

ASI-2

ASI-2: Framework frontend, services orientés architecture (S9)

Diagramme Monolithique :

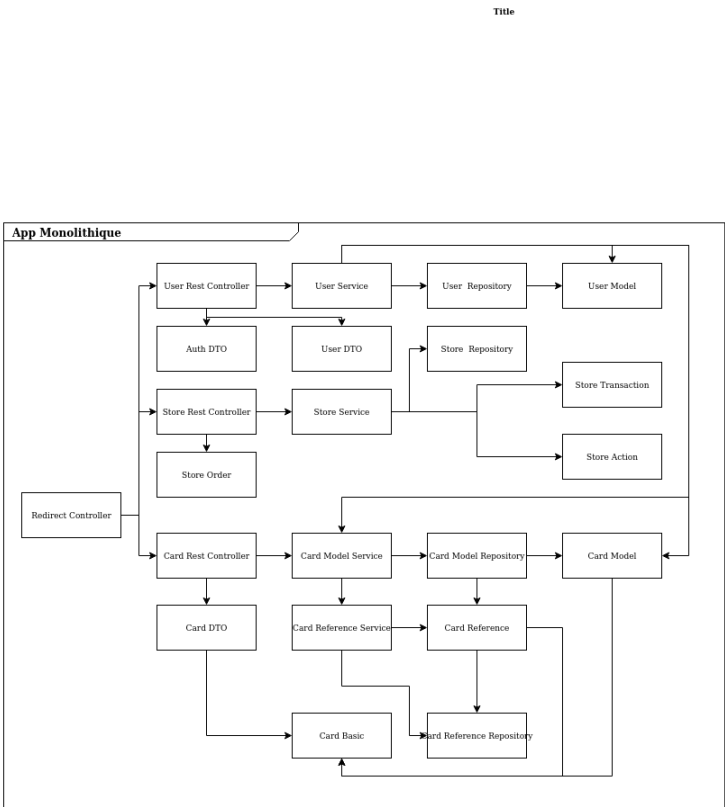
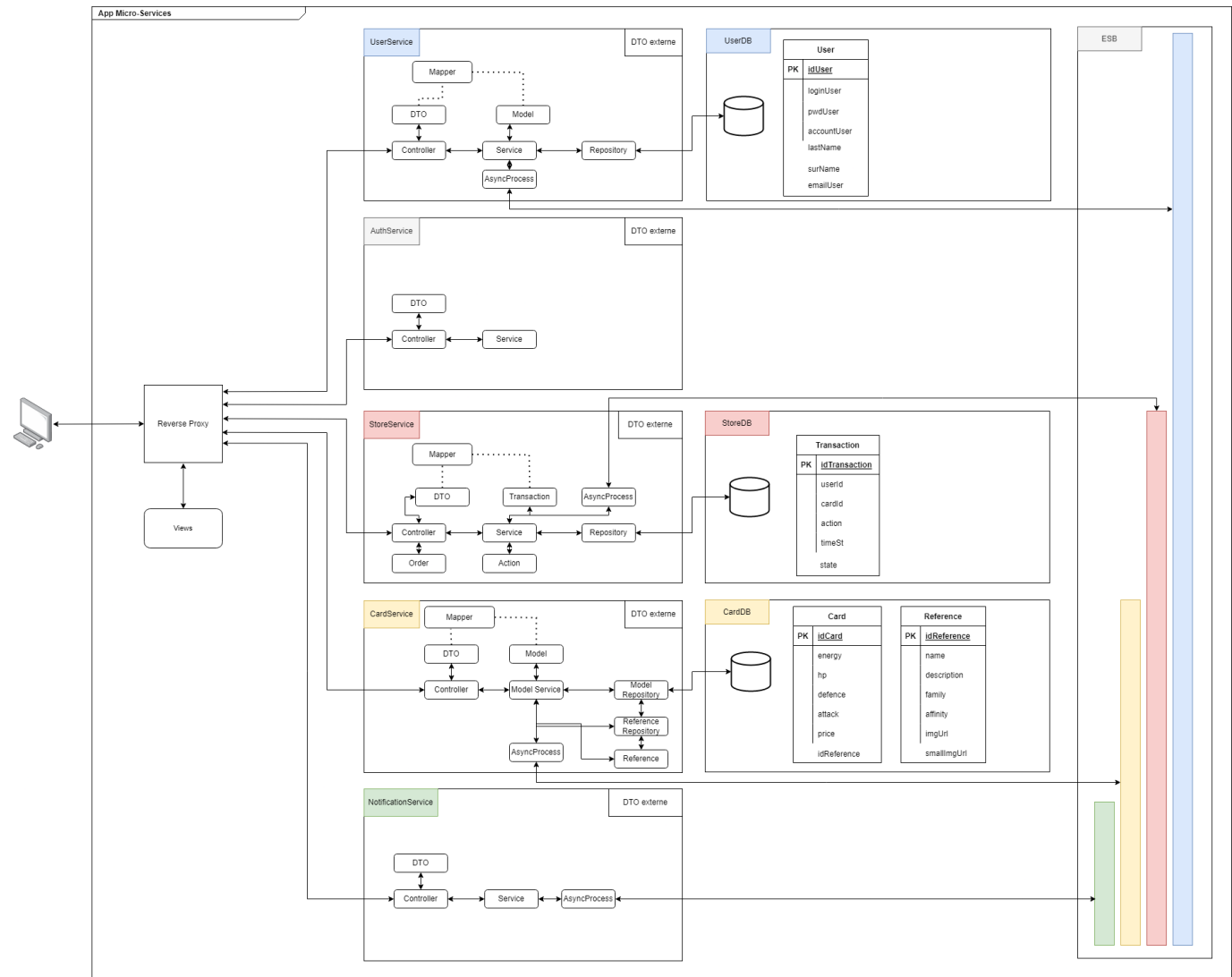


