

SR03 - Devoir n°2

**Développement d'une application
de salon de discussion**

Partie I - Contexte

A. Le projet

Le projet de chat présenté dans ce rapport a été réalisé en avril 2021 dans le cadre de l'UV SR03, Architecture des applications internet, à l'Université Technologique de Compiègne, par Dylan Cornélie et Pauline Breteau.

Ce projet a pour but de développer une application de salons de discussion pour organiser une discussion entre différents participants.

Dans le cadre de l'UV SR03, le langage Java est le langage utilisé pour la partie développement, et le Javascript et le HTML pour la partie web. Nous avons utilisé Eclipse comme IDE. Nous avons utilisé GitLab pour gérer l'intégration continue du projet.

B. Les concepts

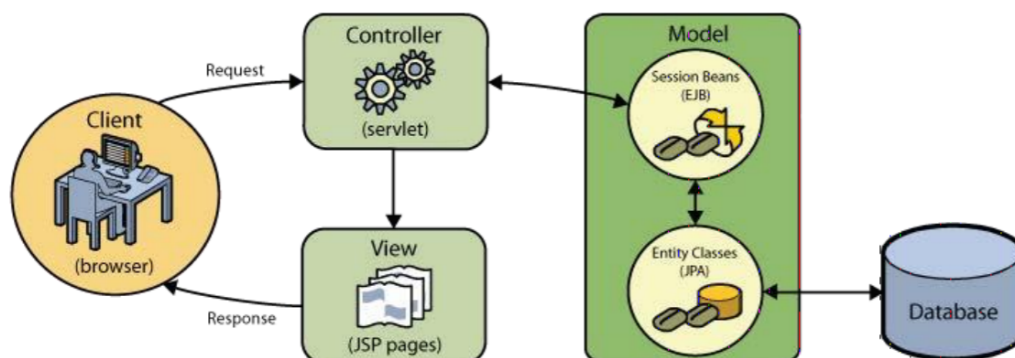
Ce projet nous fait manipuler différents concepts que lesquelles nous allons revenir brièvement :

Premier partie : application web de chat

1. Architecture MVC

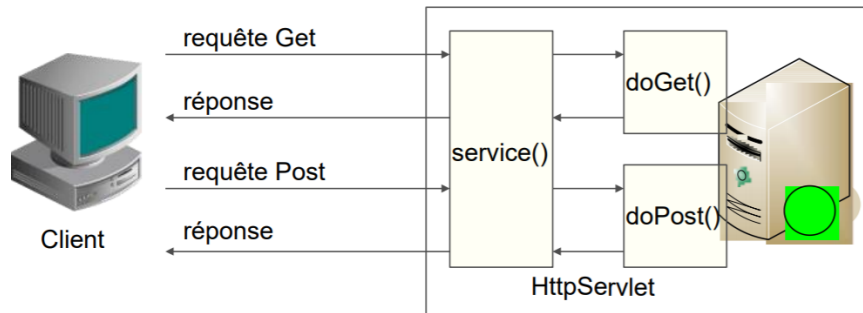
L'architecture MVC pour Modèle / Vue / Contrôleur permet de bien organiser son code source en séparant la logique du code en trois parties distinctes :

- modèle : qui permet d'aller récupérer les données dans la base de données. Dans notre application, on y trouve les requêtes SQL. C'est donc dans cette partie que nous utilisons le patron de conception ActiveRecord.
- vue : qui permet de s'occuper notamment de l'interface en récupérant les variables en les affichant. Dans notre application, ce sont majoritairement des pages HTML et des JSP. Ces dernières sont utilisées pour séparer le traitement de la requête et la génération du code HTML.
- contrôleur : qui gère la partie logique du code. Il reçoit la requête du client et échange avec le modèle et la vue des informations.



d'après le cours de SR03

2. La classe *HTTPServlet*



d'après le cours de SR03

Les servlets http sont utiles pour générer des pages HTML dynamique ou pour mettre à jour une base de données par exemple. On les trouve dans ce package `javax.servlet.http` qui définit plusieurs interfaces et méthodes.

La classe `HttpServlet` hérite de `GenericServlet` et permet de définir une servlet utilisant le protocole http. Elle permet de redéfinir toutes les méthodes pour développer des servlets avec le protocole HTTP.

La requête du client est encapsulée dans un objet qui implémente l'interface `HttpServletRequest` et la réponse dans `HttpServletResponse`.

3. Le design pattern *ActiveRecord*

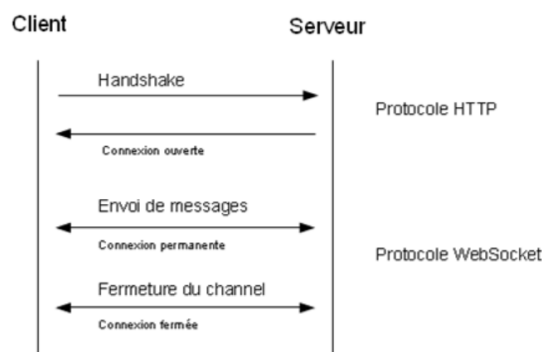
Le design pattern *ActiveRecord* permet la manipulation de données en liant les données d'une BDD à notre application. Les attributs d'une table ou d'une vue sont encapsulés dans une classe. Chaque objet instancié par la classe est lié à un tuple de la base.

