

Documentation Fonctionnelle et Technique Détaillée

LIVE QUESTION

2020

Table des matières

I. Historique des modifications

II. Introduction

1. Contexte
2. Objectif du site
3. Organisation du projet
4. Périmètre du projet
5. Rôles de la plateforme
6. Définitions des termes

III. Les données

7. Dictionnaire de données
8. Modèle Entité-Association
9. Schéma Relationnel
10. Script de création

IV. Les fonctionnalités

11. Backlog
12. Arborescence
13. Cas d'utilisation
14. Maquettes

V. Dossier Technique

- 15. Fonctionnalité : Poser des questions
- 16. Fonctionnalité : Ajouter en ami
- 17. Fonctionnalité : Rechercher une question

VI. Veille contextualisée

VII. Conduite de projet

- 18. Répartition des tâches
- 19. Compte Rendu

I. Historique des modifications

Date	Nom	Description de la version
2020-04-30	Ridel Elisa, Thiesset Marius, Monteiro Hugo	Version PHP 7 procédurale
2020-12-21	Monteiro Hugo	Version Symfony 5

II. Introduction

1. Contexte

Live Question est un réseau social permettant à des utilisateurs de poser et de répondre à des questions. Il a été réalisé dans le cadre d'un stage interne lors de ma première année en BTS au Lycée Saint-Vincent de Senlis pour la version PHP 7 procédurale et en groupe de 3 personnes.

La seconde version du projet est une initiative personnelle de refonte du site en utilisant le framework Symfony 5 afin de m'auto-former à Symfony 5.

2. Objectif du site

Live Question donne la possibilité de s'inscrire ou de se connecter, le site autorise les internautes à consulter la page d'accueil et à s'inscrire sur le site. De surcroît, le site accorde aux utilisateurs le droit de se connecter sur le site, d'interagir entre eux en se posant des questions publiques ou privées et de s'ajouter en amis.

3. Organisation du projet

La première version du projet a été réalisé en groupe de 3 personnes qui est Ridel Elisa, Thiesset Marius et Monteiro Hugo.

La seconde version du projet a été réalisé uniquement par Monteiro Hugo.

4. Périmètre du projet

Live Question est composé de 7 parties qui sont :

- Page d'Accueil (non connecté)
- Page d'Accueil (connecté)
- Page des questions
- Page d'une question
- Page utilisateur
- Page du profil
- Page d'administration

5. Rôles de la plateforme

Live Question recueille 3 types de profils avec des accès distincts :

- Les **internauts** qui ont accès uniquement à la page d'accueil non connecté
- Les **membres** qui ont accès à toutes les fonctionnalités du site hormis le système d'administration
- Les **administrateurs** qui ont les mêmes accès que les utilisateurs ainsi que l'accès au système d'administration permettant l'ajout, suppression et modification de contenu ou utilisateur.

6. Définitions des termes

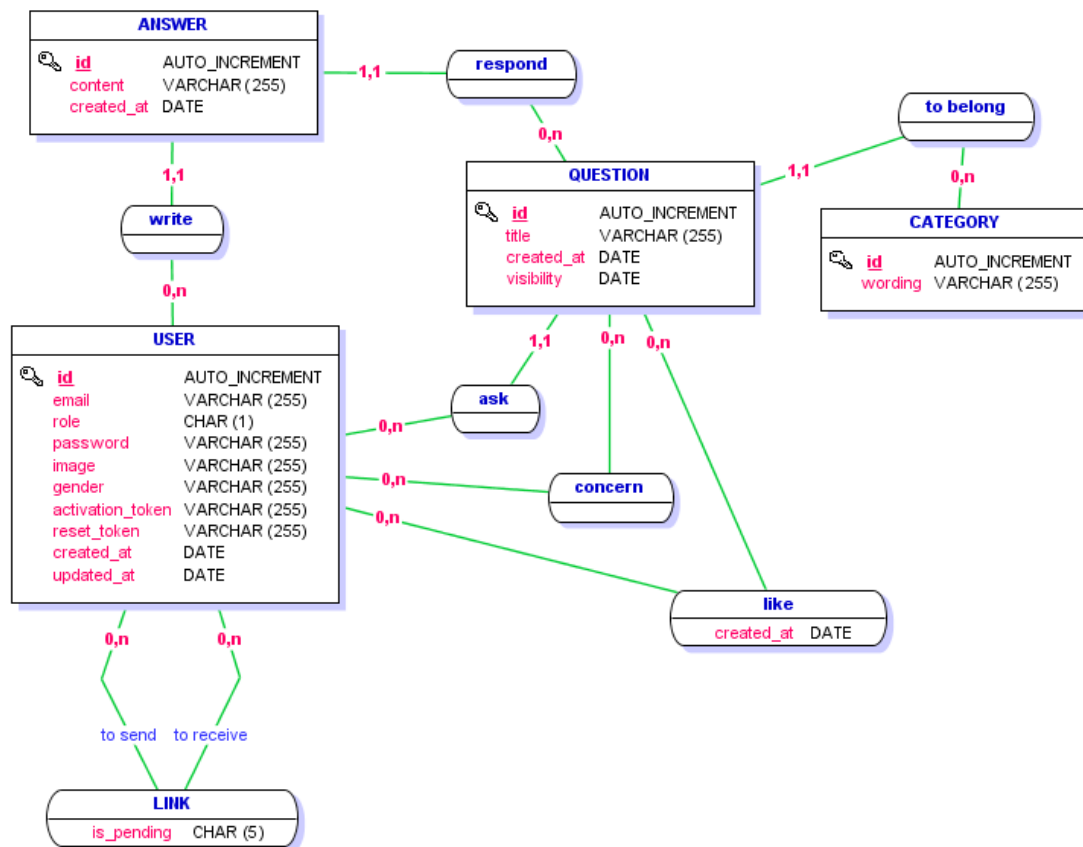
Terme	Définition
Internaute	Un internaute est un utilisateur du site Live Question dont on ne connaît pas l'identité, outre celle fournie par son fureteur (adresse IP, langue, etc ...)
Membre	Un internaute qui a complété son inscription et qui a fourni son adresse courriel.
Administrateur	L'administrateur est un membre particulier qui possède des droits d'administration sur le site.