Diagramme des micro-services :



Liste requêtes :

Type	Source	Destination	Requête	Attributs
GET	Client	Auth	/api/auth/	user_id, mdp
GET	Auth	User	/api/user/	user_id
GET	Client	Store	/api/store/buy/	user_id, card_id
POST	Store	Card	/api/card/	idTransaction, card_id , user_id
POST	Store	User	/api/user/	idTransaction, user_id, money
GET	Card	Store	/api/store/transaction/card/	idTransaction, card_id
GET	User	Store	/api/store/transaction/user/	idTransaction, user_id

Composants REACTJS :

NavBar :

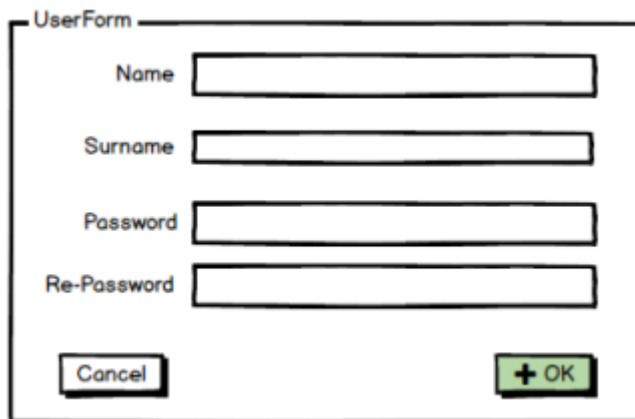
Paramètres : Utilisateur, pageTitle



Header bar containing a balance of 5000 \$ on the left and a user profile icon with the name Jdoe on the right.

Log Up :

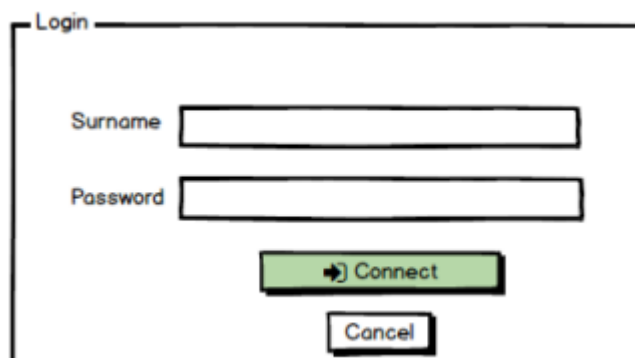
Sorties : addUser()



UserForm dialog box with four input fields: Name, Surname, Password, and Re-Password. At the bottom, there are two buttons: Cancel and + OK.

