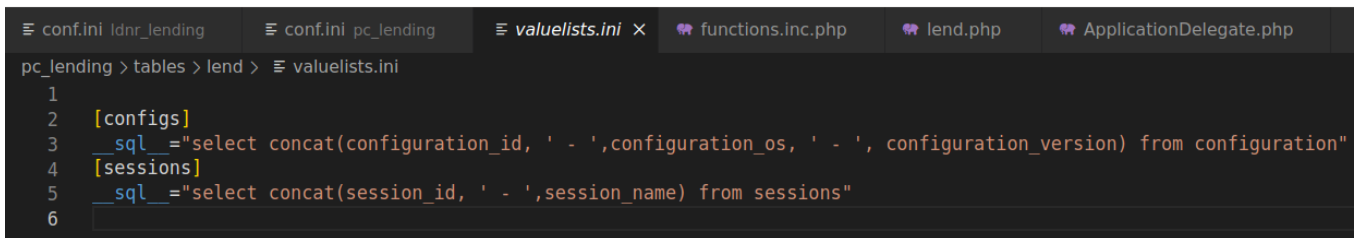
L’instruction ‘*vocabulary*’ fait référence aux différentes sections déclarées dans le fichier *relationships.ini* de la table en question et permet de récupérer et d’afficher les informations en provenance d’autres tables.

### 3. *valuelists.ini*

Afin que l'utilisateur ait accès à une sélection de sessions et configurations, nous avons édité le fichier *valuelists.ini* de la table *lend*. Ce fichier sert normalement pour stocker des valeurs statiques, mais il est possible d'avoir recours à des valeurs dynamiques, provenant d'autres tables.

C'est ici notre cas puisque sont stockées dans la table '*lend*' les clés étrangères *fk\_session\_id* et *fk\_configuration\_id*, correspondant aux identifiants provenant des tables *session* et *configuration*.

Pour faire le lien entre ces identifiants et leurs vrais noms, nous devons définir deux requêtes SQL dans *valuelists.ini*:



```

1
2 [configs]
3 __sql__="select concat(configuration_id, ' - ',configuration_os, ' - ', configuration_version) from configuration"
4 [sessions]
5 __sql__="select concat(session_id, ' - ',session_name) from sessions"
6

```

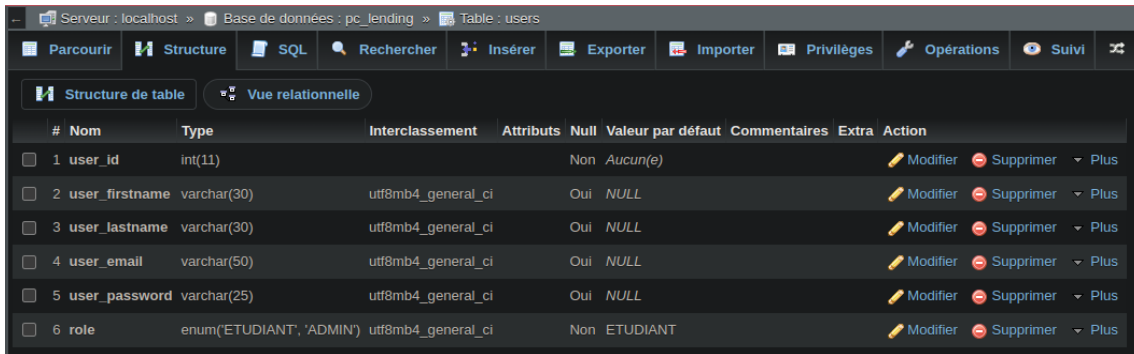
Figure 9: Fichier *valuelists.ini* de la vue 'Prêt en cours'.

C'est grâce à ces deux requêtes, concaténant les informations que l'on souhaite récupérer, qu'il est possible via le menu déroulant du formulaire, de choisir la configuration du PC ainsi que la session.



## G. Gestion des droits: rôles et permissions

Dans la table 'users', nous avons défini deux rôles: ETUDIANT et ADMIN. Lors de la création d'un nouvel utilisateur, seul un de ces deux rôles est attribuable car le champ 'role' de la table est de type ENUM:



#	Nom	Type	Interclassement	Attributs	Null	Valeur par défaut	Commentaires	Extra	Action
1	user_id	int(11)			Non	Aucun(e)			Modifier Supprimer Plus
2	user_firstname	varchar(30)	utf8mb4_general_ci		Oui	NULL			Modifier Supprimer Plus
3	user_lastname	varchar(30)	utf8mb4_general_ci		Oui	NULL			Modifier Supprimer Plus
4	user_email	varchar(50)	utf8mb4_general_ci		Oui	NULL			Modifier Supprimer Plus
5	user_password	varchar(25)	utf8mb4_general_ci		Oui	NULL			Modifier Supprimer Plus
6	role	enum('ETUDIANT', 'ADMIN')	utf8mb4_general_ci		Non	ETUDIANT			Modifier Supprimer Plus

Figure 10: Table 'users' de l'application de prêt.

Les utilisateurs dont le rôle est ETUDIANT n'ont accès en lecture seule qu'à la vue 'Prêt en cours' afin de visualiser leurs emprunts en cours alors que les ADMIN ont accès en lecture/écriture/suppression à l'ensemble des tables.

Dans le fichier *functions.inc.php* de l'application, nous nous servons de plusieurs fonctions qui permettent de récupérer le rôle affecté à l'utilisateur présentement connecté dans la table 'users' et de lui permettre de visualiser les informations en fonction de celui-ci.

```

functions.inc.php
1 <?php
2 /**
3  * @file inc/functions.php
4  * @brief This file is included at the beginning of the index.php file so it
5  * and its functions should be available to every part of the app.
6  */
7
8 /**
9  * @brief Gets the currently logged in user.
10  * @returns {Dataface_Record} Dataface_Record object encapsulating the row from the users
11  * table of the currently logged-in user. If no user is logged in, then this will return
12  * null.
13  */
14 function getUser(){
15     static $user = -1;
16     if (is_int($user) and $user == -1){
17         $user = Dataface_AuthenticationTool::getInstance()->getLoggedInUser();
18     }
19     return $user;
20 }
21
22
23 /**
24  * @brief Gets the role of the currently logged in user.
25  * @returns String The role of the currently logged in user (or null if none).
26  */
27 function getRole(){
28     static $role = -1;
29     if (is_int($role) and $role == -1){
30         $user = getUser();
31         if ( !$user ) return null;
32         $role = $user->val('role');
33     }
34     return $role;
35 }
36
37 /**
38  * @brief Checks if the currently logged-in user is an administrator.
39  * @returns boolean
40  */
41 function isAdmin(){
42     return (getRole() == 'ADMIN');
43 }

```

Figure 11: Fonctions du function.inc.php.