III. Les données

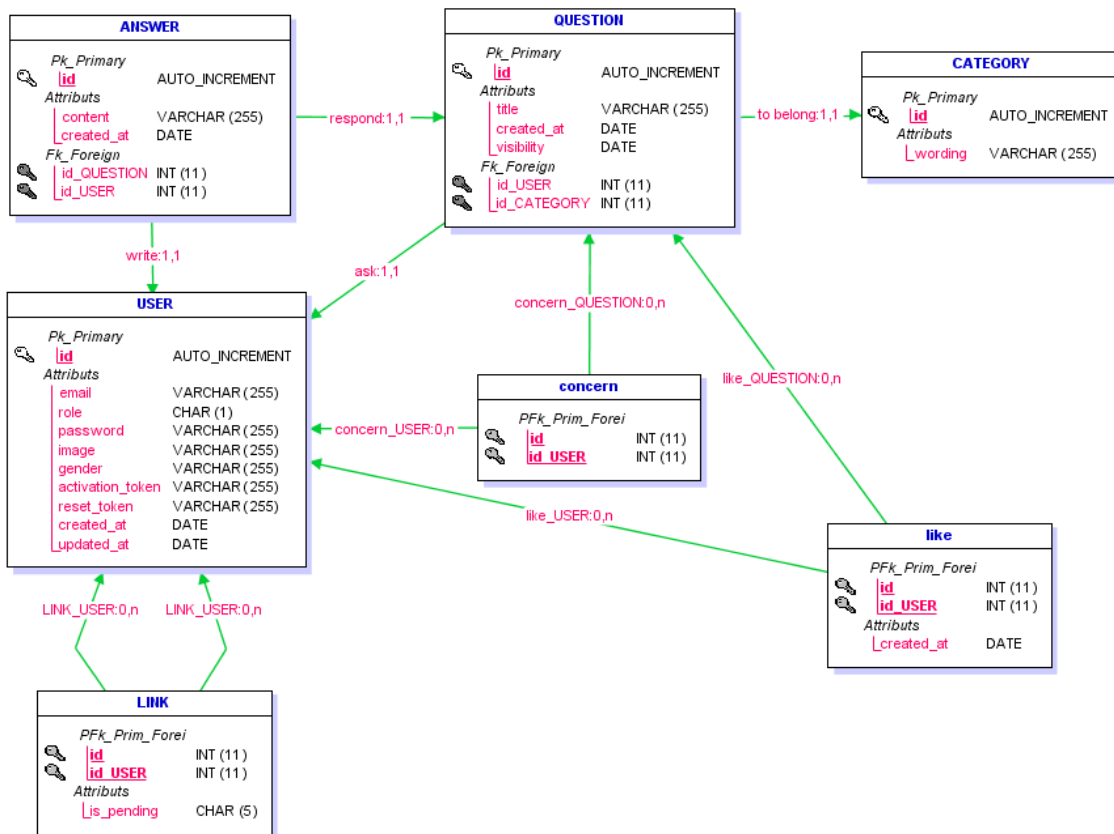
7. Dictionnaire de données

	A	B	C	D	E	F	G
1	Nom	Code	Description	Type	Taille	Spécification	
2							
3	id_user	idUser	L'identifiant de l'utilisateur	int	11	AI	
4	username_user	usernameUser	Le nom de l'utilisateur	varchar	255		
5	email_user	emailUser	L'email de l'utilisateur	varchar	255		
6	role_user	roleUser	Le role de l'utilisateur	char	1		
7	password_user	passwordUser	Le mot de passe de l'utilisateur	varchar	255		
8	image_user	imageUser	L'image de profil de l'utilisateur	varchar	255		
9	gender_user	genderUser	Le genre de l'utilisateur	varchar	255		
10	activation_token_user	activationUser	La clé d'activation du compte de l'utilisateur	varchar	255		
11	reset_token_user	resetUser	La clé de réinitialisation de l'utilisateur	varchar	255		
12	created_at_user	createdUser	La date de création de l'utilisateur	date			
13	updated_at_user	updatedUser	La date de maj de l'utilisateur	date			
14							
15	id_question	idQuestion	L'identifiant de la question	int	11	AI	
16	ref_category_question	refCategoryQuestion	La catégorie de la question	int	11	Index	
17	ref_user_question	refUserQuestion	L'utilisateur qui a écrit la question	int	11	Index	
18	title_question	titleQuestion	Le titre de la question	varchar	255		
19	created_at_question	createdAtQuestion	La date de création de la question	date			
20	visibility_question	visibilityQuestion	Le type de visibilité de la question	varchar	255		
21							
22	id_answer	idAnswer	L'identifiant de la réponse	int	11	AI	
23	ref_question_answer	refQuestionAnswer	La question lié à la réponse	int	11	Index	
24	ref_user_answer	refUserAnswer	L'utilisateur qui a écrit la réponse	int	11	Index	
25	content_answer	contentAnswer	Le contenu de la réponse	varchar	255		
26	created_at_answer	createdAtAnswer	La date de création de la réponse	date			
27							
28	id_category	idCategory	L'identifiant de la catégorie	int	11	AI	
29	wording_category	wordingCategory	Le nom de la catégorie	varchar	255		
30							
31	ref_user_concern	refUserConcern	L'utilisateur qui a accès à la question	int	11	Index	
32	ref_question_concern	refQuestionConcern	La référence de la question	int	11	Index	
33							
34	ref_user_like	refUserLike	La référence de l'utilisateur qui a like	int	11	Index	
35	ref_question_like	refQuestionLike	La référence de la question like	int	11	Index	
36	created_at_like	createdAtLike	La date du like	date			
37							
38	ref_sender_link	refSenderLink	L'envoyeur de la demande d'ami	int	11	Index	
39	ref_receiver_link	refReceiverLink	Le receveur de la demande d'ami	int	11	Index	
40	is_pending_link	isPendingLink	L'état de la demande	char	1		
41							

8. Modèle Entité-Association



9. Schéma Relationnel



10. Script de création

```
29
30 DROP TABLE IF EXISTS `answer`;
31 CREATE TABLE IF NOT EXISTS `answer` (
32   `id` int(11) NOT NULL AUTO_INCREMENT,
33   `ref_question` int(11) NOT NULL,
34   `ref_user` int(11) NOT NULL,
35   `content` varchar(255) CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci NOT NULL,
36   `created_at` datetime NOT NULL,
37   PRIMARY KEY (`id`),
38   KEY `IDX_DADD4A2530E08FC9` (`ref_question`),
39   KEY `IDX_DADD4A259AE097CE` (`ref_user`)
40 ) ENGINE=InnoDB AUTO_INCREMENT=1036 DEFAULT CHARSET=utf8mb4;
41
```

```
122
123 DROP TABLE IF EXISTS `category`;
124 CREATE TABLE IF NOT EXISTS `category` (
125   `id` int(11) NOT NULL AUTO_INCREMENT,
126   `wording` varchar(255) CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci NOT NULL,
127   PRIMARY KEY (`id`)
128 ) ENGINE=InnoDB AUTO_INCREMENT=221 DEFAULT CHARSET=utf8mb4;
129
```

```
146
147 DROP TABLE IF EXISTS `concern`;
148 CREATE TABLE IF NOT EXISTS `concern` (
149     `ref_user` int(11) NOT NULL,
150     `ref_question` int(11) NOT NULL,
151     PRIMARY KEY (`ref_user`,`ref_question`),
152     KEY `IDX_281EFBA89AE097CE` (`ref_user`),
153     KEY `IDX_281EFBA830E08FC9` (`ref_question`)
154 ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
155
156 -----
157
158 --
159 -- Structure de la table `like`
160 --
161
162 DROP TABLE IF EXISTS `like`;
163 CREATE TABLE IF NOT EXISTS `like` (
164     `ref_user` int(11) NOT NULL,
165     `ref_question` int(11) NOT NULL,
166     `created_at` datetime NOT NULL,
167     PRIMARY KEY (`ref_user`,`ref_question`),
168     KEY `IDX_AC6340B39AE097CE` (`ref_user`),
169     KEY `IDX_AC6340B330E08FC9` (`ref_question`)
170 ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
171
```

```
177
178 DROP TABLE IF EXISTS `link`;
179 CREATE TABLE IF NOT EXISTS `link` (
180     `ref_sender` int(11) NOT NULL,
181     `ref_receiver` int(11) NOT NULL,
182     `is_pending` tinyint(1) NOT NULL,
183     PRIMARY KEY (`ref_sender`,`ref_receiver`),
184     KEY `IDX_36AC99F12A192236` (`ref_sender`),
185     KEY `IDX_36AC99F1BBAF4A11` (`ref_receiver`)
186 ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
187
```

```
215
216 DROP TABLE IF EXISTS `question`;
217 CREATE TABLE IF NOT EXISTS `question` (
218   `id` int(11) NOT NULL AUTO_INCREMENT,
219   `ref_category` int(11) NOT NULL,
220   `ref_user` int(11) NOT NULL,
221   `title` varchar(255) CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci NOT NULL,
222   `created_at` datetime NOT NULL,
223   `visibility` varchar(255) CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci NOT NULL,
224   PRIMARY KEY (`id`),
225   KEY `IDX_B6F7494E805BDF46` (`ref_category`),
226   KEY `IDX_B6F7494E9AE097CE` (`ref_user`)
227 ) ENGINE=InnoDB AUTO_INCREMENT=1555 DEFAULT CHARSET=utf8mb4;
228
```