Log In :

Sorties : login()

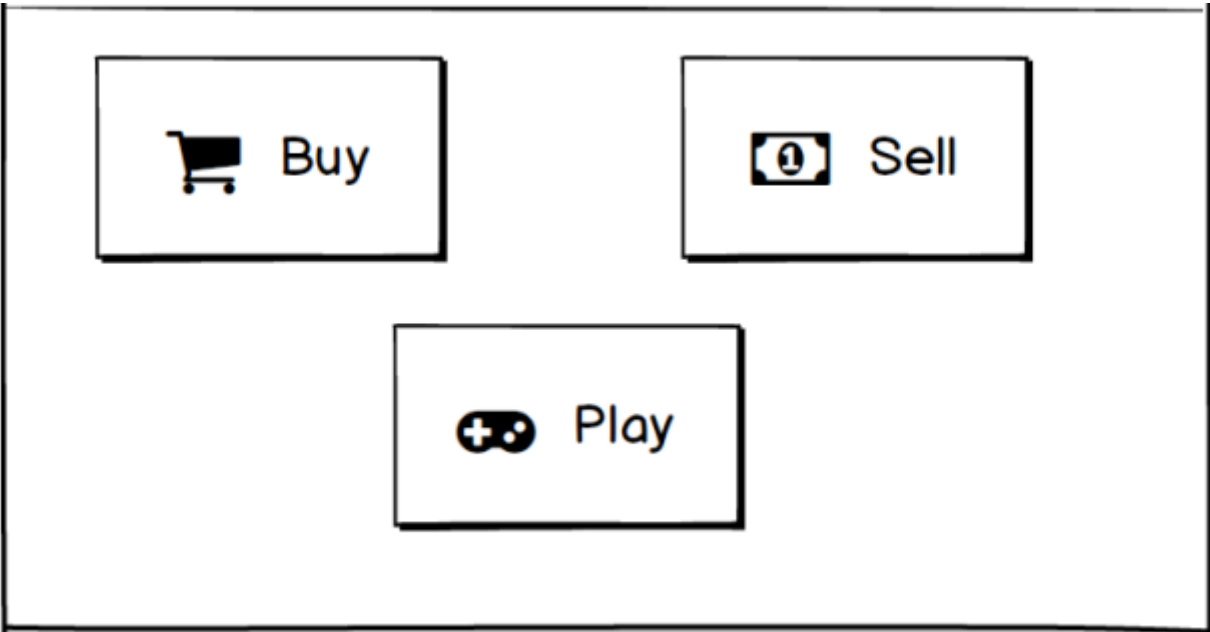


Login dialog box with two input fields: Surname and Password. At the bottom, there are two buttons: Connect (with a right arrow icon) and Cancel.

Home :

Composants : NavBar

Liens : Buy, Shell, Play



Card :

Paramètres : Card



CardList :

Paramètres : CardList Sortie: CardSelect

Name	Descript	Family	Affinity	Energy	HP	Price	
blabla	blabla blablabla	blabla	blabla	blabla	blabla	blabla	50\$
blabla	blabla blablabla	blabla	blabla	blabla	blabla	blabla	50\$
blabla	blabla blablabla	blabla	blabla	blabla	blabla	blabla	50\$
blabla	blabla blablabla	blabla	blabla	blabla	blabla	blabla	50\$
blabla	blabla blablabla	blabla	blabla	blabla	blabla	blabla	50\$
blabla	blabla blablabla	blabla	blabla	blabla	blabla	blabla	50\$
blabla	blabla blablabla	blabla	blabla	blabla	blabla	blabla	50\$
blabla	blabla blablabla	blabla	blabla	blabla	blabla	blabla	50\$

CardTransfert :

Paramètres : TransactionType, TitrePage

Sorties : Action(Buy or Sell)

Buy :

Market


CARDLIST

Name	Descript	Family	Affinity	Energy	HP	Price	
blabla	blabla blablabla	blabla	blabla	blabla	blabla	blabla	50\$
blabla	blabla blablabla	blabla	blabla	blabla	blabla	blabla	50\$
blabla	blabla blablabla	blabla	blabla	blabla	blabla	blabla	50\$
blabla	blabla blablabla	blabla	blabla	blabla	blabla	blabla	50\$
blabla	blabla blablabla	blabla	blabla	blabla	blabla	blabla	50\$
blabla	blabla blablabla	blabla	blabla	blabla	blabla	blabla	50\$
blabla	blabla blablabla	blabla	blabla	blabla	blabla	blabla	50\$
blabla	blabla blablabla	blabla	blabla	blabla	blabla	blabla	50\$

CARD

20 Happy Tree Fam! 50

Super John



Buy

Sell :