L'application delegate est une classe permettant d'implémenter des méthodes qui seront exécutées sur l'ensemble de l'application comme par exemple des permissions, des préférences d'utilisation, des déclencheurs...

La méthode 'getPermissions' permet de récupérer les autorisations associées au rôle de l'utilisateur. Dans notre cas, 'ETUDIANT' est en lecture seule et 'ADMIN' possède des droits d'écriture, d'édition et de suppression.

La méthode 'beforeHandleRequest' s'exécute avant toutes les autres requêtes et permet de filtrer l'affichage en fonction du rôle.

En effet, si le rôle de l'utilisateur est strictement égal à 'ETUDIANT', alors il sera dans l'incapacité de voir les tables 'formation', 'configuration', 'users' et 'sessions'. Elles seront également injoignables via leurs URL.

```

functions.inc.php  ApplicationDelegate.php x
conf > ApplicationDelegate.php
1  <?php
2
3  class conf_ApplicationDelegate {
4      function getPermissions(&$record){
5          $auth =& Dataface_AuthenticationTool::getInstance();
6          $user =& $auth->getLoggedInUser();
7          if ( !isset($user) ) return Dataface_PermissionsTool::NO_ACCESS();
8          $role = $user->val('role');
9          return Dataface_PermissionsTool::getRolePermissions($role);
10         // Returns all of the permissions for the user's current role.
11     }
12     function beforeHandleRequest() {
13
14         if ((getRole()) == 'ETUDIANT'){
15             $app =& Dataface_Application::getInstance();
16
17             unset($app->_conf['_tables']['formation']);
18             unset($app->_conf['_tables']['configuration']);
19             unset($app->_conf['_tables']['users']);
20             unset($app->_conf['_tables']['sessions']);
21
22             $app->_conf['_disallowed_tables']['rule_1'] = 'configuration';
23             $app->_conf['_disallowed_tables']['rule_2'] = 'sessions';
24             $app->_conf['_disallowed_tables']['rule_3'] = 'formation';
25             $app->_conf['_disallowed_tables']['rule_4'] = 'users';
26         }
27     }
28 }
29
30
31
32 ?>

```

Figure 12: Classe ApplicationDelegate.

---

# TROISIEME PARTIE: APPLICATION CANDIDATURE

## I. CONTEXTE

Depuis plusieurs années LDNR utilise une application web basée sur Xataface pour permettre aux candidats de postuler aux différentes formations proposées par le centre. Cette version a été développée par un ancien étudiant de Limayrac en BTS SIO. C'est dans ce contexte que Thomas et moi-même sommes intervenus pour la refaire de zéro.

## II. PRÉOPÉRATEUR: MCD, MLD, UML, GANTT/PERT, MODÉLISATION BDD

### A. MCD/MLD

Afin de modéliser la base de données candidature, nous avons utilisé le logiciel libre de droit Looping. Emile et Nina ont choisi de ne pas nous montrer la Bdd telle qu'elle était actuellement au moment du stage pour ne pas nous influencer lors de la modélisation.

Le MCD que nous avons proposé comporte 6 entités différentes. L'association ternaire 'is\_candidat' permet d'avoir en une table les informations indispensables à la consultation de candidatures en cours grâce notamment aux cardinalités qui lui sont affectées. Par ailleurs, cette association devient une table à part lors du passage au MLD. Le MLD et le MCD rognés sont consultables en annexe 5.

### B. Maquettes

Pour réaliser les maquettes interactives des interfaces utilisateurs et administrateurs nous avons utilisé le logiciel Balsamiq. Thomas s'est chargé de la partie administration alors que j'étais sur la partie candidats. Une partie des maquettes réalisées sont consultables à l'annexe 6. Afin de préserver la confidentialité du projet, seules les maquettes d'inscription et de connexion sont présentées visuellement.

Dans un souci d'ergonomie, nous avons opté pour des boutons moins nombreux mais plus gros, avec des symboles clairs représentant leur fonctionnalité.

De plus, la page d'accueil montre plus clairement le déroulement du procédé complet de candidature.

Une fois celle-ci lancée, et à la demande d'Emile et Nina, nous avons représenté l'avancement de la procédure pour que l'utilisateur puisse la visualiser.

## C. Diagramme de cas d'utilisation

La figure ci-dessous présente un diagramme de cas d'utilisation pour une personne souhaitant gérer sa ou ses candidatures en cours. L'administrateur et l'utilisateur doivent tous les deux passer par la connexion au site afin d'apporter des modifications.