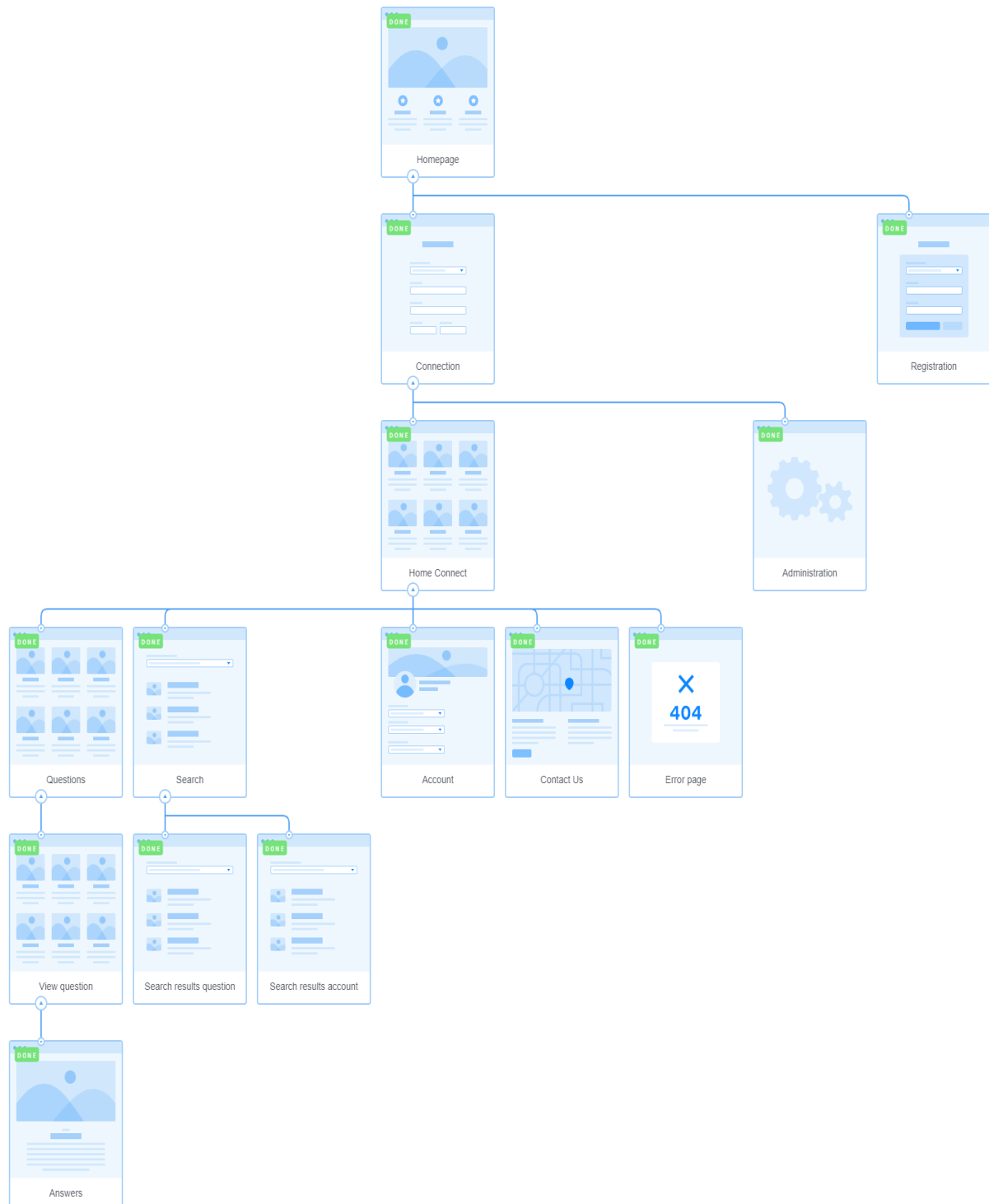
```
314
315 DROP TABLE IF EXISTS `user`;
316 CREATE TABLE IF NOT EXISTS `user` (
317   `id` int(11) NOT NULL AUTO_INCREMENT,
318   `username` varchar(255) CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci NOT NULL,
319   `email` varchar(180) CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci NOT NULL,
320   `roles` json NOT NULL,
321   `password` varchar(255) CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci NOT NULL,
322   `image` varchar(255) CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci DEFAULT NULL,
323   `gender` varchar(255) CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci NOT NULL,
324   `activation_token` varchar(255) CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci DEFAULT NULL,
325   `reset_token` varchar(255) CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci DEFAULT NULL,
326   `created_at` datetime NOT NULL,
327   `updated_at` datetime NOT NULL,
328   PRIMARY KEY (`id`),
329   UNIQUE KEY `UNIQ_8D93D649E7927C74` (`email`),
330   UNIQUE KEY `username` (`username`)
331 ) ENGINE=InnoDB AUTO_INCREMENT=645 DEFAULT CHARSET=utf8mb4;
332
```

IV. Les fonctionnalités

11. Backlog

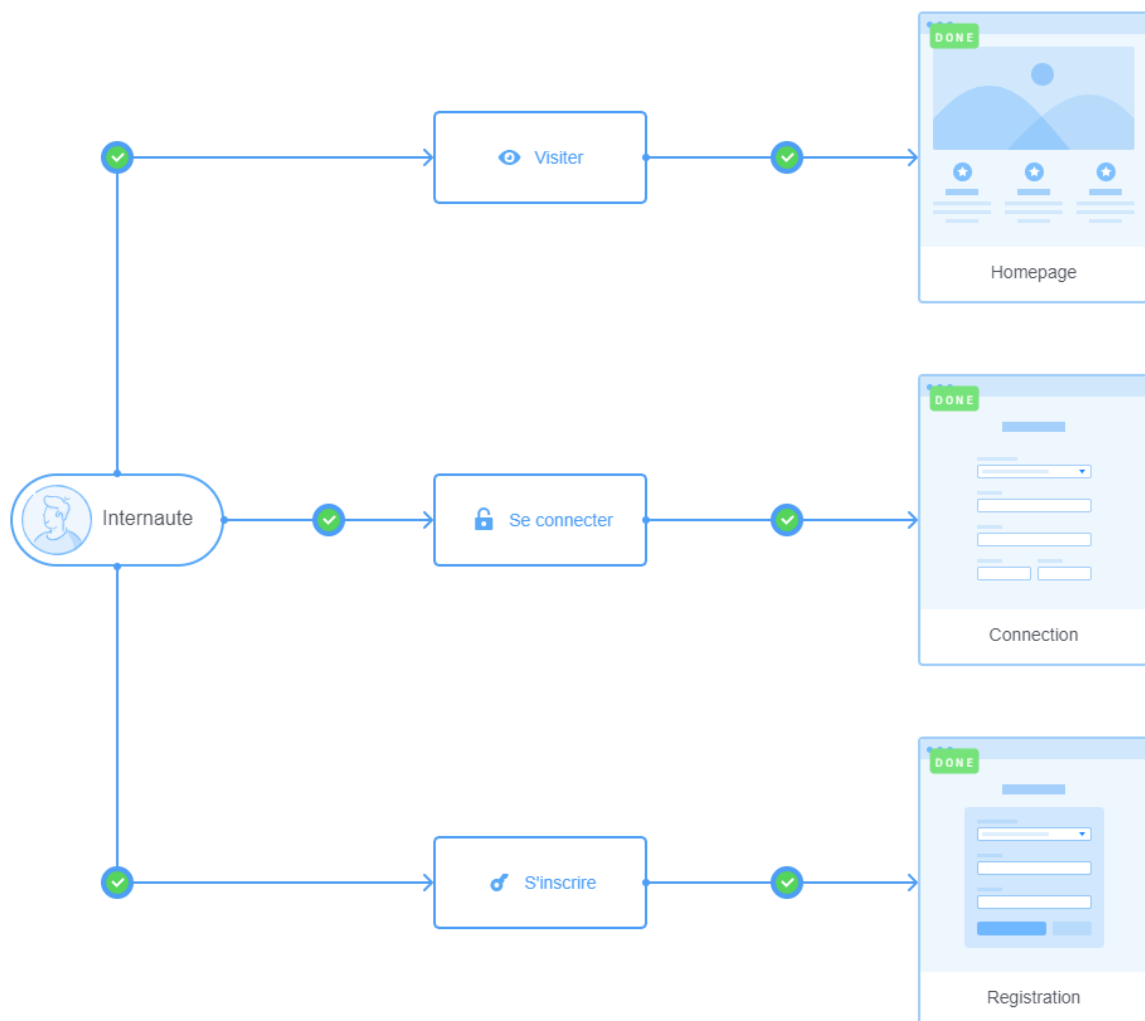
1	Fonctionnalité	Description
2		
3	flux de questions	affiche l'ensemble des questions
4	ajouter question	ajouter une question stockée en base de données
5	supprimer question	supprimer une question stockée en base de données
6	modification question	modifier une question stockée en base de données
7	rechercher question	rechercher une question par son titre
8	répondre question	répondre à une question
9	supprimer réponse	supprimer sa réponse à une question
10	question privé	question uniquement visible par ses amis
11	aimer question	aimer une question
12	je n'aime pas question	ne pas aimer une question
13	retirer aime question	retirer le j'aime d'une question
14	modification profil	modifier ses données personnel
15	inscription utilisateur	formulaire pour s'inscrire
16	connexion utilisateur	formulaire pour se connecter
17	vue question	voir une question en fonction de son id
18	vue utilisateur	voir un utilisateur en fonction de son id
19	ajouter ami	ajouter un utilisateur en ami
20	supprimer ami	supprimer un utilisateur en ami
21	voir ami	voir la liste de ses amis
22	espace administrateur	back end pour gérer le site
23	supprimer utilisateur	permet de supprimer définitivement un utilisateur quand t-on est admin
24	supprimer son compte	permet de supprimer son compte