Deuxième partie : serveur de chat

1. Le protocole WebSocket

D'après le cours, c'est " un standard du Web désignant un protocole réseau de la couche application et une interface de programmation du World Wide Web visant à créer des canaux de communication full-duplex par-dessus une connexion TCP pour les navigateurs web." Ainsi, une WebSocket est utilisée pour l'échange de données entre un client et un serveur de manière asynchrone et bidirectionnelle. Elle repose sur un protocole réseau composé de 2 phases :

- une phase de requête / réponse pour établir une connexion entre un client et un serveur.
- une phase d'échange de données.



© Jean-Michel DOUDOUX.

C. Objectifs fixés

Notre objectif était de parvenir à la réalisation fonctionnelle de l'application avec un code optimisé et de réaliser les différentes fonctionnalités de l'application :

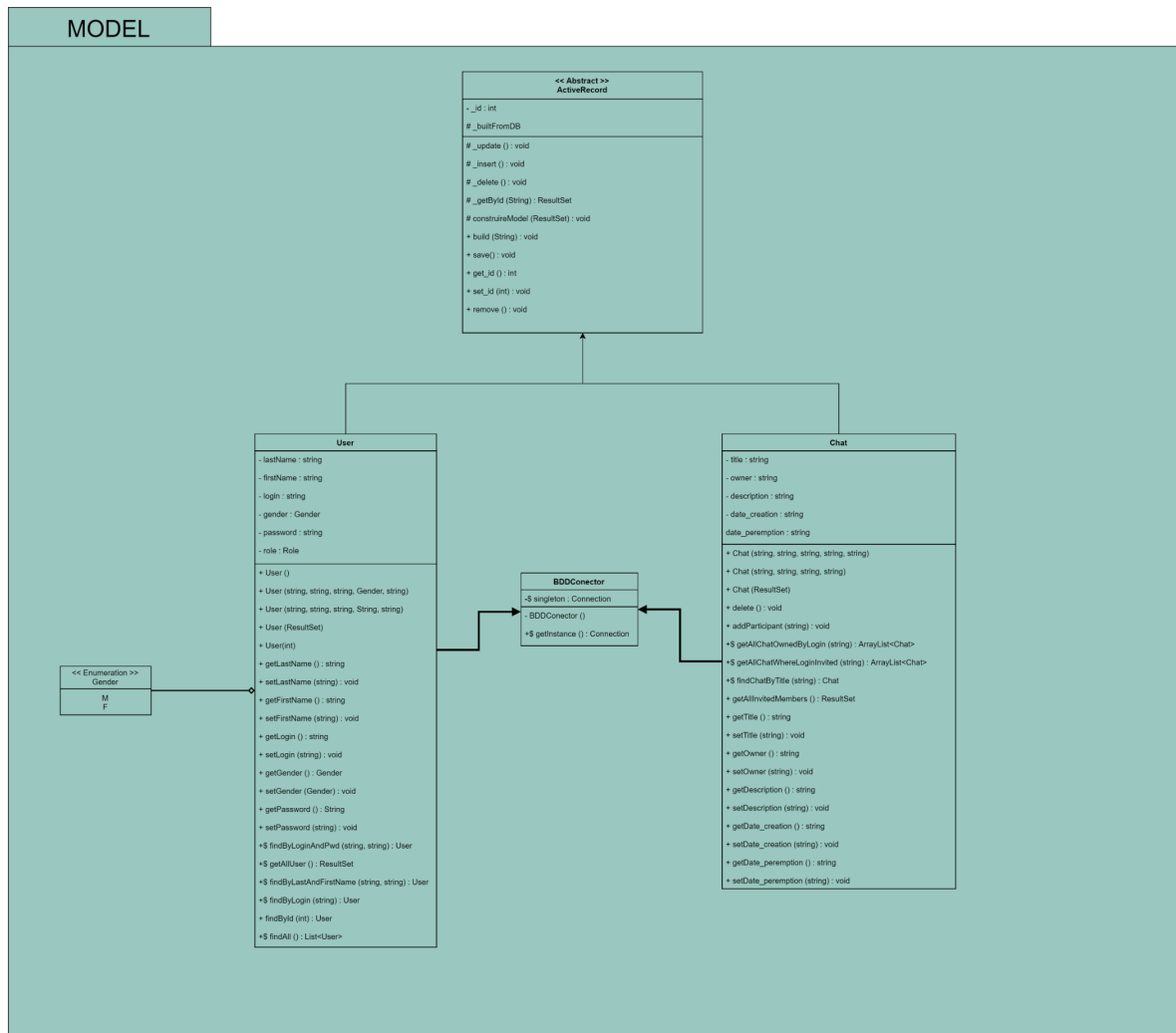
- s'inscrire sur l'application ;
- créer un chat et inviter des personnes à le rejoindre ;
- discuter sur un fil de discussion.

Partie II - Conception et développement

Le livrable est architecturé selon le modèle MVC :

- Modèle : classes métiers avec une couche accès aux données (Active Record)
- Vue : avec des pages HTMLs/JSPs
- Contrôleur : Servlets

A. Modèle



a. Structure de la base de données

La table qui stocke les utilisateurs :

```

CREATE TABLE `users` (
  `id` INT NOT NULL UNIQUE AUTO_INCREMENT,
  `fname` VARCHAR(50) NOT NULL,
  `lname` VARCHAR(50) NOT NULL,
  `login` VARCHAR(50) NOT NULL UNIQUE,
  `gender` enum('M','F') NOT NULL,

```

```

        `pwd` varchar(50) NOT NULL,
        PRIMARY KEY (`id`)
    )