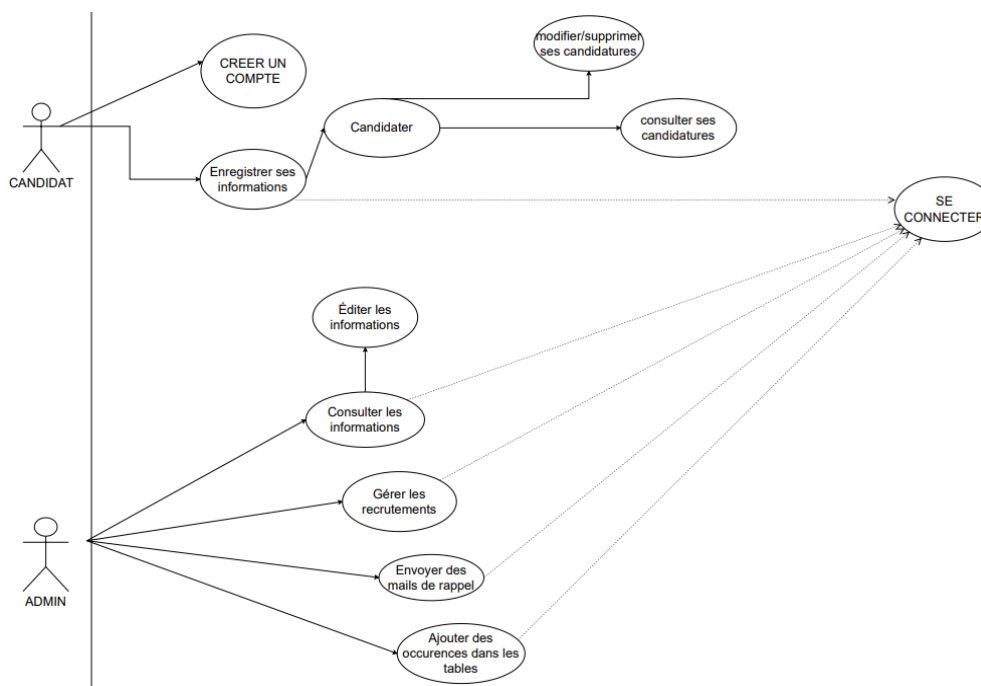


Figure 13: Diagramme de cas d'utilisation de l'application candidature.

## D. Diagramme prédictif de GANTT

La figure ci-dessous présente le diagramme de GANTT pour la planification des activités. Nous avons fait ce planning afin de mieux répartir les tâches et de visualiser le temps nécessaire à leur réalisation.

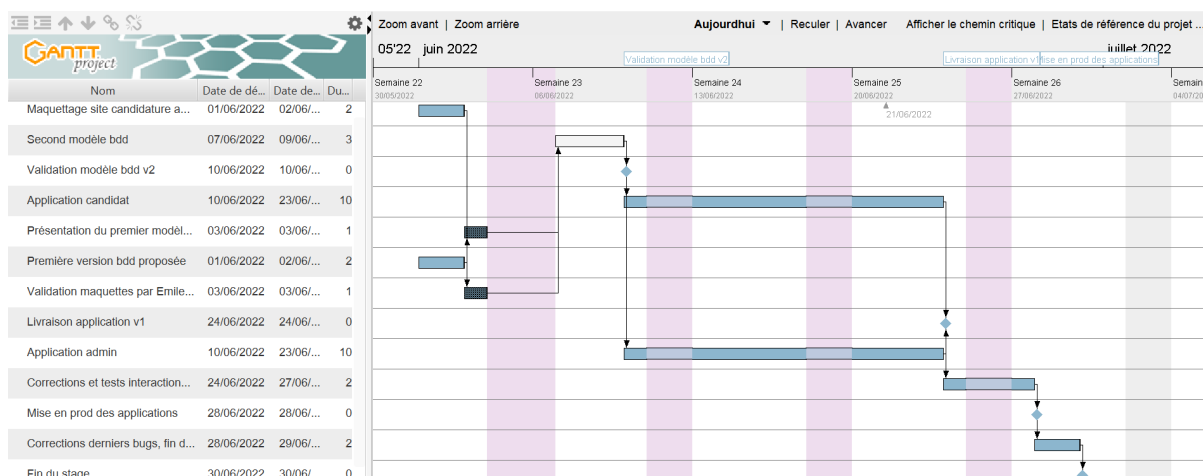


Figure 14: Diagramme prédictif de GANTT de l'application candidature.

### III. RÉALISATION DE L'APPLICATION CANDIDAT

Les différentes fonctions utilisées sont présentées les annexes 7-8 et 9, ce sont respectivement `db_functions.php`, `functions.inc.php` et l'Application Delegate de Xataface.

#### A. Création de la page d'accueil `index.html`

Afin de permettre à l'utilisateur de choisir entre se connecter et s'inscrire, j'ai créé une page '`index.html`' dédiée à cela, accessible à tous.

La première option permet à l'aide de l'attribut '`onclick`' de rediriger l'utilisateur si celui-ci clique sur la balise vers la page de connexion à l'application.

La seconde option redirige vers le formulaire d'inscription.

Le code est présenté en annexe 10.

#### B. Le formulaire d'inscription

##### 1. `Signup.html`

Le formulaire d'inscription a été réalisé avec la méthode POST et en respectant les valeurs énumérées dans la base de données.

Le code est présenté en annexe 11.

##### 2. `Inscription.php`

C'est grâce à cette partie que va se faire l'ajout dans la table 'utilisateur' de la personne nouvellement inscrite.

La fonction `db_connect()` permet de se connecter à la base de données.

Ensuite nous récupérons les valeurs entrées dans les différents champs du formulaire par la méthode POST.

Enfin, la requête SQL ne sera exécutée que si `$submit` et `$cgu`, qui sont toutes les deux des variables booléennes, sont égales à 1.

Suite à l'ajout dans la table, l'utilisateur est redirigé vers la page de connexion grâce à la fonction '`header()`'.

Le code est présenté en annexe 12.

### 3. Mdpcheck.js

Ce petit script JavaScript permet d'afficher de façon dynamique si le mot de passe entré dans la case de vérification du mot de passe correspond bien à celui saisi par l'utilisateur.

Le code est présenté ci-dessous:

```
var check = function() {  
    if (document.getElementById('mdp').value ==  
        document.getElementById('confirm_mdp').value) {  
        document.getElementById('message').style.color = 'green';  
        document.getElementById('message').innerHTML = 'Les mots de passe correspondent';  
    } else {  
        document.getElementById('message').style.color = 'red';  
        document.getElementById('message').innerHTML = 'Les mdp ne correspondent pas!';  
    }  
}
```

Figure 15: mdpcheck.js de l'application candidature.

## C. Permissions

Comme pour l'application de prêt, c'est la fonction 'getPermissions()' de la classe 'ApplicationDeleguate' qui permet de récupérer les permissions associées à la personne connectée. Elle est visualisable à l'annexe 9.

Dans le cas de la candidature, l'administrateur et les candidats ont les mêmes droits.

En revanche, si la personne connectée n'est pas un admin, alors un filtre s'applique pour n'afficher que les informations relatives à son identifiant grâce à la fonction 'getUser' présentée dans l'annexe 8.