12. Arborescence



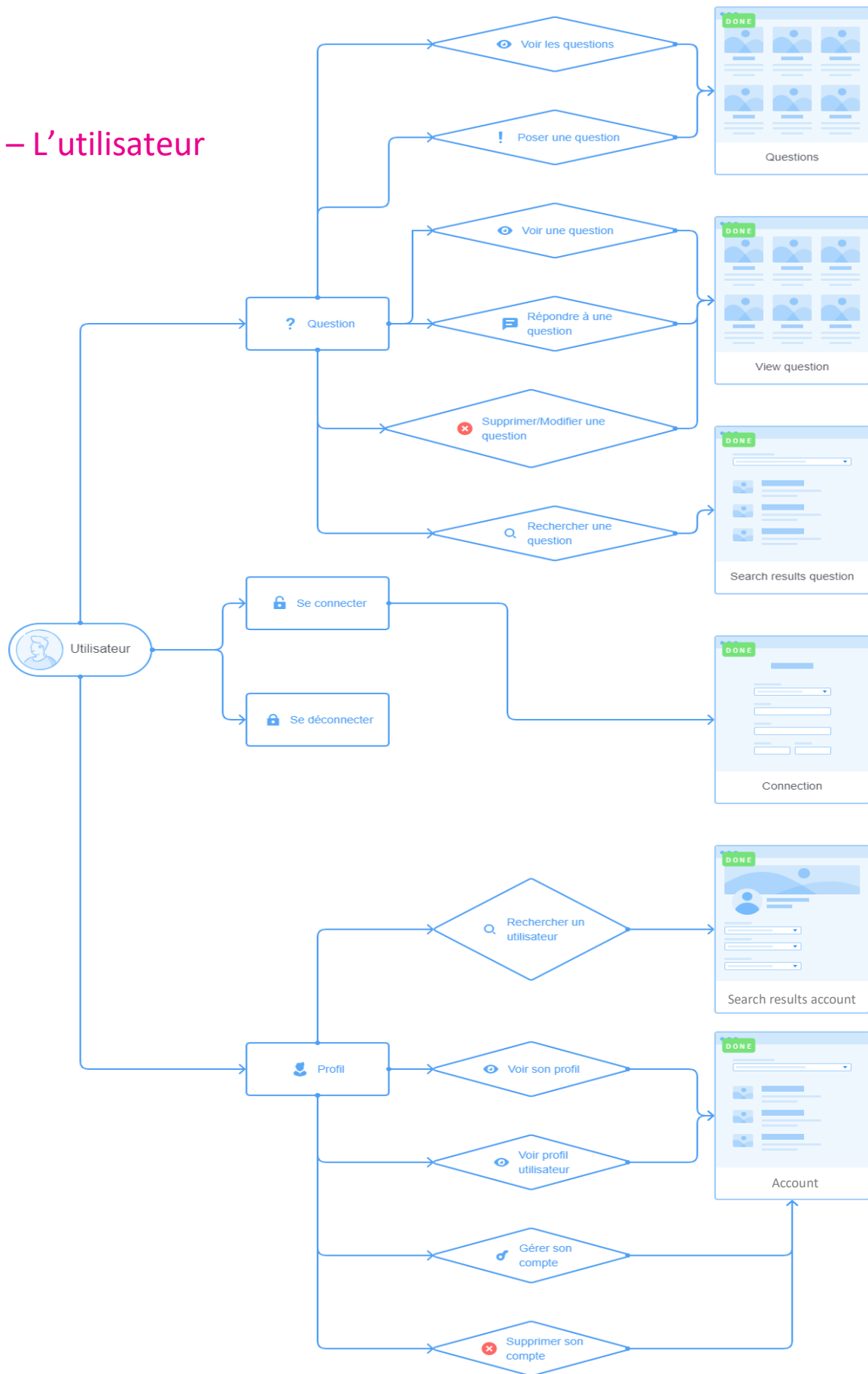
13. Cas d'utilisation

I – L'internaute



Acteur(s) Actrice(s)	Fonctionnalité	Page	Description
Internaute	Visiter	Home Page	L'internaute peut visiter la page d'accueil pour avoir un aperçu du réseau social.
Internaute	Se connecter	Connection	L'internaute peut se connecter après son inscription en renseignant ses identifiants pour accéder au site. Il est désormais un utilisateur.
Internaute	S'inscrire	Registration	L'internaute peut s'inscrire sur le site en renseignant ses informations personnelles. Il devient un utilisateur après cette étape.