```

La table qui stocke les chats :

```

CREATE TABLE `chat` (
    `id` INT NOT NULL UNIQUE AUTO_INCREMENT,
    `title` VARCHAR(50) NOT NULL UNIQUE,
    `owner` VARCHAR(50) NOT NULL,
    `description` VARCHAR(250) NOT NULL,
    `date_creation` DATETIME NOT NULL DEFAULT
CURRENT_TIMESTAMP,
    `date_peremption` DATE NOT NULL,
    PRIMARY KEY (`id`)
)

```

La table qui représentent les utilisateurs invités à un chat :

```

CREATE TABLE `participant_chat` (
    `login` VARCHAR(50) NOT NULL,
    `chat` VARCHAR(50) NOT NULL,
    PRIMARY KEY(`login`,`chat`),
    FOREIGN KEY(`login`) REFERENCES users(login),
    FOREIGN KEY(`chat`) REFERENCES chat(title)
)

```

b. BDDConector

Cette classe permet d'établir une connexion à une base de données. Afin que la classe puisse obtenir les informations nécessaires pour se connecter à une base de données, un fichier `.properties` avec : le nom du driver de la base de données, l'url, le nom de l'utilisateur et son mot de passe doit être renseigné au préalable.

La méthode `getInstance()` de cette classe renvoie un objet `Connection` qui représente alors la connexion à la base de données, cet objet nous permet de réaliser des requêtes sur la base de données. La méthode s'appuie sur les informations contenues dans le fichier `.properties` pour établir la connexion à la base de données.

c. ActiveRecord

Cette classe abstraite nous permet de mettre en place le design pattern « Active Record », elle fournit une interface permettant la manipulation de données venant de notre base de données.

Les méthodes virtuelles `_update()` , `_insert()` , `_delete()` permettent de réaliser les opérations correspondantes dans la base de données. Elles seront donc définies dans les classes filles.

La méthode virtuelle `_getId()` , permet de trouver un élément de la base de données grâce à son identifiant, cette méthode est elle aussi définie dans les classes filles.

La méthode virtuelle `construireModel()` , permet de construire notre modèle à partir du résultat d'une requête à notre base de donnée, cette méthode sera définie dans les classes filles.

L'attribut `_builtFromDB` permet de savoir si l'objet est construit avec les informations contenues dans la base de données.

La méthode `build()` , est une méthode public permettant de créer un objet de la classe fille en renseignant son identifiant.

La méthode `save()` permet de sauvegarder le modèle dans la base de données.

La méthode `remove()` permet de supprimer le modèle de la base de données

L'attribut `_id` peut être accédé et modifié via ses getter et setter.

d. User

La classe User représente un utilisateur du chat, elle est composée de 6 attributs `lastName`, `firstName`, `login`, `gender` et `password` représentant respectivement le nom de la personne, son prénom, son login, son genre et son mot de passe.

La classe User inclut également une énumération pour le genre { 'M' , 'F' }.

Tous les attributs de cette classe peuvent être modifiés ou accédés via leur getter et setter respectifs.

La méthode `equals()` a été redéfinie afin de pouvoir comparer deux objets User si on le souhaite et avoir un comportement adéquat.

La méthode `toString()` a elle aussi été redéfinie afin de pouvoir afficher les informations d'un objet User de manière textuelle.

La méthode `findByLoginAndPwd()` permet de retrouver un utilisateur grâce à son login et son mot de passe, `findByLastAndFirstName()` grâce à son nom et prénom, `findByLogin()` grâce à son login `findById()` grâce à son identifiant.

La méthode `getAllUser()` permet d'obtenir un `ResultSet` contenant toutes les entrées de la table `users`.

La méthode `findAll()` renvoie une `List<User>` contenant tous les utilisateurs inscrits dans la base de données.

e. Chat

La classe Chat représente un chat, elle est composée de 5 attributs `title`, `owner`, `description`, `date_creation` et `date_peremption` représentant respectivement le nom du chat, son propriétaire, sa description, sa date de création et sa date de fin de validité.

Tous les attributs de cette classe peuvent être modifiés ou accédés via leur getter et setter respectifs.

La méthode `delete()` permet de supprimer le chat de la base de données.

La méthode `addParticipant()` permet d'ajouter un participant au chat.

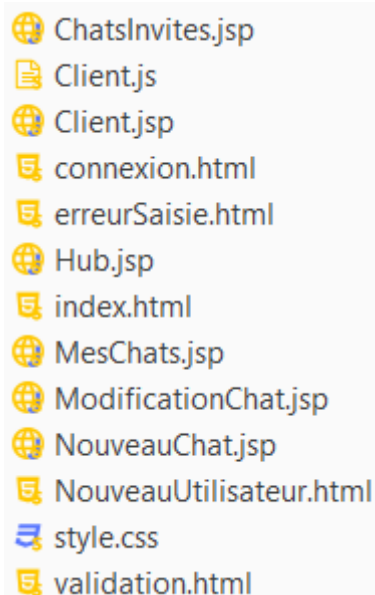
La méthode `getAllChatOwnedByLogin()` permet d'avoir un tableau avec tous les chats appartenant à l'utilisateur en question.

La méthode `getAllInvitedMembers()` permet de retrouver toutes les personnes invitées à un chat, la méthode retourne un `ResultSet`.

La méthode `getAllChatWhereLoginInvited()` permet de retrouver tous les chats où l'utilisateur en question est invité.

La méthode statique `findChatByTitle()` permet de retrouver un chat grâce à son nom.