## IV. Conclusion

Ce stage a également été l'occasion pour moi de découvrir Xataface et de manière plus générale, les frameworks et de valider des compétences du bloc n°1 pour l'épreuve E4 du BTS SIO. Malheureusement par manque de temps, l'application de candidature n'a pas pu être finalisée. Cependant, j'ai pu acquérir de nouvelles compétences en PHP grâce notamment à l'utilisation d'orienté objets. Ce tableau de synthèse est présenté ci-dessous :

**TABLEAU DE SYNTHÈSE – ACTIVITÉS – BLOCS DE COMPÉTENCES – UNITÉS**

Brevet de technicien supérieur BTS services informatiques aux organisations

Activités	Blocs de compétences	Unités
<b>Domaine d'activité 1 – Support et mise à disposition de services informatiques</b> Gestion du patrimoine informatique Réponse aux incidents et aux demandes d'assistance et d'évolution Développement de la présence en ligne de l'organisation Travail en mode projet Mise à disposition des utilisateurs d'un service informatique Organisation de son développement professionnel	<b>Bloc n° 1 – Support et mise à disposition de services informatiques</b> Gérer le patrimoine informatique Répondre aux incidents et aux demandes d'assistance et d'évolution Développer la présence en ligne de l'organisation Travailler en mode projet Mettre à disposition des utilisateurs un service informatique Organiser son développement professionnel	<b>UNITÉ U4</b> Support et mise à disposition de services informatiques

Figure 16: Extrait du tableau de synthèse des compétences du B1 du BTS SIO.

Voici les activités réalisées qui valident successivement les attentes du B1:

- Répondre aux incidents et aux demandes d'assistance et d'évolution: la demande formulée par Emile et Nina, la refonte et l'amélioration de l'application de candidature à LDNR est l'élément central de la mission. Cette demande d'évolution est également passée par la refonte de la BdD et nous avons tous ensemble repensé l'ergonomie du site pour accompagner au maximum l'utilisateur dans ses démarches.
- Travailler en mode projet: Thomas et moi-même avons utilisé GitLab afin de synchroniser et d'uniformiser nos codes et développements. Le diagramme de GANTT nous a également permis de mieux nous répartir les tâches et de visualiser les étapes critiques.
- Organiser son développement professionnel: Nina et Emile nous ont entraînés, Thomas et moi, durant des simulations d'entretiens et nous ont aidé à améliorer nos CV. Nous avons également eu la chance de participer au salon DIGI'TALENT où nous avons rencontré des recruteurs de différentes entreprises du numérique.

Malgré le fait que nous n'ayons pas terminé l'application à temps, ce qui a été fait durant le stage servira à Emile et Nina pour finaliser l'application.

## QUATRIÈME PARTIE: Annexes

### Annexe 1: script de création de la BdD prêt

```
CREATE TABLE users(  
    user_id INT AUTO_INCREMENT,  
    user_firstname VARCHAR(30),  
    user_lastname VARCHAR(30),  
    user_email VARCHAR(50),  
    user_password VARCHAR(25),  
    PRIMARY KEY(user_id)  
);  
CREATE TABLE formation(  
    formation_id INT AUTO_INCREMENT,  
    formation_name VARCHAR(80),  
    PRIMARY KEY(formation_id)  
);  
CREATE TABLE session(  
    session_id INT AUTO_INCREMENT,  
    session_name VARCHAR(50),  
    session_start DATE,  
    session_end DATE,  
    fk_formation_id INT NOT NULL,  
    PRIMARY KEY(session_id)  
);  
CREATE TABLE configuration(  
    configuration_id INT AUTO_INCREMENT,  
    configuration_os VARCHAR(50),  
    configuration_version VARCHAR(30),  
    PRIMARY KEY(configuration_id)  
);  
CREATE TABLE role(  
    role_id INT AUTO_INCREMENT,  
    role_name VARCHAR(25),  
    PRIMARY KEY(role_id)  
);  
CREATE TABLE user_role(  
    fk_user_id INT,  
    fk_role_id INT,  
    PRIMARY KEY(fk_user_id, fk_role_id)  
);  
CREATE TABLE lend(  
    fk_users_id INT,  
    fk_session_id INT,  
    fk_configuration_id INT,  
    lend_id INT AUTO_INCREMENT,  
    PRIMARY KEY(lend_id)  
);
```



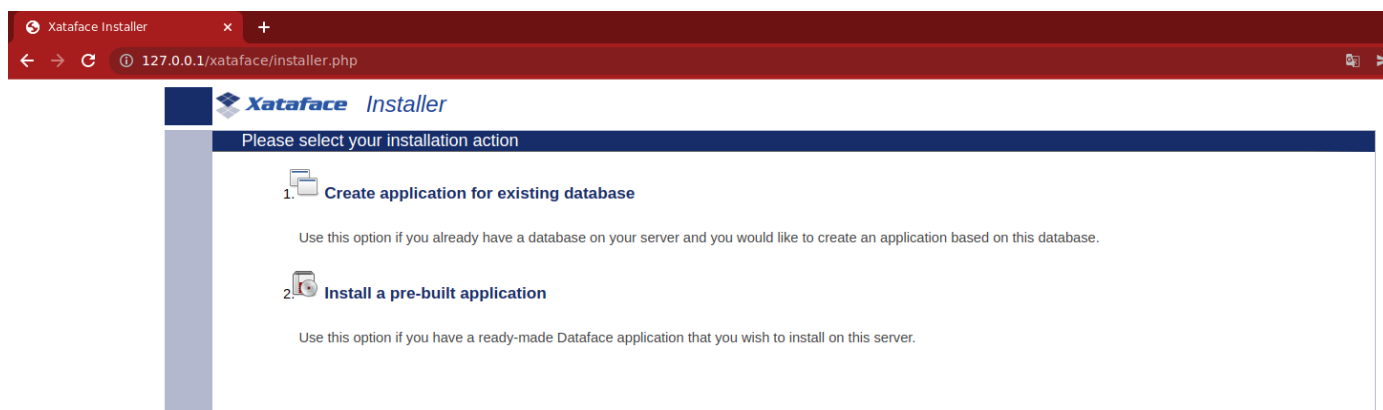
## Annexe 2: protocole d'installation de Xataface

Dans un premier temps en local, dans le répertoire /var/www/html/xataface, il faut vérifier que l'installer soit en *.enabled*.