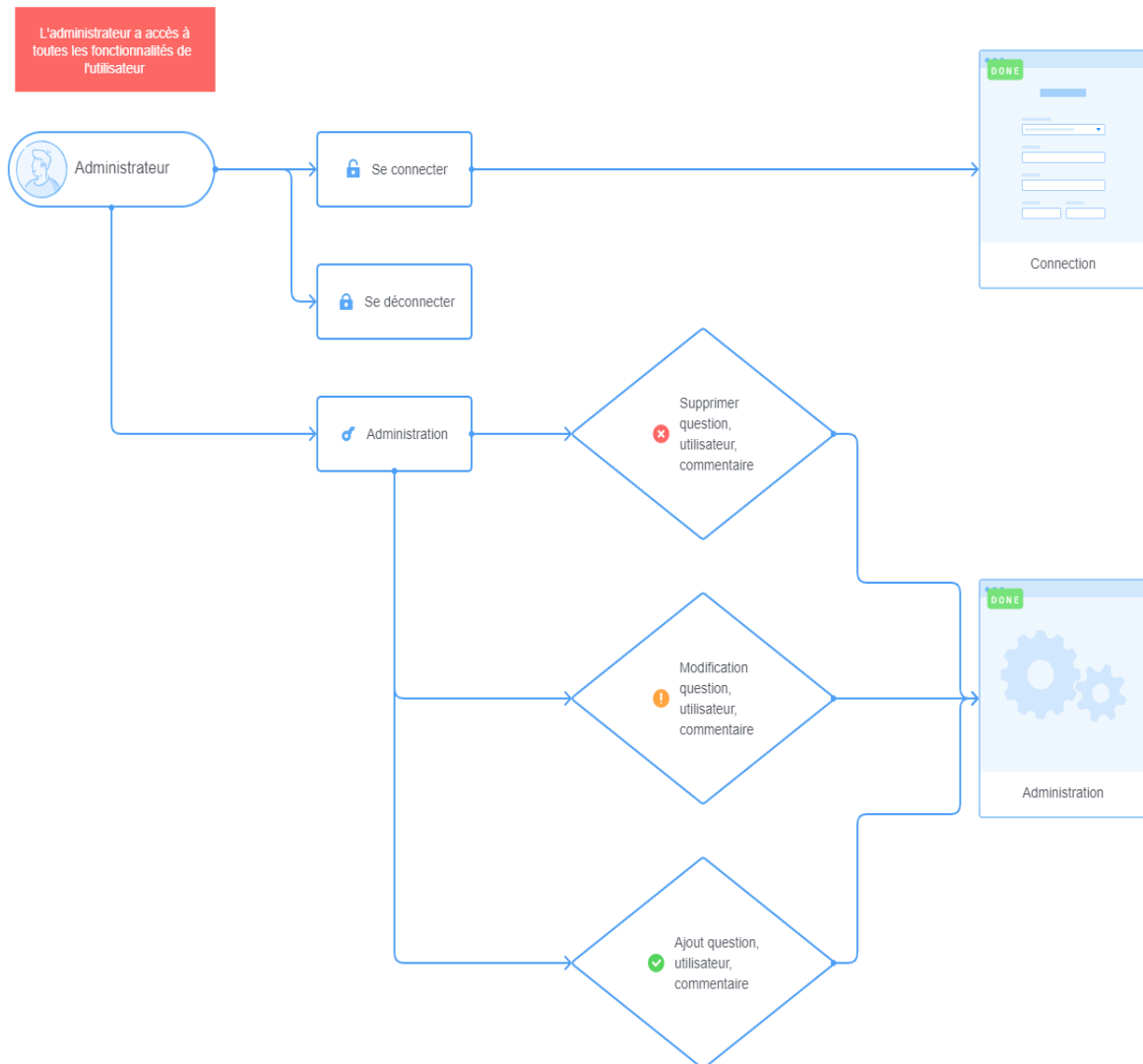
II – L'utilisateur



Acteur(s) Actrice(s)	Fonctionnalité	Page	Description
Utilisateur Utilisatrice	Voir les question	Questions	L'utilisateur/trice peut voir l'ensemble des questions et interagir avec elles.
Utilisateur Utilisatrice	Poser une question	Questions	L'utilisateur/trice peut poser une question visible par d'autres utilisateurs.
Utilisateur Utilisatrice	Voir une question	View Question	L'utilisateur/trice peut voir une question en particulier.
Utilisateur Utilisatrice	Répondre à une question	View Question	L'utilisateur/trice peut répondre à une question en laissant un commentaire/réponse.
Utilisateur Utilisatrice	Supprimer Modifier une question	View Question	L'utilisateur/trice peut supprimer ou modifier une question lui appartenant.
Utilisateur Utilisatrice	Rechercher une question	Search results question	L'utilisateur/trice peut rechercher une question en fonction de son titre.

Utilisateur Utilisatrice	Se connecter	Connection	L'utilisateur/trice peut se connecter en renseignant ses identifiants pour accéder au site.
Utilisateur Utilisatrice	Se déconnecter		L'utilisateur/trice peut se déconnecter en cliquant sur un bouton accessible depuis la barre de navigation.
Utilisateur Utilisatrice	Rechercher un utilisateur	Search results account	L'utilisateur/trice peut rechercher un autre utilisateur par son nom.
Utilisateur Utilisatrice	Voir son profil	Account	L'utilisateur/trice peut voir son profil.
Utilisateur Utilisatrice	Voir profil d'un utilisateur	Account	L'utilisateur/trice peut voir le profil d'un autre utilisateur ainsi que les questions qu'il a posées et commentaire qu'il a écrit.
Utilisateur Utilisatrice	Gérer son compte	Account	L'utilisateur/trice peut depuis son profil modifier ses informations personnelles.
Utilisateur Utilisatrice	Supprimer son compte	Account	L'utilisateur/trice peut supprimer son compte.

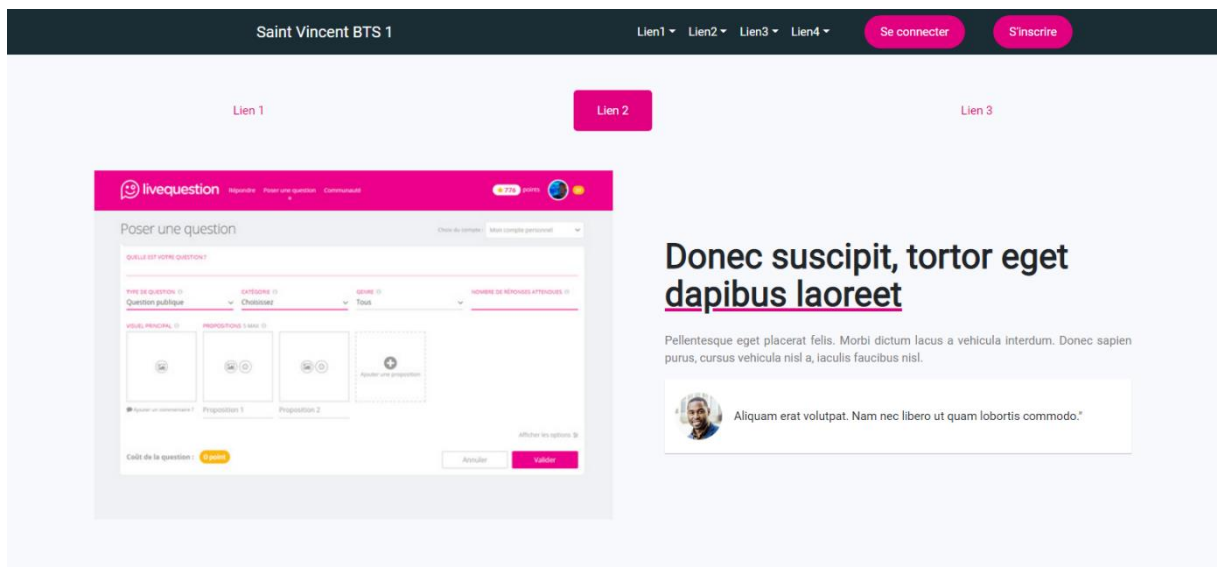
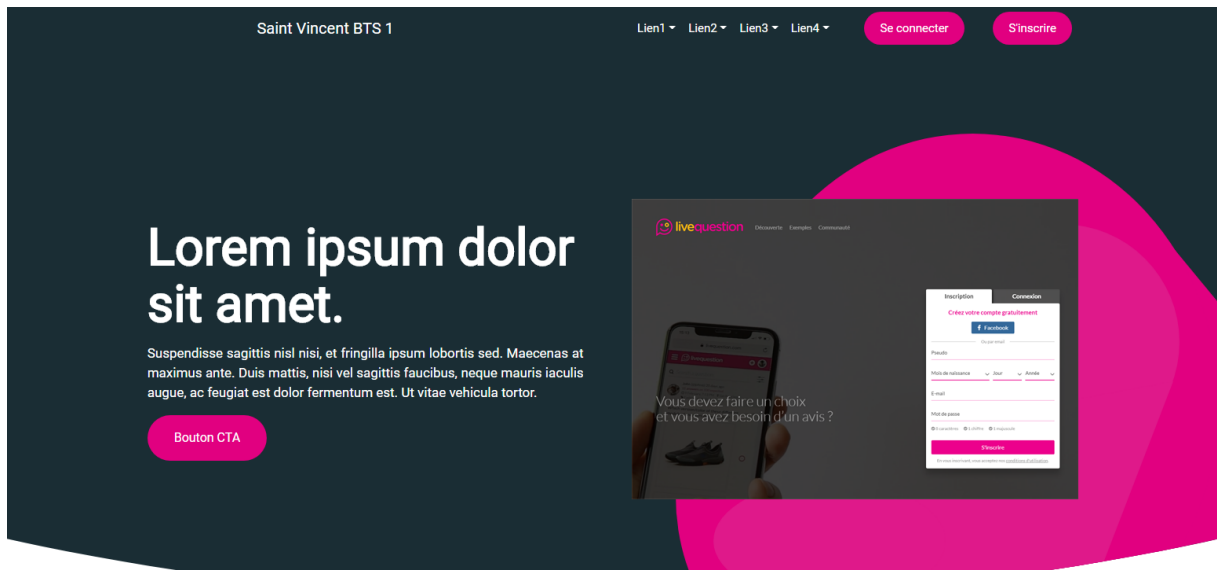
III – L'administrateur



Acteur(s) Actrice(s)	Fonctionnalité	Page	Description
Administrateur Administratrice	Se connecter	Connection	Administrateur/trice peut se connecter en renseignant ses identifiants pour accéder à l'espace d'administration.
Administrateur Administratrice	Se déconnecter		Administrateur/trice peut se déconnecter en cliquant sur un bouton accessible depuis la barre de navigation.
Administrateur Administratrice	Supprimer question, utilisateur, commentaire	Administration	Administrateur/trice peut supprimer une question, un utilisateur ou un commentaire
Administrateur Administratrice	Modifier question, utilisateur, commentaire	Administration	Administrateur/trice peut modifier une question, un utilisateur ou un commentaire
Administrateur Administratrice	Ajouter question, utilisateur, commentaire	Administration	Administrateur/trice peut ajouter une question, un utilisateur ou un commentaire

14. Maquettes

I – Page d'accueil (déconnecté)



FAQ

Sed laoreet efficitur quam, sit amet fermentum risus feugiat vel. Integer quis nisl nisi. Cras vitae hendrerit risus. Sed eget odio purus. Maecenas eu neque id arcu semper bibendum. Morbi vitae nunc blandit libero varius blandit at non arcu.