B. Vue



- ChatsInvites.jsp
- Client.js
- Client.jsp
- connexion.html
- erreurSaisie.html
- Hub.jsp
- index.html
- MesChats.jsp
- ModificationChat.jsp
- NouveauChat.jsp
- NouveauUtilisateur.html
- style.css
- validation.html

Concernant la partie vue, nous avons utilisé des JSP et des pages HTML ainsi qu'une feuille de style CSS pour harmoniser notre application. On utilise les pages JSP pour ajouter du java dans des pages statiques : cela permet par exemple de récupérer des données pour les afficher dans l'application.

Le fichier `connexion.html` permet d'afficher la page de connexion de notre application et elle redirige vers 2 possibilités : s'inscrire et le fichier renvoie vers `NouveauUtilisateur.html` ou l'utilisateur peut rentrer ses identifiants et le fichier fait appel à la servlet de connexion.

Le fichier `client.jsp` correspond à la page où l'utilisateur peut voir les messages envoyés par les autres utilisateurs et peut envoyer des messages à son tour. L'utilisateur peut également quitter le chat et

Le fichier `erreurSaisie.html` permet d'annoncer une erreur de saisie dans le formulaire d'inscription à l'utilisateur et de l'inviter à recommencer son inscription. Il renvoie vers la Servlet Validation qui permet de vérifier la bonne saisie des éléments.

Le fichier `Hub.jsp` est la page sur laquelle arrive l'utilisateur après connexion : il donne les informations de l'utilisateur et affiche le menu de l'application :

- la création d'un chat → renvoie vers la Servlet `NouveauChat` ;
- aller sur un chat de l'utilisateur → renvoie vers la Servlet `MesChats` ;
- un chat sur lequel on a été invité → renvoie vers la Servlet `ChatsInvites`.

Le fichier `MesChats.jsp` permet d'afficher les différents chats que l'utilisateur a créé et que l'utilisateur peut rejoindre. Ce fichier renvoie vers `ConnexionChat`.

Le fichier `ChatsInvites.jsp` permet d'afficher les différents chats auxquels l'utilisateur a été invité et que l'utilisateur peut rejoindre. Ce fichier renvoie vers `ConnexionChat`.

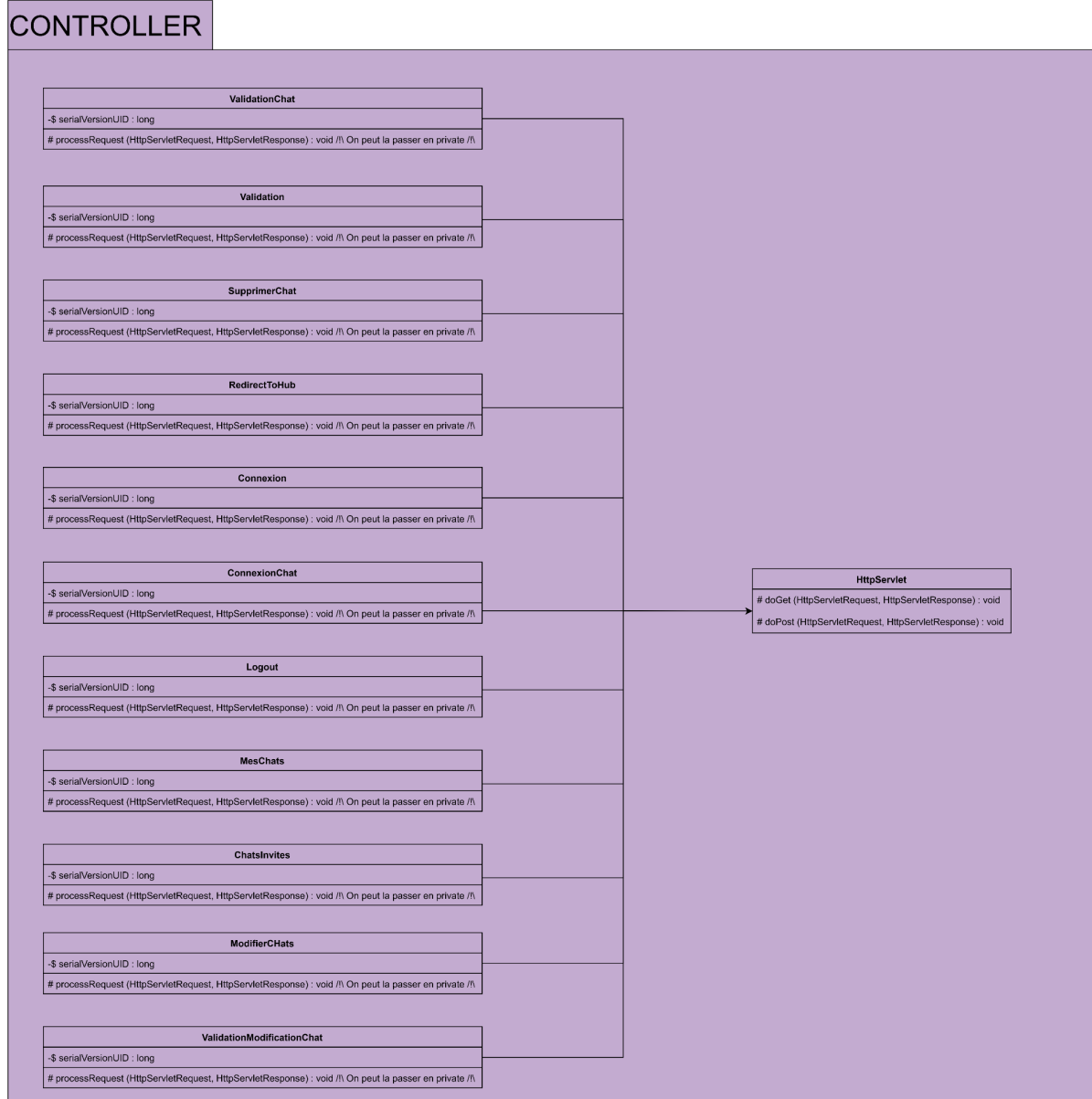
Le fichier `NouveauChat.jsp` permet l'affichage du formulaire de la création d'un nouveau chat par l'utilisateur. Il renvoie vers la Servlet `ValidationChat`.