S'il est en *.disabled* il suffit de le renommer en *installer.enabled*:

```
stag@formation-ldnr-5CG4504BTD: /var/www/html/xataface$ ls
403.html      config.inc.php  DB              index.php       js.php          makesite        permissions.ini.php  README.md      tests
actions        css             fonts           init.php        lang            metadata.ini.php plone.css          scaffold_template tools
actions.ini.php css.php         help            install          lib             modules         plone_javascripts.js  setup.php      uilibs
bin            Dataface        HTML            installer.disabled LICENSE.txt      nbproject       plone_javascripts-src.js  site_skeleton version.txt
build.xml      dataface_info.php  iframe.css      installer.php    logo.png        PEAR            plone_menu.js       snippets      xf
changes.txt    dataface-public-api.php  images         js              logo-small.png  PEAR.php       public-api.php       SQL
stag@formation-ldnr-5CG4504BTD: /var/www/html/xataface$ sudo mv installer.disabled installer.enabled
stag@formation-ldnr-5CG4504BTD: /var/www/html/xataface$ ls
403.html      config.inc.php  DB              index.php       js.php          makesite        permissions.ini.php  README.md      tests
actions        css             fonts           init.php        lang            metadata.ini.php plone.css          scaffold_template tools
actions.ini.php css.php         help            install          lib             modules         plone_javascripts.js  setup.php      uilibs
bin            Dataface        HTML            installer.enabled LICENSE.txt      nbproject       plone_javascripts-src.js  site_skeleton version.txt
build.xml      dataface_info.php  iframe.css      installer.php    logo.png        PEAR            plone_menu.js       snippets      xf
changes.txt    dataface-public-api.php  images         js              logo-small.png  PEAR.php       public-api.php       SQL
stag@formation-ldnr-5CG4504BTD: /var/www/html/xataface$
```

Se rendre sur l'installer, depuis le localhost à l'adresse suivante: 127.0.0.1/xataface/installer.php:



Dans 'Create application for existing database', l'étape 1 permet de sélectionner la BdD phpMyAdmin que l'on souhaite raccorder à Xataface à l'aide d'un menu déroulant.

L'étape 2 permet de saisir le nom d'utilisateur et le mot de passe utilisés pour se connecter à la BdD. Une fois le test de connexion effectué et concluant, il est possible de passer à l'étape suivante.

## Suite Annexe 2: protocole d'installation de Xataface

Lors de cette étape, il est demandé de choisir le type d'installation que l'on souhaite effectuer:

### Step 3: Select Installation Type

You can either install the application directly on your web server (requires FTP connection information), or download the application as a tar archive, so that you can install it on your server manually.

Please select your preferred method of installation:

Nous passons par **Tarball** qui résulte de la compression de fichiers d'archives.

Dans `/var/www/html`, il faut créer le répertoire qui va contenir notre application à l'aide de la commande suivante: `sudo mkdir ldnr_lending`

Dans ce répertoire nouvellement créé, nous créons le dossier `tables`, et à l'intérieur de ce dossier, autant de dossiers que de tables présentes dans la BdD:

```
root@formation-ldnr-5CG4504BTD:/var/www/html# mkdir -p ldnr_lending/tables/{users, formation, session, configuration, role, user_role}
```

Nous avons alors l'arborescence suivante:

`/var/www/html/ldnr_lending`

- tables
- configuration
- formation
- lend
- session
- users
- role
- user\_role

Après avoir récupéré le dossier `.tar.gz`, il faut le décompresser dans le dossier `ldnr_lending` grâce à la commande suivante:

```
root@formation-ldnr-5CG4504BTD:/var/www/html# tar -xzf ldnr_lendingEbs0WF.tar.gz -C ldnr_lending/
```

Pour finaliser le lien entre xataface et la BdD, dans `/var/www/html/ldnr_lending`, il faut, dans un premier temps, créer un dossier nommé `templates_c` afin d'initialiser le cache de l'application.

Enfin, nous devons donner les droits de lecture/écriture au serveur web apache à l'aide de la commande suivante:

```
root@formation-ldnr-5CG4504BTD:/var/www/html# chown -R www-data:www-data ldnr_lending
```

## Annexe 3: fichier relationships.ini de la table lend

```
≡ conf.ini ldnr_lending  ≡ conf.ini pc_lending  ≡ relationships.ini ×
pc_lending > tables > lend > ≡ relationships.ini
1  [Sessions]
2  sessions.session_id="$fk_session_id"
3  [Configurations]
4  configuration.configuration_id="$fk_configuration_id"
5  [Users]
6  users.user_id="$fk_users_id"
7  
```

## Annexe 4: requête pour la génération de la vue 'prêt en cours'

Exécuter une ou des requêtes SQL sur la table « **pc\_lending.lend** »:

```
1 select `l`.`lend_id` AS `lend_id`,
2 concat(`u`.`user_firstname`,` `,`u`.`user_lastname`) AS `identity`,
3 `f`.`formation_name` AS `formation_name`,
4 `s`.`session_name` AS `session_name`,
5 concat(`c`.`configuration_os`,` `,`c`.`configuration_version`) AS `configuration`,
6 `l`.`lend_ldnr_number` AS `lend_ldnr_number`,
7 `l`.`lend_checknumber` AS `lend_checknumber`,
8 if(`l`.`lend_keyboard`,`Oui`,`Non`) AS `Clavier`,
9 if(`l`.`lend_mouse`,`Oui`,`Non`) AS `Souris`,
10 `u`.`user_id` AS `user_id`
11 from (((`pc_lending`.`lend` `l` join `pc_lending`.`sessions` `s`) join `pc_lending`.`configuration` `c`) join
12 `pc_lending`.`formation` `f`) join `pc_lending`.`users` `u`)
13 where `l`.`fk_users_id` = `u`.`user_id`
14 and `l`.`fk_session_id` = `s`.`session_id`
15 and `l`.`fk_configuration_id` = `c`.`configuration_id`
16 and `s`.`fk_formation_id` = `f`.`formation_id`;
```