Can I upgrade later on? ▶

Can I port my data from another provider? ▶

Are my food photos stored forever in the cloud? ▶

Who foots the bill for that? ▶

What's the real cost ▶

Can my company request a custom plan? ▶

II - Page de connexion

Saint Vincent BTS 1 Lien1 ▾ Lien2 ▾ Lien3 ▾ Lien4 ▾ Se connecter S'inscrire

Se connecter

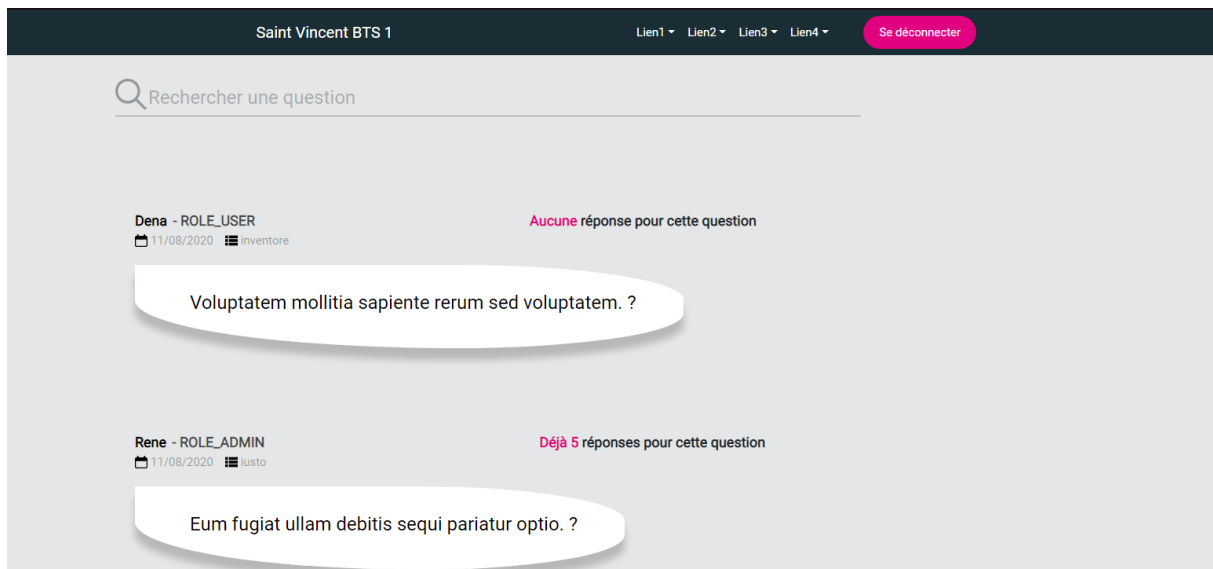
Connexion sécurisé

Email

Mot de passe

Envoyer

II - Page des questions



The screenshot displays the 'livequestion' web application interface. At the top, a dark blue header bar contains the text 'Saint Vincent BTS 1' on the left, a series of links 'Lien1', 'Lien2', 'Lien3', and 'Lien4' in the center, and a pink 'Se déconnecter' button on the right. Below the header is a light gray main area. At the top of this area is a search bar with a magnifying glass icon and the placeholder text 'Rechercher une question'. Below the search bar, two question entries are visible. The first entry is from 'Dena - ROLE_USER', dated '11/08/2020', with a status of 'inventore'. To the right of this entry, it says 'Aucune réponse pour cette question'. The question text is 'Voluptatem mollitia sapiente rerum sed voluptatem. ?'. The second entry is from 'Rene - ROLE_ADMIN', dated '11/08/2020', with a status of 'iusto'. To the right of this entry, it says 'Déjà 5 réponses pour cette question'. The question text is 'Eum fugiat ullam debitis sequi pariatur optio. ?'.

Saint Vincent BTS 1

Lien1 Lien2 Lien3 Lien4

Se déconnecter

Rechercher une question

Dena - ROLE_USER
11/08/2020 inventore

Aucune réponse pour cette question

Voluptatem mollitia sapiente rerum sed voluptatem. ?

Rene - ROLE_ADMIN
11/08/2020 justo

Déjà 5 réponses pour cette question

Eum fugiat ullam debitis sequi pariatur optio. ?

V. Dossier Technique

15. Fonctionnalité : Poser des questions

Objectif

La fonctionnalité « poser des questions » est accessible par l'ensemble des utilisateurs inscrit sur le site. Cette fonctionnalité accorde la possibilité de pouvoir poser une question visible par tous les utilisateurs. Les autres utilisateurs peuvent voir et interagir (aimer et y répondre) avec cette question. Il est possible de créer des questions privées uniquement visible par les amis de l'auteur de la question ou uniquement par les amis que l'auteur a choisis et des questions publiques visible par tous.

Extrait de code

```
$db = connexionBdd(); //connexion à la base de données

$select = $db->prepare( statement: "SELECT * FROM categorie"); //récupère les categories
$select->execute();
$categorie = $select->fetchAll(); //stock les données récupérées

$query = $db->prepare( statement: "
    SELECT F.*, P.Id_profil, P.Pseudo_profil, P.Genre_profil, R.Libelle_role
    FROM friend F
    LEFT JOIN profil P ON F.username_1 = :user_c
    AND F.username_2 = P.Id_profil OR F.username_1 = P.Id_profil
    AND F.username_2 = :username
    LEFT JOIN role R ON P.`Id_role` = R.Id_role
    WHERE F.is_pending = 0"); //récupère les amis

$query->execute([
    "user_c" => $_SESSION['id'], //assigne la valeur de user_c avec l'id de l'utilisateur connecté
]);
$data = $query->fetchAll( fetch_style: PDO::FETCH_ASSOC); //stock les données récupérer
```

```
if (!empty($_POST['type_question'])) { //vérifie si $_POST['type_question'] n'est pas vide
    //insère la question en base
    $query = $db->prepare( statement: "
        INSERT INTO question (Titre_question, Date_creation_question, Id_profil, Id_categorie, type_question)
        VALUES (:Titre_question, :Date_creation_question, :Id_profil, :Id_categorie, :type_question)");

    date_default_timezone_set( timezone_identifer: 'Europe/Paris'); //définie la timezone
    $dateCreationQuestion = date( format: 'Y-m-d'); //format de la date

    //assigne les valeurs aux paramètres de la requête
    $query->bindParam( parameter: 'Titre_question', &variable: $_POST['Titre_question']);
    $query->bindParam( parameter: 'Date_creation_question', &variable: $dateCreationQuestion);
    $query->bindParam( parameter: 'Id_profil', &variable: $_SESSION['id']);
    $query->bindParam( parameter: 'Id_categorie', &variable: $_POST['type_question']);
    $query->bindParam( parameter: 'type_question', &variable: $_POST['type_question']);

    $query->execute();
    $query->closeCursor();
}
```

```
<h2>Posez votre question !</h2>
<form method="POST" action="#" class="col-4">
    <div class="form-group">
        <label for="Titre_question">Posez votre question : </label>
        <input class="form-control" id="Titre_question" name="Titre_question" maxlength="255">
        <small class="form-text text-muted">Maximum 255 caractères !</small>
    </div>
    <div class="form-group">
        <label for="id_categorie">Catégorie de votre question : </label>
        <select class="form-control" id="id_categorie" name="id_categorie" size="1">
            <?php
                foreach ($categories as $categorie) {
                    echo '<option value= "'. $categorie['Id_categorie']. ">'. $categorie['Libelle_categorie']. ' </option>';
                }
            ?>
        </select>
    </div>
    <div class="form-group">
        <label for="type_question">Question privé/publique ? </label>
        <select class="form-control" id="type_question" name="type_question" size="1">
            <option value="0">Publique</option>
            <option value="1">Privé</option>
        </select>
    </div>
    <label>Choisissez quels amis peuvent voir votre question ( uniquement pour les questions privées) :</label>
    <?php
        foreach ($data as $friend) {
            echo '
                <div class="form-check">
                    <input class="form-check-input member_check" name="member_check[]" type="checkbox" value="'. $friend['Id_profil']. ">
                    <label class="form-check-label" for="member_check">'. $friend['Pseudo_profil']. '</label>
                </div>';
            }
        ?>
    <button class="bouton-envoi" type="submit">Ajouter !</button>
</form>
```

16. Fonctionnalité : Ajouter en ami

Objectif

La fonctionnalité « ajouter en ami » est accessible par l'ensemble des utilisateurs inscrit sur le site. Cette fonctionnalité accorde la possibilité de pouvoir ajouter en ami d'autres utilisateurs. Les autres utilisateurs peuvent accepter ou refuser la demande en ami. Il est possible pour un utilisateur d'annuler la demande en ami envoyer.