Le fichier `NouveauUtilisateur.html` permet d'afficher le formulaire de création d'un nouvel utilisateur. Il renvoie vers la Servlet `Validation` qui permet de vérifier la bonne saisie des éléments.

Le fichier `validation.html` permet d'afficher à l'utilisateur un message de confirmation de création du profil utilisateur. Un bouton permet de repartir vers la page de connexion.

Le fichier `ModificationChat.jsp` est un formulaire permettant de modifier un chat déjà existant.

C. Contrôleur



Toutes les classes du package controller héritent de la classe HttpServlet de Jakarta EE.

a. Validation

La Servlet `Validation` permet de s'assurer que les informations saisies par l'utilisateur dans le formulaire d'inscription sont correctes avant d'ajouter l'utilisateur à la base de données.

Une fois ce traitement effectué et si les informations sont correctes l'utilisateur est redirigé vers la page de connexion, dans le cas contraire l'utilisateur est redirigé vers le formulaire d'inscription.

b. Connexion

La Servlet `Connexion` permet de vérifier les informations saisies par l'utilisateur sur la page de connexion, on va vérifier ici qu'un utilisateur correspond bien au login et au mot de passe saisi par l'utilisateur.

Si les informations sont correctes alors l'utilisateur est redirigé vers la page d'accueil de notre application, dans le cas contraire l'utilisateur est informé que sa saisie est erronée et est invité à se reconnecter.

Lors de la redirection vers la page d'accueil un cookie contenant le login de l'utilisateur est créé et une session contenant son login, son rôle et l'objet `User` le représentant est créé.

c. ValidationChat

La Servlet `ValidationChat` permet de s'assurer que les informations saisies par l'utilisateur dans le formulaire de création de chat sont correctes avant de créer le chat dans la base de données et d'ajouter les invités dans la table « `participant_chat` ».

Une fois ce traitement effectué et si les informations sont correctes l'utilisateur est redirigé vers la page de connexion, dans le cas contraire l'utilisateur est averti que le formulaire n'a pas été rempli correctement et est invité à remplir le formulaire ou à revenir sur la page d'accueil.

d. MesChats

La Servlet `MesChats` permet d'ajouter à la session une liste des chats dont l'utilisateur est propriétaire afin de les afficher dans la page `MesChats.jsp` vers laquelle la Servlet redirige l'utilisateur. On fait bien attention dans la servlet à n'inclure dans la liste des chats que les chats dont la date de fin de validité n'est pas dépassée.

e. ChatsInvites

La Servlet `MesChats` permet d'ajouter à la session une liste des chats où l'utilisateur est invité afin de les afficher dans la page `ChatsInvites.jsp` vers laquelle la Servlet redirige l'utilisateur. On fait bien attention dans la servlet à n'inclure dans la liste des chats que les chats dont la date de fin de validité n'est pas dépassée.

f. SupprimerChats

La Servlet `SupprimerChats` permet de supprimer un chat, elle est appelée lorsque l'on clique sur le bouton supprimer un chat sur la page `MesChats.jsp`.

g. ConnexionChat

La Servlet `ConnexionChat` permet de créer un cookie contenant le nom du Chat que l'utilisateur veut rejoindre puis l'utilisateur est redirigé vers le client de chat.

h. Logout

La Servlet `Logout` permet de supprimer la session de l'utilisateur ainsi que d'effacer les cookies créés par notre application.