✓ Affichage des lignes 0 - 6 (total de 7, traitement en 0.0052 seconde(s).)

```
select `l`.`lend_id` AS `lend_id`, concat(`u`.`user_firstname`,` `,`u`.`user_lastname`) AS `identity`, `f`.`formation_name` AS `formation_name`, ,
`s`.`session_name` AS `session_name`, concat(`c`.`configuration_os`,` `,`c`.`configuration_version`) AS `configuration`, `l`.`lend_ldnr_number` AS
`lend_ldnr_number`, `l`.`lend_checknumber` AS `lend_checknumber`, if(`l`.`lend_keyboard`,`Oui`,`Non`) AS `Clavier`, if(`l`.`lend_mouse`,`Oui`,`Non`) AS
`Souris`, `u`.`user_id` AS `user_id` from (((`pc_lending`.`lend` `l` join `pc_lending`.`sessions` `s`) join `pc_lending`.`configuration` `c`) join
`pc_lending`.`formation` `f`) join `pc_lending`.`users` `u`) where `l`.`fk_users_id` = `u`.`user_id` and `l`.`fk_session_id` = `s`.`session_id` and
`l`.`fk_configuration_id` = `c`.`configuration_id` and `s`.`fk_formation_id` = `f`.`formation_id`;
```

☐ Profilage [ Éditer en ligne ] [ Éditer ] [ Expliquer SQL ] [ Créer le code source PHP ] [ Actualiser ]






☐ Tout afficher | Nombre de lignes : 25 | Filtrer les lignes: Chercher dans cette table

+ Options

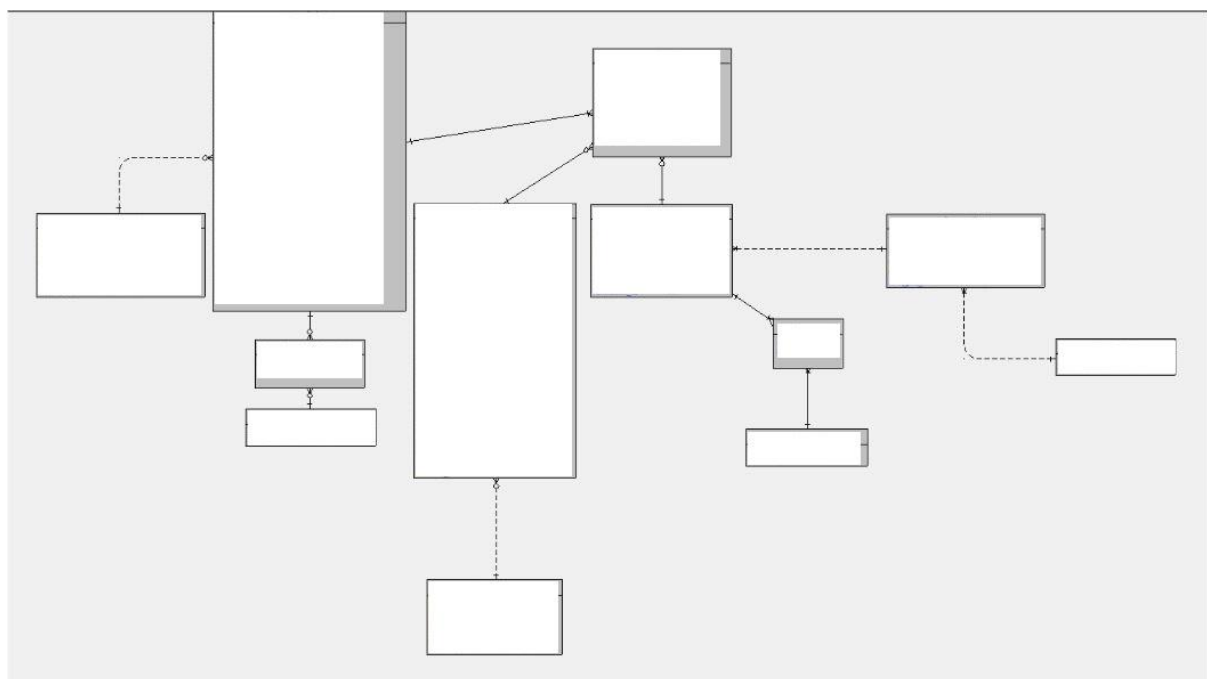
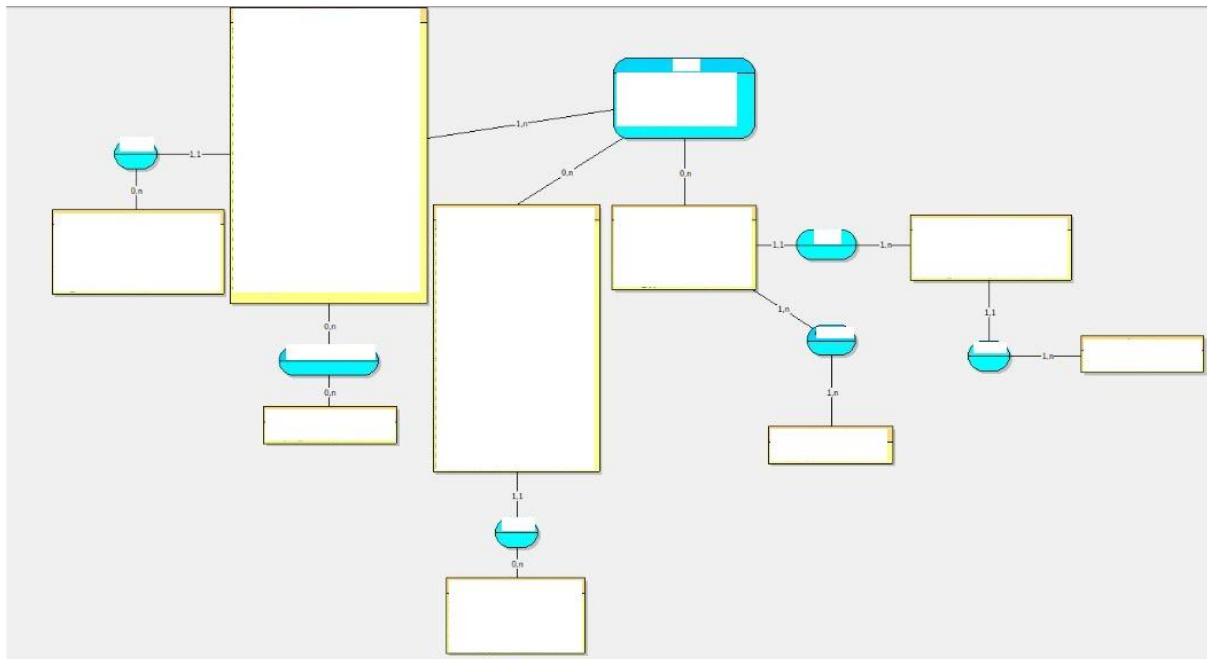
lend_id	Identity	formation_name	session_name	configuration	lend_ldnr_number	lend_checknumber	Clavier	Souris	user_id
8	Thomas LANDES	Langage C – initiation	C_INI_2022-2023	Debian 9.13	5CG6165BR1	1234567	Oui	Oui	1
9	Benoit DUPOND	Python initiation	Python_INI_2022/2023	Debian 8.2	14CG234567	2645367	Non	Oui	3
10	Alexandre DUPOND	Algorithmique – initiation (base de la programmat...	ALGO_INI_2022-2023	Debian 9.13	14CG234567	1234567	Non	Oui	2
11	Benoit DUPOND	Python initiation	Python_INI_2022/2023	Debian 9.13	17CG234567	4564231	Oui	Non	3
12	Alexandre DUPOND	Langage C – initiation	C_INI_2022-2023	Windows Professional 10.1	19CG234567	4564231	Non	Non	2
13	Benoit DUPOND	Python initiation	Python_INI_2022/2023	Debian 9.13	17CG234567	4564231	Oui	Non	3
14	Benoit DUPOND	Langage C – initiation	C_INI_2022-2023	Debian 8.2	19CG234567	4564231	Non	Oui	3