Extrait de code

```
function getMember() //: array|string //type de retour union type tableau ou string uniquement possible avec php8
{
    $error = ''; //déclaration de la variable error

    if (!empty($_POST['add-friend'])) { //vérifie que la condition soit différente de vide
        //récupération du profil correspondant au paramètre
        $query = connexionBdd()->prepare( "
            SELECT P.*, R.Libelle_role
            FROM profil P
            LEFT JOIN role R
            ON P.`#Id_role` = R.Id_role
            LEFT JOIN friend f
            ON F.username_1 = :username
            AND F.username_2 = P.Id_profil
            OR F.username_1 = P.Id_profil
            AND F.username_2 = :username
            WHERE Pseudo_profil
            LIKE CONCAT('%', :search, '%')
            AND F.username_1 IS NULL
            AND P.Id_profil != :username");

        $query->execute([
            "search" => htmlspecialchars(trim($_POST['add-friend'])),
            "username" => $_SESSION['id'],
        ]);

        if ($query->rowCount() > 0) { //si le résultat de la requête est supérieur à 0
            $data = $query->fetchAll( fetch_style: PDO::FETCH_ASSOC); //on stock les données
            $query->closeCursor(); //ferme le curseur, permettant à la requête d'être de nouveau exécutée

            return $data; //on retourne les données
        }
    }
}
```

```

        elseif ($query->rowCount()===0) { //si aucun résultat
            //on assigne le message d'erreur
            $error = "<div>
                <img src='../public/images/svg/find.svg' alt='svg find'>
            </div>
            <p>Il n'y a aucun résultat</p>";
        }
    }

    return $error; //on retourne le message d'erreur
}

function getApprovedFriend(): ?array //type de retour tableau
{
    //requête qui récupère les demandes d'amis
    $query = connexionBdd()->prepare( statement: "
        SELECT F.*, P.Id_profil, P.Pseudo_profil, P.Genre_profil, R.Libelle_role
        FROM friend F
        LEFT JOIN profil P
        ON F.username_1 = :username
        AND F.username_2 = P.Id_profil
        OR F.username_1 = P.Id_profil
        AND F.username_2 = :username
        LEFT JOIN role R
        ON P.`Id_role` = R.Id_role");

    $query->execute([
        "username" => $_SESSION['id'],
    ]);

    $data = $query->fetchAll( fetch_style: PDO::FETCH_ASSOC); //on stock le résultat
    $query->closeCursor(); //ferme le curseur, permettant à la requête d'être de nouveau exécutée

    return $data; //on retourne les données
}

```

```

function checkDuplicate(): ?array //type de retour tableau
{
    //requête qui récupère toutes les demandes d'amis en fonction des utilisateurs passé en paramètre
    $query = connexionBdd()->prepare( statement: "
        SELECT F.*
        FROM friend F
        WHERE (F.username_1 = :username_1_check
        AND F.username_2 = :username_2_check)
        OR (F.username_1 = :username_2_check
        AND F.username_2 = :username_1_check)");

    $query->execute([
        "username_1_check" => $_SESSION['id'],
        "username_2_check" => $_POST['id'],
    ]);

    $data = $query->fetch(); //on stock le résultat
    $query->closeCursor(); //ferme le curseur, permettant à la requête d'être de nouveau exécutée

    return $data; //on retourne les données
}

```

```
$_SESSION['message']='';

if (isset($_POST['add'])) { //vérifie que la condition existe et a une valeur assigné
    if (checkDuplicate() === false) { //vérifie que la condition soit strictement égal à false
        //ajoute la demande en base
        $query = connexionBdd()->prepare( statement: "
            INSERT INTO friend(username_1, username_2, is_pending)
            VALUES (:username_1, :username_2, :is_pending)");
        $query->execute([
            "username_1" => $_SESSION['id'],
            "username_2" => $_POST['id'],
            "is_pending" => 1,
        ]);
        $query->closeCursor(); //ferme le curseur, permettant à la requête d'être de nouveau exécutée

        header( string: 'Location:add-friend.php'); //redirection vers la page cible
        exit(); //termine le script courant
    } else { //si la condition est différente de false
        $_SESSION['message'] = '<div><p>Déjà en ami</p></div>'; //assigne le message d'erreur

        header( string: 'Location:add-friend.php'); //redirection vers la page cible
        exit(); //termine le script courant
    }
} elseif (isset($_POST['delete'])) { //vérifie que la condition existe et a une valeur assigné
    //supprime la demande d'ami en base
    $query = connexionBdd()->prepare( statement: "
        DELETE FROM friend
        WHERE username_1 = :username_1
        AND username_2 = :username_2");
    $query->execute([
        "username_1" => $_POST['id2'],
        "username_2" => $_POST['id3'],
    ]);
}
```

```
$query->closeCursor(); //ferme le curseur, permettant à la requête d'être de nouveau exécutée

header( string: 'Location:add-friend.php'); //redirection vers la page cible
exit(); //termine le script courant
} elseif (isset($_POST['accept'])) { //vérifie que la condition existe et a une valeur assigné
    //modifier la demande d'ami en base pour montrer qu'elle est acceptée
    $query = connexionBdd()->prepare( statement: "
        UPDATE friend
        SET is_pending = 0
        WHERE username_1 = :username_1
        AND username_2 = :username_2");

    $query->execute([
        "username_1" => $_POST['id4'],
        "username_2" => $_POST['id5'],
    ]);
    $query->closeCursor(); //ferme le curseur, permettant à la requête d'être de nouveau exécutée

    header( string: 'Location:add-friend.php'); //redirection vers la page cible
    exit(); //termine le script courant
} elseif (isset($_POST['cancel'])) { //vérifie que la condition existe et a une valeur assigné
    //supprime la demande d'ami en base
    $query = connexionBdd()->prepare( statement: "
        DELETE FROM friend
        WHERE username_1 = :username_1
        AND username_2 = :username_2
        AND is_pending = 1");

    $query->execute([
        "username_1" => $_SESSION['id'],
        "username_2" => $_POST['id7'],
    ]);
    $query->closeCursor(); //ferme le curseur, permettant à la requête d'être de nouveau exécutée

    header( string: 'Location:add-friend.php'); //redirection vers la page cible
    exit(); //termine le script courant
```

17. Fonctionnalité : Rechercher une question

Objectif

La fonctionnalité « rechercher une question » est accessible par l'ensemble des utilisateurs inscrit sur le site. Cette fonctionnalité accorde la possibilité de pouvoir rechercher une question par rapport à son titre. Il y a aussi la possibilité de rechercher un ensemble de question comportant les caractères rechercher par l'utilisateur. Il se peut donc qu'il y est plusieurs résultats. Dans la seconde version du site en Symfony 5 il sera possible de filtrer les questions par catégorie, langue, popularité, nombre de likes et nombre de commentaires.