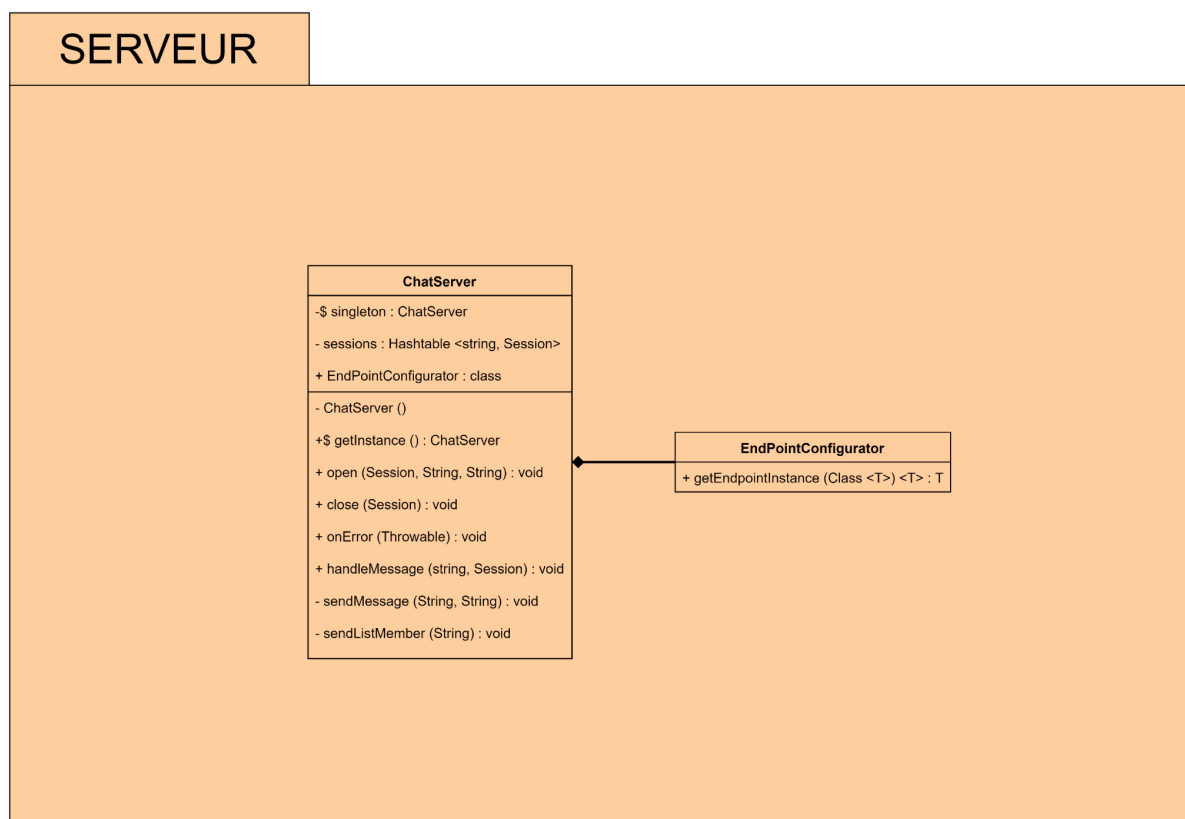
i. ModifierChat

La Servlet `ModifierChat` permet de rediriger l'utilisateur vers un formulaire lui permettant de modifier un chat. Avant la redirection, on stock dans la session le nom du chat et le login de l'utilisateur.

j. ValidationModificationChat

La Servlet `ValidationModificationChat` permet de valider les modifications apportées au chat et de les appliquer sur la base de données, si des informations ne sont pas correctes l'utilisateur en est informé.

D. Serveur



a. ChatServer

L'attribut `sessions` permet de stocker les sessions des utilisateurs connectés au serveur.

L'attribut `singleton` permet la mise en place du design pattern singleton.

La classe `ChatServer` comprend différentes méthodes :

- `getInstance()` : acquisition de l'unique instance `ChatServer`
- `open()` : cette méthode est déclenchée à chaque connexion d'un utilisateur, la session de l'utilisateur est stockée dans l'attribut `sessions` de la classe `ChatServer`.
- `close()` : cette méthode est déclenchée à chaque déconnexion d'un utilisateur, la session de l'utilisateur est retirée de l'attribut `sessions` de la classe `ChatServer`.
- `onError()` : cette méthode est déclenchée en cas d'erreur de communication.
- `handleMessage()` : cette méthode est déclenchée à chaque réception d'un message utilisateur et appelle la méthode `sendMessage()` qui dispatche le message aux utilisateurs d'un chat.
- `sendMessage()` : Une méthode privée, spécifique à notre exemple. Elle permet l'envoi d'un message aux participants de la discussion.
- `sendListMember()` permet de mettre à jour la liste des utilisateurs connectés sur un chat.
- `EndpointConfiguration()` permet de ne pas avoir une instance différente par client. `ChatServer` est donc géré en "singleton" et le configurateur utilise ce singleton.

Partie III - Scénarios & cas d'utilisations théoriques

Veuillez vous connecter sur GitLab à l'aide de vos identifiants. Récupérez le projet en le clonant grâce à la commande : `git@gitlab.utc.fr:dcorneli/sr03_devoir2.git` dans votre shell. Les outils utilisés :

- Eclipse IDE for Enterprise Java and Web Developers
- WampServer Version 3.2.3 64bit pour la base de données MySQL & PhpMyAdmin
- Apache Tomcat v8.0
- mysql-connector-java-8.0.24.jar pour les drivers de la base de données
- jdk-15.0.2_windows-x64_bin

Créez la base de données à l'aide du fichier `dataBaseCreation.sql` puis lancez l'application sur le serveur : vous voilà sur notre application UT'Chat ! Nous allons explorer les fonctionnalités de notre application :

1. Se créer un compte

Dans l'application : vous pouvez entrer vos informations et créer votre compte UT'Chat. Lorsque le compte est correctement créé, l'application vous en informe et vous pouvez à présent vous connecter.

Créer votre compte

Prénom

Pauline

Nom de famille

Breteau

Pseudo

pbreteau

Mot de passe

.....

Homme

☐

Femme

☒

Créer

Retour

Félicitations!

Bienvenu sur UT'Chat, votre compte a été correctement créé!

[Se connecter](#)

Dans la base de données : une fois que vous avez appuyer sur “créer”, les informations sont stockées dans la base de données.

<input type="checkbox"/>	Éditer	Copier	Supprimer	14	Pauline	Breteau	pbreteau	F	0	N3zqcp00
--------------------------	--------	--------	-----------	----	---------	---------	----------	---	---	----------

a. Cas d'erreur 1

Si vous essayez de créer un compte en oubliant une information. Nous avons utilisé la balise `required` pour s'assurer que toutes les informations sont bien saisies.


Créer votre compte

Prénom

Pauline

Nom de famille

Pseudo

 Veuillez renseigner ce champ.

Mot de passe

.....

Homme

☐

Femme

☒

Créer

Retour

b. Cas d'erreur 2

Si vous essayez de créer un compte avec un pseudo déjà utilisé, l'application vous renvoie vers le formulaire pour recréer votre compte en vous indiquant que ce pseudo est déjà utilisé et qu'il faut recommencer.

