Binôme : -Myriem MOULOUEL
-Roukia MAHAD-ISMAIL
Groupe2: technoWeb (3I017)

Tableaux des Services :

Création d'un utilisateur

Nom du web service	Création d'un utilisateur
URL du web service	La requête http://localhost:PORT/api/user fait appel à la route **PUT(« /user »)** qui a son tour fait appel à la méthode **create(…)** de la classe Users si tout les champs cité plus tôt sont bien renseigner (ex : mail=X@latoile.fr, Login=X, password=XX, lastname=X, firstname=X) alors on créée un tuple correspondant dans la table User de la base de donnée db.
Description du service	créer un utilisateur sur le service Latoile
Paramètres en entrée	Login, password, lastname, firstname
Format de sortie	JSON
Exemple de sortie	{"login":"3", "password":"ttt", "lastname":"tOty","firstname":"dido"} /status 201
Erreurs possibles	/Status 400=> oublier un champs /Status 500=> erreur dans la promesse qui permet de créer un utilisateur
Avancement du Service	100%
Classes/Fichiers JavaScript	Classe :Users Fichiers : -/src/entities/users.js -/src/api.js
Informations additionnelles	Point d'entrée de l'application web : -/src/app.js -/src/index.js

Connexion

Nom du web service	Connexion d'utilisateur
URL du web service	"/user/login"
Description du service	Dès l'appel de la requête POST http://localhost:PORT/api/use/login on vérifie les erreurs liés à la requête http(ex : login ou password non renseigner), puis on vérifie l'existence du compte avec la méthode exists(login) de la classe Users. Et, on examine le password du login checkpassword(). -Création d'un identifiant de session et l'envoie dans les cookie
Paramètres en entrée	login, password
Format de sortie	-Format JSON -Redirect(/user/user_id)
Exemple de sortie	{ status : 200 Message : Vous êtes connecté à latoile} Et une clé de session
Erreurs possibles	erreur 400 => login et password manquants erreur 403 => login ou password invalide erreur 401 => Utilisateur inconnu
Avancement du Service	100%
Classes/Fichiers JavaScript	Classe : Users Fichiers : /src/entities/users.js, /src/api.js, /src/app.js, /src/index.js
Informations additionnelles	A l'issu d'une connexion établie, l'utilisateur est rediriger vers une route « /home »

Binôme : -Myriem MOULOUEL
-Roukia MAHAD-ISMAIL
Groupe2: technoWeb (3I017)

Déconnexion

Nom du web service	Logout
URL du web service	"/user/logout"
Description du service	Dès l'appel de la requête DELETE http://localhost:PORT/api/user/logout on vérifie si l'utilisateur a une clé de session active et on supprime cette clé en le redirigeant vers la route du login.
Paramètres en entrée	Aucun, on récupère tout avec les cookies du client.
Format de sortie	Format JSON + Redirection vers une route
Exemple de sortie	{status : 200, Message : vous êtes déconnecter}
Erreurs possibles	404 une erreur liée à la clé de session 500 autre erreur liés aux middleware
Avancement du Service	70%
Classes/Fichiers JavaScript	/src/api.js, /src/app.js, /src/index.js
Informations additionnelles	

suppression Utilisateur

Nom du web service	suppression Utilisateur
URL du web service	"/user/:user_id"
Description du service	supprime l'utilisateur de la table users de la base de données db, seul l'utilisateur peut supprimer son compte. La requête "http DELETE localhost:4000/api/user/:user_id" fait appel à la route delete de api.js qui à son tour appelle la méthode deleteUser de users.js, après avoir vérifié que le user existe dans la table users de bd et password et login correctes
Paramètres en entrée	login, password
Format de sortie	un message sous forme de chaine de caractere "delete user user_id succeded"
Exemple de sortie	delete user 9 succeded
Erreurs possibles	(user inexistant)=>status 404 (password incorrect)=>status 406 (delete invalid)=>status 500
Avancement du Service	on créé une base de donnée sqlite3 qu'on appelle db, qui contient déjà la table users des utilisateurs avec {login, password, lastname, firstname}
Classes/Fichiers JavaScript	classe : users fichier : src/entities/users.js, src/api.js,src/app.js, src/index.js
Informations additionnelles	on créé une base de donnée sqlite3 qu'on appelle db, qui contient déjà la table users des utilisateurs avec {login, password, lastname, firstname}