Extrait de code

```
function searchQuestion() //: array|string //type de retour union type tableau ou string uniquement possible avec php8
{
    $error = ''; //déclare la variable d'erreur

    if (!empty($_POST['search_pcl'])) { //si la variable n'est pas vide
        //récupère la question, le profil de l'auteur, sa catégorie et son rôle en fonction de la recherche de l'utilisateur
        $query = connexionBdd()->prepare('statement: '
            SELECT *
            FROM question Q
            LEFT JOIN restriction_question RQ ON RQ.`#id_question` = Q.Id_question
            LEFT JOIN categorie C ON Q.`#Id_categorie` = C.Id_categorie
            LEFT JOIN profil P ON Q.`#Id_profil` = P.Id_profil
            LEFT JOIN role R ON P.`#Id_role` = R.Id_role
            WHERE Q.Titre_question LIKE CONCAT("%", :search, "%")
            AND Q.type_question = 0 OR Q.type_question = 1 AND Q.`#Id_profil` = :id
            UNION
            SELECT *
            FROM question Q
            LEFT JOIN restriction_question RQ ON RQ.`#id_question` = Q.Id_question
            LEFT JOIN categorie C ON Q.`#Id_categorie` = C.Id_categorie
            LEFT JOIN profil P ON Q.`#Id_profil` = P.Id_profil
            LEFT JOIN role R ON P.`#Id_role` = R.Id_role
            WHERE Q.Titre_question LIKE CONCAT("%", :search, "%")
            AND RQ.`#id_profil` = :id
            ORDER BY Date_creation_question DESC, Id_profil, Titre_question');

        $query->execute([
            "search" => htmlspecialchars(trim($_POST['search_pcl'])),
            "id" => $_SESSION['id'],
        ]);
    }
}
```



```

        if($query->rowCount()>0){ //si le résultat de la requête est supérieur à 0
            $data = $query->fetchAll( fetch_style: PDO::FETCH_ASSOC); //on stock les données
            $query->closeCursor(); //ferme le curseur, permettant à la requête d'être de nouveau exécutée

            return $data; //on retourne les données
        }
        elseif($query->rowCount()===0) { //si aucun résultat
            $error = "Il n'y a aucun résultat"; //on assigne le message d'erreur
        }
    } else { //si la variable est vide

        return getQuestion(); //exécute la fonction getQuestion()
    }

    return $error; //on retourne le message d'erreur
}

```

```

<div class="search_pcl_question">
    <div class="question">
        <?php
            require_once("search-rechercher.php"); //inclut et exécute le fichier spécifié en argument une seul fois
            $questions = searchQuestion(); //assigne la variable avec le résultat de la requête effectué dans search-rechercher
            if (is_array($questions) === true) { //si il y a des questions
                foreach ($questions as $question){ //on parcourt et on les affiche
                    ?>
                    <div class="question-group">
                        <div class="question-head">
                            <div class="question-head-top">
                                <p>
                                    <a href="../../../user-view/user-view.php?id=<?= $question['Id_profil'];?>">
                                        <?= $question['Pseudo_profil'];?>
                                        <span class="role-utilisateur"> -
                                            <?= $question['Libelle_role'];?>
                                        </span>
                                    </a>
                                </p>
                                <p>
                                    <span class="text-rose">Déjà <?php
                                        require_once("search-reponse.php"); //inclut et exécute le fichier spécifié en argument une seul fois
                                        $reponses = getAnswer(); //assigne la variable avec le résultat de la requête effectué dans search-reponse
                                        $nbrReponse = 0; //assigne 0 à la variable

                                        foreach ($reponses as $reponse) { //on parcourt reponses
                                            //si il y a une réponse pour la question actuel
                                            if ($reponse['#Id_question'] === $question['Id_question']) {
                                                $nbrReponse+= 1; //on ajoute un à notre variable
                                            }
                                        }
                                    </span>
                                </p>
                            </div>
                        </div>
                    </div>
                }
            }
        </?php>
    </div>

```

```

        echo $nbrReponse; //affiche le nbr de réponse

        if ($nbrReponse === 1) { //si une seul réponse
            echo " réponse"; //réponse sans s
        } else { //autrement
            echo " réponses"; // réponses avec un s
        } ?>
    </span> pour cette question</p>
    <div class="clear"></div>
</div>
<div class="question-under-head">
    
    <p><?= $question['Date_creation_question'];?></p>
    
    <p><?= $question['Libelle_categorie'];?></p>
    <div class="clear"></div>
</div>
</div>
<div class="clear"></div>
<div class="question-body">
    <p>
        <a href="../../../view_question/voir_une_question.php?id=<?= $question['Id_question'];?>">
            <?= $question['Titre_question'];?>
        </a>
    </p>
    <div class="clear"></div>
</div>
<div class="clear"></div>
</div>
<div class="clear"></div>
<?php }

```

```

<?php }
    } else { ?>
        <div class="question-error">
            <div>
                
            </div>
            <p><?= $questions; //affiche l'erreur ?></p>
        </div>
    <?php }

```

VI. Veille contextualisée

Lors de ce projet j'ai effectué de nombreuses veilles. Pendant la première version du site je me suis formé à Bootstrap 4 et JQuery. Je me suis également amélioré en PHP, en SQL, et en modélisation. J'ai pris connaissance des conventions existantes en Css qui sont OOCSS et BEM et je les ai appliqués dans le projet. J'ai approfondi mes connaissances sur Git et GitHub.

Pour la refonte du projet en Symfony 5 je me suis auto-formé au framework Symfony à sa toute dernière version. J'ai également amélioré mes connaissances en JavaScript et en Ajax. La refonte de ce projet m'a permis d'approfondir mes connaissances en PHP ainsi que de suivre les conventions PSR. J'ai énormément appris sur l'ensemble des META en HTML et de leur importance pour la SERP, SEO. Je me suis intéressé aux stratégies digitale et numérique et pris connaissance du Search Engine Advertising, Search Engine Marketing, Search Engine Optimization, Search eXperience Optimization, Search Engine Results Page, Social Media Optimization, ShowRooming, Geofencing, Store To web, Spam et Spamdexing. Je me suis formé à CSS4, utilisé les variables en CSS et utiliser les animations, transitions, transformations, clip-path, cubic-bezier, flex avancé et grid avancé.

VII. Conduite de projet

18. Répartition des tâches

Le tableau concerne la répartition des tâches pour la première version du réseau social. Concernant la seconde version du réseau social seul Hugo Monteiro a fait les tâches.

Développeur	Tâches	Stack	Durée
Hugo	Inscription	PHP7 – HTML 5 – CSS4	2 jours
Marius	Poser une question	PHP7 – HTML 5 – CSS4	4 jours
Elisa	Profil utilisateur	PHP7 – HTML 5 – CSS4	7 jours
Marius	Vue d'une question	PHP7 – HTML 5 – CSS4	2 jours
Hugo	Menu Principal	PHP7 – HTML 5 – CSS4	3 jours
Elisa	Vue d'un utilisateur	PHP7 – HTML 5 – CSS4	4 jours

Marius	Administration	PHP7 – HTML 5 – CSS4	2 jours
Elisa	Vue d'un utilisateur	PHP7 – HTML 5 – CSS4	3 jours
Hugo	Page d'accueil connecté	PHP7 – HTML 5 – CSS4	2 jours
Hugo	Page d'accueil déconnecté (intégration maquette)	PHP7 – HTML 5 – CSS4	3 jours
Hugo	Header / Footer	PHP7 – HTML 5 – CSS4	1 jours
Hugo	Réorganisation du CSS	CSS4	1 jours
Hugo	Sécurité	PHP7	2 jours
Hugo	Responsive	CSS4	2 jours
Elisa	Désinscrire	PHP7 – HTML 5 – CSS4	3 jours
Hugo	Ajout like	PHP7 – HTML 5 – CSS4	1 jours

Elisa et Marius	Ajout avatar	HTML 5 – CSS4	2 jours
-----------------	--------------	---------------	---------

19. Compte Rendu

Ce que je retiens de ce projet est qu'il m'a permis d'approfondir considérablement mes connaissances dans tous les langages que j'ai pu utiliser lors de ce projet. La découverte des conventions et de leur importance. J'ai aussi pu découvrir les fondamentaux de gestion, de management de projet, stratégie et organisation pour mener efficacement la mission. Ce projet m'a fait découvrir comment se dérouler le processus de production d'un projet professionnel.