Créer votre compte

Prénom

Pauline

Nom de famille

Breteau

Pseudo

pbreteau

Mot de passe

.....

Homme

☐

Femme

☒

Créer

Retour

Ce pseudo est déjà utilisé !
Veuillez recommencer.

Prénom

Nom de famille

Pseudo

Mot de passe

Homme

☐

Femme

☒

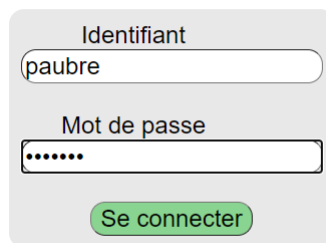
Créer

Retour

2. Se connecter

Lorsque l'utilisateur lance l'application, il arrive sur cet écran qui lui propose de se connecter ou de se créer un profil utilisateur. Une fois connecté, on arrive sur le menu de l'application qui recense les informations de l'utilisateur et donne accès aux différentes fonctionnalités.

Bienvenu sur UT'Chat !



Identifiant
paubre

Mot de passe
.....

Se connecter

Nouveau sur le chat ? [Inscris-toi !](#)

Echec login ou mot de passe erroné

[Revenir vers la page de connexion](#)

[Créer un compte](#)

Si le mot de passe n'est pas correct ou que l'identifiant n'existe pas, on arrive sur cette page qui permet ensuite de retenter une connexion.

3. Créer un chat

Dans l'application : Une fois sur le menu, l'utilisateur peut créer un chat en cliquant sur ce bouton :



Mes informations personnelles

Nom : Cornélie
Prénom : Dylan
Login : Higashu

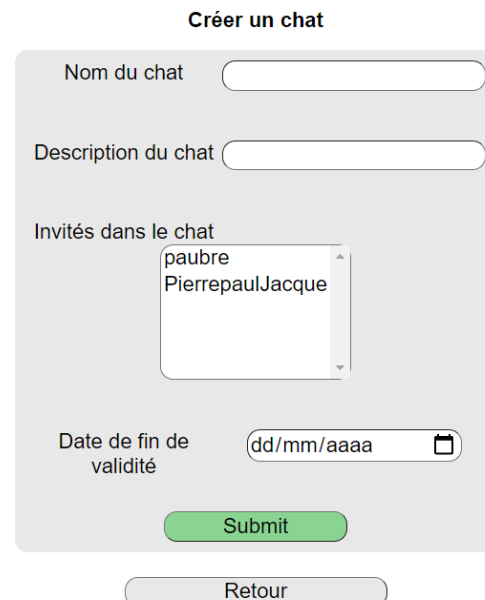
Gestion des chats

Créer un chat

Mes chats

Mes invitations

Logout



Créer un chat

Nom du chat

Description du chat

Invités dans le chat

paubre
PierrepaulJacque

Date de fin de validité dd/mm/aaaa

Submit

Retour

L'application mène l'utilisateur jusqu'à la page de création de chat où l'utilisateur peut remplir un nom, une description, choisir un ou plusieurs invités dans le chat et définir une date de validité. Il n'est pas possible de choisir une date inférieure à la date actuelle.

Créer un chat

Nom du chat

Description du chat

Invités dans le chat

paubre

PierrepaulJacque

Date de fin de validité

[Submit](#)

[Retour](#)

On crée un chat “Anniversaire Pierre” sur le compte `dcorneli`. On peut ainsi entrer une description, inviter l'utilisateur `paubre` et choisir une date de fin de validité. Lorsque l'on clique sur Submit, le chat est créé car toutes les informations sont valides.

On trouve bien le chat dans “Mes chats” depuis le menu depuis le compte `dcorneli` (à gauche).

On le retrouve également dans “Mes invitations” depuis le compte `paubre`.

Mes chats

Se connecter sur [Anniversaire Pierre](#)

[Modifier Anniversaire Pierre](#)
[Supprimer Anniversaire Pierre](#)

[Revenir à l'accueil](#)

Mes invitations

Se connecter sur [Anniversaire Pierre](#)

[Revenir à l'accueil](#)

Dans la base de données :

- a. Cas d'erreur : créer un chat avec une date de fin de validité passée conduit à une erreur et invite l'utilisateur à recommencer.

Créer un chat

Nom du chat

Description du chat

Invités dans le chat

paubre

PierrepaulJacque

Date de fin de validité

Saisie incorrecte

Les informations saisies ne sont pas correctes, veuillez réessayer !

[Recommencer la saisie](#)

[Retourner vers la page d'accueil](#)

- b. Cas d'erreur : créer un chat sans sélectionner d'invité conduit à une erreur et invite l'utilisateur à recommencer.

Créer un chat

Nom du chat

Description du chat

Invités dans le chat

paubre

PierrepaulJacque

Date de fin de validité

Saisie incorrecte

Les informations saisies ne sont pas correctes, veuillez réessayer !

[Recommencer la saisie](#)

[Retourner vers la page d'accueil](#)

4. Modifier un chat

Dans l'application, on constate que seule la personne qui a créé le chat a la possibilité de modifier ou de supprimer le chat. On peut ainsi changer le nom, la description, les invités ou la date comme ci-dessous. Comme pour la création d'un chat, l'utilisateur obtient un message d'erreur si les informations saisies ne sont pas conformes.