Binôme : -Myriem MOULOUEL
-Roukia MAHAD-ISMAIL
Groupe2: technoWeb (3I017)

visualisation de followings

Nom du web service	visualisation de followings (le sien ou d'autrui)
URL du web service	"/followings/:user_id"
Description du service	affiche les followings et les tweets de l'utilisateur dont le user_id est passé en argument, avec la requête "http GET localhost:4000/api/user/profile/followings/:user_id" dans la route get de api.js qui fait appel à getFollowings de users.js pour récupérer les followings, s'assure que l'utilisateur existe, ensuite renvoie tous les followings
Paramètres en entrée	(aucun)
Format de sortie	JSON /status 200
Exemple de sortie	{1,2}
Erreurs possibles	(utilisateur inexistant) => status 404 (followings not founded/tweets not founded) => status 204 (erreur interne du serveur) => status 500
Avancement du Service	50%
Classes/Fichiers JavaScript	classe : users, messages fichier : src/entities/users.js, src/api.js, src/entities/messages.js
Informations additionnelles	Communs à tous les services: src/app.js src/index.js

visualisation de commentaires

Nom du web service	visualisation de commentaires (le sien ou d'autrui)
URL du web service	"/messages/:user_id"
Description du service	affiche les tweets de l'utilisateur dont le user_id est passé en argument, avec la requete "http GET localhost:4000/api/user/profile/comments/:user_id" dans la route get de api.js qui fait appel à getTweets de messages.js pour recuperer les tweets, s'assure que l'utilisateur existe, ensuite renvoie tous les tweets
Paramètres en entrée	(aucun)
Format de sortie	JSON /status 200
Exemple de sortie	{...}
Erreurs possibles	(utilisateur inexistant) => status 404 (followings not founded/tweets not founded) => status 204 (erreur interne du serveur) => status 500
Avancement du Service	0%
Classes/Fichiers JavaScript	classe : users, messages fichier : src/entities/users.js, src/api.js, src/entities/messages.js
Informations additionnelles	

Binôme : -Myriem MOULOUEL
-Roukia MAHAD-ISMAIL
Groupe2: technoWeb (3I017)

Unfollow user

Nom du web service	Unfollow user
URL du web service	"/follow/delete/:user_id"
Description du service	supprime le follow d'un utilisateur login de la table frienduser de la base de données db, seul l'utilisateur peut supprimer son follow à quelqu'un. La requete "http DELETE localhost:4000/api/follow/delete/:user_id login=id" fait appel a la route delete de api.js qui à son tour appelle la methode deleteFollowing de users.js, après avoir verifié que le user et le following existent dans la table users de bd et dans la table frienduser et password et login correctes
Paramètres en entrée	login (du following)
Format de sortie	un message sous forme de chaine de caractere "delete followuser user_id succeded"
Exemple de sortie	"delete followuser 3 succeded"
Erreurs possibles	(user inexistant)=>status 404 (password incorrect)=>status 406 (delete invalid)=>status 500
Avancement du Service	0%
Classes/Fichiers JavaScript	classe : users fichier : src/entities/users.js, src/api.js,
Informations additionnelles	

Recherche d'utilisateurs à partir de leur firstname et lastname

Nom du web service	Recherche d'utilisateurs à partir de leur firstname et lastname
URL du web service	"/searchuser"
Description du service	à partir du firstname et/ou du lastname on recherche des utilisateurs. La requête "http GET localhost:4000/api/searchuser login=id firstname=xx lastname=xx" fait appel a la route get de api.js qui à son tour appelle la méthode searchUser de users.js, fait une requête SQL pour prendre tous les logins des personnes dont firstname ou lastname match
Paramètres en entrée	login, (lastname ou fisrtname)
Format de sortie	JSON
Exemple de sortie	{1, 2, 3}
Erreurs possibles	(internal server error) => 500
Avancement du Service	0%
Classes/Fichiers JavaScript	classe : users fichier : src/entities/users.js, src/api.js,
Informations additionnelles	

Binôme : -Myriem MOULOUEL
-Roukia MAHAD-ISMAIL
Groupe2: technoWeb (3I017)

Suppression d'un tweet

Nom du web service	Suppression d'un tweet
URL du web service	"/message/delete/user_id"
Description du service	supprime le message de la base de données mongo, seul l'utilisateur peut supprimer son compte. La requête "http DELETE localhost:4000/api/message/delete/:user_id message=x" fait appel a la route delete de api.js s'assure que le password est correct après à son tour appelle la méthode deleteMessage de messages.js qui supprime le message
Paramètres en entrée	password, message
Format de sortie	message sur le terminal
Exemple de sortie	"delete message succeeded"
Erreurs possibles	(user inexistant)=>status 404 (password incorrect)=>status 406 (delete invalid)=>status 500
Avancement du Service	0%
Classes/Fichiers JavaScript	classe : users, messages fichier : src/entities/users.js, src/api.js, src/entities/messages.js
Informations additionnelles	