☐ Tout afficher | Nombre de lignes : 25 | Filtrer les lignes: Chercher dans cette table

Opérations sur les résultats de la requête

 Imprimer  Copier dans le presse-papiers  Exporter  Afficher le graphique  Créer une vue

## Annexe 5: MCD/MLD de l'application candidature



## Annexe 6: Maquettes accueil et inscription

The image displays two wireframe mockups of a web application interface, presented as browser windows.

**Top Mockup (Login Page):**

- Header:** "BIENVENUE SUR LE SITE DE GESTION DES CANDIDATURES POUR LES FORMATIONS LDNR" and "VEUILLEZ VOUS CONNECTER OU CREER UN COMPTE".
- Form Area:**
  - Option 1: "J'ai déjà un compte" with a user icon and a "Je me connecte" button.
  - Option 2: "Je n'ai pas encore de compte" with a user icon and a "+" sign, and a "Je crée un compte" button.
- Footer:** Logos for "L'ECOLE REGIONALE DU NUMERIQUE", "Occitanie", "GEN", "pôle emploi", and "Qualiopi processus certifié".

**Bottom Mockup (Registration Page):**

- Header:** "BIENVENUE SUR LE SITE DE GESTION DES CANDIDATURES POUR LES FORMATIONS LDNR" and "CREATION DE COMPTE".
- Form Area:**
  - Fields for "Nom", "Prénom", "Adresse mail", and "Numéro téléphone".
  - A checkbox labeled "Ces données" and a button "Inscription effectuée avec succès".
  - A link "Retour à l'accueil".
  - Fields for "Mot de passe" (password) and "Mot de passe" (confirm password).
  - A button "S'inscrire".
- Footer:** Logos for "L'ECOLE REGIONALE DU NUMERIQUE", "Occitanie", "GEN", "pôle emploi", and "Qualiopi processus certifié".

## Annexe 7: db\_functions.php de l'application candidature

```
<?php
//
// Connexion à la base de données
//
function db_connect() {
    $dsn = 'mysql:host=localhost;dbname=Candidature'; // contient le nom du serveur et de la base
    $user = 'root';
    $password = '12345';
    try {
        $dbh = new PDO($dsn, $user, $password, array(PDO::MYSQL_ATTR_INIT_COMMAND => "SET NAMES utf8"));
        $dbh->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
    } catch (PDOException $ex) {
        die("Erreur lors de la connexion SQL : " . $ex->getMessage());
    }
    return $dbh;
}

//
// Selection des données du user connecté en fonction de son ID
//
function db_find_candidate($id)
{
    global $dbh; // Récupère la connexion dans le contexte global
    $sql = "select * from utilisateur where utilisateur_id=:id";
    try {
        $sth = $dbh->prepare($sql);
        $sth->execute(array(":id" => $id));
        $row = $sth->fetch(PDO::FETCH_ASSOC);
    } catch (PDOException $e) {
        die("<p>Erreur lors de la requête SQL : " . $e->getMessage() . "</p>");
    }
    return $row; // On retourne l'enregistrement
}
?>
```

## Annexe 8: functions.inc.php de l'application candidature

```
<?php
function getUser(){
    static $user = -1;
    if (is_int($user) and $user == -1 ){
        $user = Dataface AuthenticationTool::getInstance()->getLoggedInUser();
    }
    return $user;
}

function getRole(){
    static $role = -1;
    if ( is_int($role) and $role == -1 ){
        $user = getUser();
        if ( !$user ) return null;
        $role = $user->val('utilisateur_role');
    }
    return $role;
}

function getId(){
    static $id = -1;
    if ( is_int($id) and $id == -1 ){
        $user = getUser();
        if ( !$user ) return null;
        $id = $user->val('utilisateur_id');
    }
    return $id;
}
?>
```



## Annexe 9: Application Delegate de l'application candidature

```
<?php
class conf_ApplicationDelegate
{
    function getPermissions(&$record)
    {
        $auth = &Dataface_AuthenticationTool::getInstance();
        $user = &$auth->getLoggedInUser();
        if (!isset($user)) return Dataface_PermissionsTool::NO_ACCESS();
        $role = $user->val('utilisateur_role');
        return Dataface_PermissionsTool::getRolePermissions($role);
        // Returns all of the permissions for the user's current role.
    }
    function beforeHandleRequest()
    {
        xf_stylesheet('css/footer.css', false);
        $app = &Dataface_Application::getInstance();
        $query = &$app->getQuery();
        if ($query['-table'] == 'dashboard' and ($query['-action'] == 'browse' or $query['-action'] == 'list')) {
            $query['-action'] = 'dashboard';
        }
    }
}
```