Modification de Anniversaire Pierre et Paul

Nom du chat

Description du chat

Invités dans le chat

paubre

PierrePaulJacques

Date de fin de validité

Mes chats

Se connecter sur [Anniversaire Pierre et Paul](#)

[Modifier Anniversaire Pierre et Paul](#)

[Supprimer Anniversaire Pierre et Paul](#)

Dans la base de données : les changements sont bien pris en compte dans la base de données.

id	title	owner	description	date_creation	date_peremption
4	Anniversaire Pierre et Paul	dcorneli	La description change	2021-04-30 21:38:35	2021-05-28

login	chat
paubre	Anniversaire Pierre et Paul
PierrePaulJacques	Anniversaire Pierre et Paul

5. Supprimer un chat

Dans l'application :

Mes chats

Se connecter sur [Chat pour Pierre et paubre](#)

[Modifier Chat pour Pierre et paubre](#)

[Supprimer Chat pour Pierre et paubre](#)

Mes chats

PierrePaulJacques supprime le chat qu'il a créé avec paubre, le chat n'apparaît donc plus dans la liste des chats de PierrePaulJacques ni dans les invitations de paubre.

Dans la base de données :

id	title	owner	description	date_creation	date_peremption
4	Anniversaire Pierre et Paul	dcorneli	La description change	2021-04-30 21:38:35	2021-05-28

6. Converser sur un chat

Lorsqu'on accède à un chat, on arrive sur la page qui permet d'échanger. Cette page donne le titre du chat, sa description, permet de fermer le chat, de revenir à l'accueil et d'envoyer des messages. Enfin, quand un utilisateur se connecte au chat, il apparaît dans la liste des utilisateurs connectés en bas de la page.

Anniversaire Pierre et Paul

La description change

dcorneli : Vient de se connecter sur "Anniversaire Pierre et Paul"

paubre : Vient de se connecter sur "Anniversaire Pierre et Paul"

Liste des utilisateurs connectés

paubre

dcorneli

Anniversaire Pierre et Paul

La description change

dcorneli : Vient de se connecter sur "Anniversaire Pierre et Paul"

paubre : Vient de se connecter sur "Anniversaire Pierre et Paul"

paubre : Vient de se déconnecter

Liste des utilisateurs connectés

dcorneli

Enfin, quand un utilisateur se déconnecte, il est enlevé de la liste des utilisateurs connectés.

Anniversaire Pierre et Paul

La description change

dcorneli : Vient de se connecter sur "Anniversaire Pierre et Paul"

paubre : Vient de se connecter sur "Anniversaire Pierre et Paul"

paubre : Vient de se déconnecter

paubre : Vient de se connecter sur "Anniversaire Pierre et Paul"

paubre : Salut !

dcorneli : Salut ça va ?

paubre : Oui et toi ?

PierrepaulJacque : Vient de se connecter sur "Anniversaire Pierre et Paul"

PierrepaulJacque : Attendez moi !

Liste des utilisateurs connectés

PierrepaulJacque

paubre

dcorneli

Finalement, comme on le voit à gauche, le chat permet d'échanger des messages entre utilisateurs connectés. Les personnes peuvent se connecter en cours de chat et celui-ci peut accueillir tous les invités du chat sans limite de nombre.

On crée un second chat pour bien vérifier l'indépendance entre les différents chats malgré l'utilisation d'un unique serveur. Pour cela, on crée le "chat pour pierre et paubre" et on constate que `PierrepaulJacque` écrit des messages sur ce chat sans que cela n'impacte le chat "Anniversaire Pierre et Paul".

Chat pour Pierre et paubre

Ce chat n'envoie pas de message aux autres chat

PierrepaulJacque : Vient de se connecter sur "Chat pour Pierre et paubre"
 PierrepaulJacque : Ce message ne s'enverra pas sur le Chat Anniversaire Pierre et Paul

Envoyer

Fermer ce chat
Revenir à l'accueil

Liste des utilisateurs connectés

PierrepaulJacque

Anniversaire Pierre et Paul

La description change

dcorneli : Vient de se connecter sur "Anniversaire Pierre et Paul"
 paubre : Vient de se connecter sur "Anniversaire Pierre et Paul"
 paubre : Vient de se déconnecter
 paubre : Vient de se connecter sur "Anniversaire Pierre et Paul"
 paubre : Salut !
 dcorneli : Salut ça va ?
 paubre : Oui et toi ?
 PierrepaulJacque : Vient de se connecter sur "Anniversaire Pierre et Paul"
 PierrepaulJacque : Attendez moi !
 PierrepaulJacque : Vient de se déconnecter
 paubre : Vient de se déconnecter

Envoyer

Fermer ce chat
Revenir à l'accueil

Liste des utilisateurs connectés

dcorneli

7. Voir ses invitations

Dans l'application : suite aux différents chats créés, `dcorneli` a créé deux chats et a invité `paubre` à participer à ceux-ci. Quand `paubre` se connecte, elle voit bien ces chats apparaître dans ses invitations. Si la date de fin de validité d'un chat est dépassée, il n'apparaît pas sur la page mes Chats pour `dcorneli` et sur la page mes invitations sur `paubre` et `Pierrepaul` même si ils ont été invités.

Mes informations personnelles

Nom : Breteau
Prénom : Pauline
Login : paubre

Gestion des chats

Créer un chat

Mes chats

Mes invitations

Logout

Mes invitations

Se connecter sur [Anniversaire Pierre et Paul](#)

Se connecter sur [Chat pour Pierre et paubre](#)

Revenir à l'accueil

Dans la base de données :

login	chat
paubre	Anniversaire Pierre et Paul
PierrepaulJacque	Anniversaire Pierre et Paul
paubre	Test date fin valide passe
PierrepaulJacque	Test date fin valide passe

Dans la base de données, la table subscription ci-dessus permet de lier un login à un chat et permet de modéliser les invités des différents chats.