Like/Unlike comments

Nom du web service	Like/Unlike comments
URL du web service	"/messages/reaction/:user_id"
Description du service	Like ou Unlike un commentaire. La requête "http POST localhost:4000/api/message/reaction/:user_id password=xx message=xx like=x unlike=x" fait appel a la route get de api.js s'assure que le password est correct, s'assure que soit (like=1 et unlike=0) soit (like=0 et unlike=1) soit (like=0 et unlike=0) qui à son tour appelle la méthode insertLike et insertUnlike de messages.js, fait une requête find de NoSQL pour inserer dans likedb ou unlikedbun like ou un unlike pour un message donné
Paramètres en entrée	password, message, like, unlike
Format de sortie	JSON
Exemple de sortie	{"login": "3", "message":xxx, "like":"1", "unlike":"0"} /status 201
Erreurs possibles	/Status 400=> oublier un champs /Status 500=> erreur dans la promesse qui permet de créer un utilisateur
Avancement du Service	0%
Classes/Fichiers JavaScript	classe : users, messages fichier : src/entities/users.js, src/api.js, src/entities/messages.js
Informations additionnelles	on créé deux bases de données de types mongodb const likedb = new DataStore(); const unlikedb = new DataStore()

Binôme : -Myriem MOULOUEL
-Roukia MAHAD-ISMAIL
Groupe2: technoWeb (3I017)

Ajouter un ami

Nom du web service	Follow User
URL du web service	"/follow/add/:autre_user"
Description du service	ajouter un ami
Paramètres en entrée	Body vide
Format de sortie	JSON
Exemple de sortie	{status :201, Message : vous suivez :autre_user}
Erreurs possibles	401 : utilisateur ami inconnu 500 : Problème avec la bdd
Avancement du Service	90%
Classes/Fichiers JavaScript	Follow.js => Class Follow User.js=> Class Users Api.js
Informations additionnelles	

Ajouter un message

Nom du web service	Add tweet
URL du web service	"/message/add/"
Description du service	Ce service permet d'ajouter un post d'un utilisateur.
Paramètres en entrée	Sid, message
Format de sortie	JSON qui contient le message
Exemple de sortie	{status:201, Message:{login :2, message: "hello world" } }
Erreurs possibles	400 : message vide 500 : Problème avec la bdd
Avancement du Service	10%
Classes/Fichiers JavaScript	/src/entities/message.js => Class Message /src/entities/users.js=> Class Users
Informations additionnelles	Sid contient l'id de l'utilisateur courant

Republier un message avec un commentaire

Nom du web service	RepostMessage
URL du web service	"/message/get/"
Description du service	Dès qu'on fait un POST /message/get ça voudrait dire que l'utilisateur voudrait republier un message d'un utilisateur donné et le reposter avec un commentaire.
Paramètres en entrée	message, comment, login
Format de sortie	JSON
Exemple de sortie	{ TheMessage : « blablabla », TheComment : « reblablabla » }
Erreurs possibles	410 : Commentaire Vide 400 : Message est indisponible 500 : problème lié à la bdd ou autre
Avancement du Service	0%
Classes/Fichiers JavaScript	/entities/message.js => Class Message /entities/users.js=> Class User
Informations additionnelles	

Binôme : -Myriem MOULOUEL
-Roukia MAHAD-ISMAIL
Groupe2: technoWeb (3I017)

Rechercher un Message

Nom du web service	Search Message
URL du web service	/message/rechercher
Description du service	A partir d'un mot trouver les messages contenant ce mot et affiche les messages de mes amis en premier.
Paramètres en entrée	{Object :Mot-clé }
Format de sortie	JSON + Boolean true (avec les messages) Boolean false
Exemple de sortie	{status : 404 Message : Aucun message avec ce mot-clé}/status 201
Erreurs possibles	/status:404 => si il n'y a pas de message qui match avec le mot-clé /status:410 => si aucun mot renseigner
Avancement du Service	0%
Classes/Fichiers JavaScript	/entities/message.js => Class Message /entities/users.js=> Class Users /api.js
Informations additionnelles	

Modifier Informations Utilisateurs

Nom du web service	UpdateUSer
URL du web service	/user/:user_id/update
Description du service	Modifier le mot de passe, le prénom ou le nom. Pour cela on vérifie s'il existe une session active pour cet utilisateur si oui on modifie le tuple correspondant dans la base.
Paramètres en entrée	Password || Firstname||Lastname
Format de sortie	JSON
Exemple de sortie	{login: titi, Password :nouveau, Firstname :ancien,Lastname :nouveau}/status 201
Erreurs possibles	/status:406=> Impossible de remettre l'ancien /status:500=> Erreur interne
Avancement du Service	0%
Classes/Fichiers JavaScript	/src/entities/users.js => Class Users /src/api.js
Informations additionnelles	

Modification d'un message

Nom du web service	UpdateMessage
URL du web service	/message/:idMessage
Description du service	Ce service permet de modifier une publication d'un utilisateur existant et connecté.
Paramètres en entrée	message,login
Format de sortie	JSON
Exemple de sortie	{login: id, message: nouveau message} /status 201
Erreurs possibles	/status:401=> Message n'existe pas /status:406=> Nouveau message vide /status:500=> Problème lié à la bdd
Avancement du Service	0%

Binôme : -Myriem MOULOUEL
-Roukia MAHAD-ISMAIL

Groupe2: technoWeb (3I017)

Classes/Fichiers JavaScript	/src/entities/message.js => Class Message /src/api.js
Informations additionnelles	