## Annexe 10: index.html de l'application candidature

```
<!DOCTYPE html>
<html lang="fr">

<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <link rel="stylesheet" href="css/index.css">
  <link href="https://maxcdn.bootstrapcdn.com/font-awesome/4.6.3/css/font-awesome.min.css" rel="stylesheet"
    integrity="sha384-T8GyShrQnKt+hzMcL4p1YTQ06cYprQmhrYwI1Q/3axmI1hQomh7Ud2hP0y85P1" crossorigin="anonymous">
  <title>LDNR CANDIDATURE</title>
</head>

<body>
  <h1 class="h1">
    BIENVENUE SUR LE SITE DE GESTION DES CANDIDATURES POUR LES FORMATIONS LDNR<br>
    VEUILLEZ VOUS CONNECTER OU CREER UN COMPTE
  </h1>
  <div class="frame">
    <div class="frame_connection">
      onclick="location.href='http://127.0.0.1/Candidature/index.php?action=login_prompt&table=dashboard&cursor=0&skip=0&limit=30&mode=list';"
      style="cursor: pointer;">
      <span class="text">
        J'ai déjà un compte.<br>
        <i class="fa fa-sign-in" aria-hidden="true"></i><br>
        Je me connecte!<br>
      </span>
    </div>
    <div class="frame_inscription">
      onclick="location.href='http://127.0.0.1/Candidature/signup.html';"
      style="cursor: pointer;">
      <span class="text">
        Je n'ai pas de compte.<br>
        <i class="fa fa-user-plus" aria-hidden="true"></i><br>
        Je m'inscris!<br>
      </span>
    </div>
  </div>
  <div>
    
  </div>
</body>

</html>
```

## Annexe 11: Signup.html de l'application candidature

```
<!DOCTYPE html>
<html lang="fr">

<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <link rel="stylesheet" href="css/signup.css">
  <title>LDNR Inscription</title>
</head>

<body>
  <h1 class="h1">
    BIENVENUE SUR LE SITE DE GESTION DES CANDIDATURES POUR LES FORMATIONS LDNR<br>
    INSCRIPTION
  </h1>
  <div class="frame">
    <form id="form_container" action="inscription.php" method="post">
      <select name="civ_candidat" id="civ" required>
        <option value="H">Monsieur</option>
        <option value="F">Madame</option>
      </select>

      <input type="text" name="prenom_candidat" placeholder="Prénom" id="firstname" required><br>
      <input type="text" name="nom_candidat" placeholder="Nom de famille" id="lastname" required><br>
      <input type="email" name="mail_candidat" placeholder="Votre adresse mail" id="email" required><br>
      <input type="tel" name="telephone_candidat" placeholder="Votre numéro de téléphone" id="tel" required><br>
      <input type="checkbox" name="CGU" id="cgu" required>Conditions générales d'utilisation des données<br>

      <p>En cochant cette case, vous acceptez que les données collectées sur le formulaire de candidature aux
        formation seront communiquées aux seuls destinataires suivants :
        LDNR, financeurs / co-financeur de la formation. Les données sont conservées pendant le temps nécessaire
        à la réalisation des traitements et au respect des obligations légales et contractuelles de LDNR l'égard
        des financeurs / co-financeurs de la formation. Conformément à l'article 34 de la loi Informatique et
        Libertés n°78-17 du 6 janvier 1978, vous pouvez accéder aux données vous concernant, les rectifier,
        demander leur effacement ou exercer votre droit à la limitation du traitement de vos données. Vous
        pouvez retirer à tout moment votre consentement au traitement de vos données. Vous pouvez également vous
        opposer au traitement de vos données en nous contactant par courriel à contact@ldnr.fr. Consultez le
        site www.cnil.fr pour plus d'informations sur vos droits.
      </p>
      <script src="JS/mdpcheck.js"></script>
      <input type="password" name="pwd_candidat" placeholder="Mot de passe" id="mdp" required onkeyup="check()"><br>
      <input type="password" name="pwd_candidat" placeholder="Mot de passe" id="confirm_mdp" required onkeyup="check()"><br>
      <span id="message"></span>
      <p><input type="reset" value="Réinitialiser" id="reset"><input type="submit" name="submit" value="Inscription" id="submit"></p>
    </form>
  </div>
  <div>
    
  </div>
</body>

</html>
```

## Annexe 12: Inscription.php de l'application candidature

```
<?php
include 'functions/db_functions.php';
// Connexion à la base
$dbh=db_connect();
// Lecture du formulaire
$civ = isset($_POST['civ_candidat']) ? $_POST['civ_candidat'] : '';
$prenom = isset($_POST['prenom_candidat']) ? $_POST['prenom_candidat'] : '';
$nom=isset($_POST['nom_candidat']) ? $_POST['nom_candidat'] : '';
$mail = isset($_POST['mail_candidat']) ? $_POST['mail_candidat'] : '';
$tel = isset($_POST['telephone_candidat']) ? $_POST['telephone_candidat'] : '';
$pwd=isset($_POST['pwd_candidat']) ? $_POST['pwd_candidat'] : '';
$cgu=isset($_POST['CGU']) ? $_POST['CGU'] : '';
$submit = isset($_POST['submit']);

// Ajout dans la base
if ($submit && $cgu){
    $sql="
insert into utilisateur (utilisateur_prenom,
utilisateur_nom,
utilisateur_email,
utilisateur_tel,
utilisateur_mdp,
utilisateur_civ,
utilisateur_cgu) values (:prenom,:nom,:mail,:tel,:pwd, :civ, :cgu);
    try {
        $sth = $dbh->prepare($sql);
        $sth->execute(array(
            ":prenom"=>$prenom,
            ":nom"=>$nom,
            ":mail"=>$mail,
            ":tel"=>$tel,
            ":pwd"=>$pwd,
            ":civ"=>$civ,
            ":cgu"=>$cgu
        ));
        $nb = $sth->rowCount();
    } catch (PDOException $e) {
        die( "<p>Erreur lors de la requête SQL : " . $e->getMessage() . "</p>");
    }
    echo "Inscription effectuée avec succès! Vous allez être automatiquement redirigé vers la page de connexion.";
    header( "Refresh:3; url=http://127.0.0.1/Candidature/index.php?action=login_prompt&table=dashboard&cursor=0&skip=0&limit=30&mode=list", true, 303);
} else {
    $message="Problème rencontré lors de l'inscription";
}
?>
```