My CARD


CARDLIST

Name	Descript	Family	Affinity	Energy	HP	Price	
blabla	blabla blablabla	blabla	blabla	blabla	blabla	blabla	50\$
blabla	blabla blablabla	blabla	blabla	blabla	blabla	blabla	50\$
blabla	blabla blablabla	blabla	blabla	blabla	blabla	blabla	50\$
blabla	blabla blablabla	blabla	blabla	blabla	blabla	blabla	50\$
blabla	blabla blablabla	blabla	blabla	blabla	blabla	blabla	50\$
blabla	blabla blablabla	blabla	blabla	blabla	blabla	blabla	50\$
blabla	blabla blablabla	blabla	blabla	blabla	blabla	blabla	50\$
blabla	blabla blablabla	blabla	blabla	blabla	blabla	blabla	50\$

CARD

20 Happy Tree Fam! 50

Super John



SELL

UML monolithique :
diagramme UML card

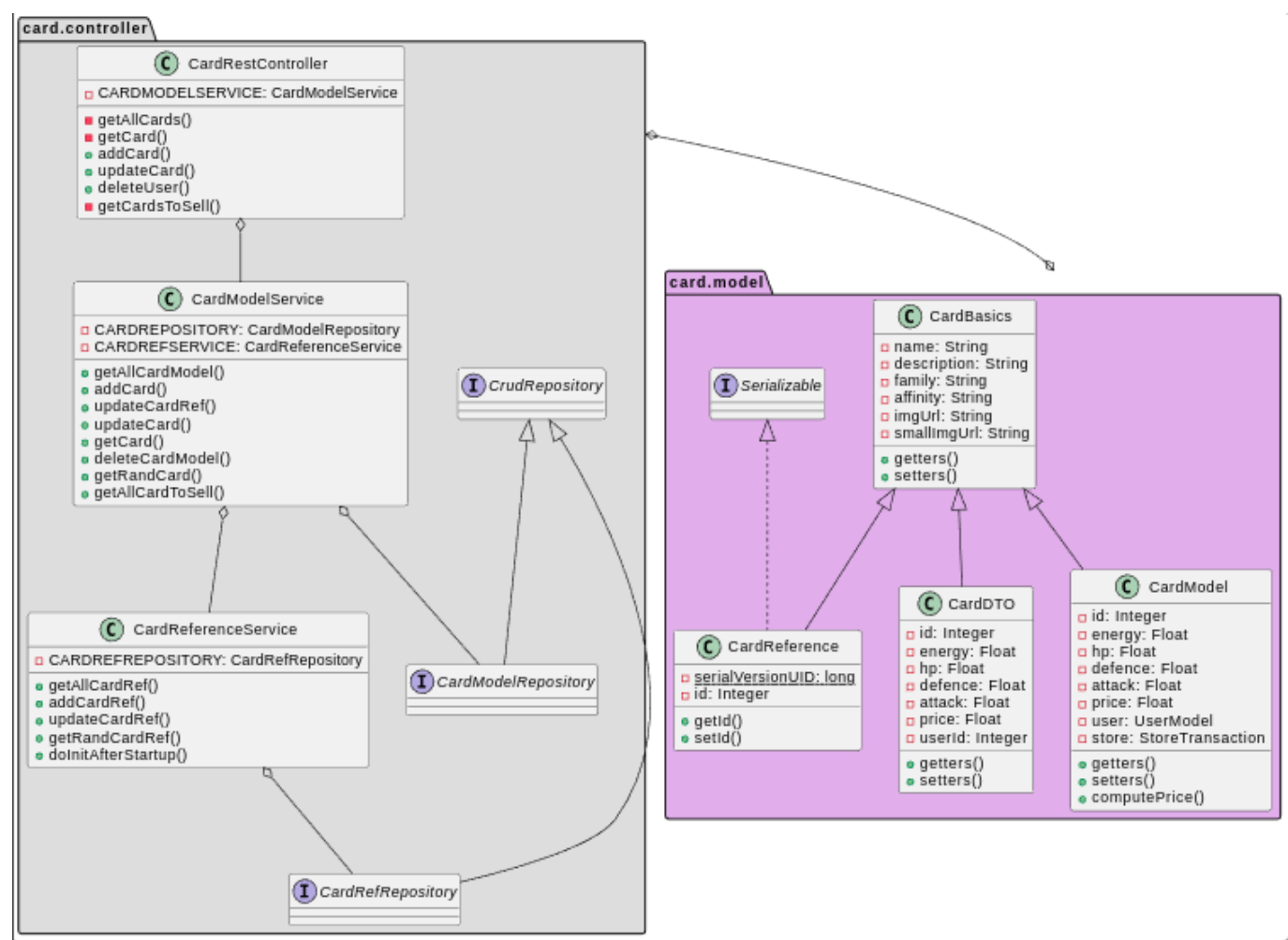


diagramme UML store

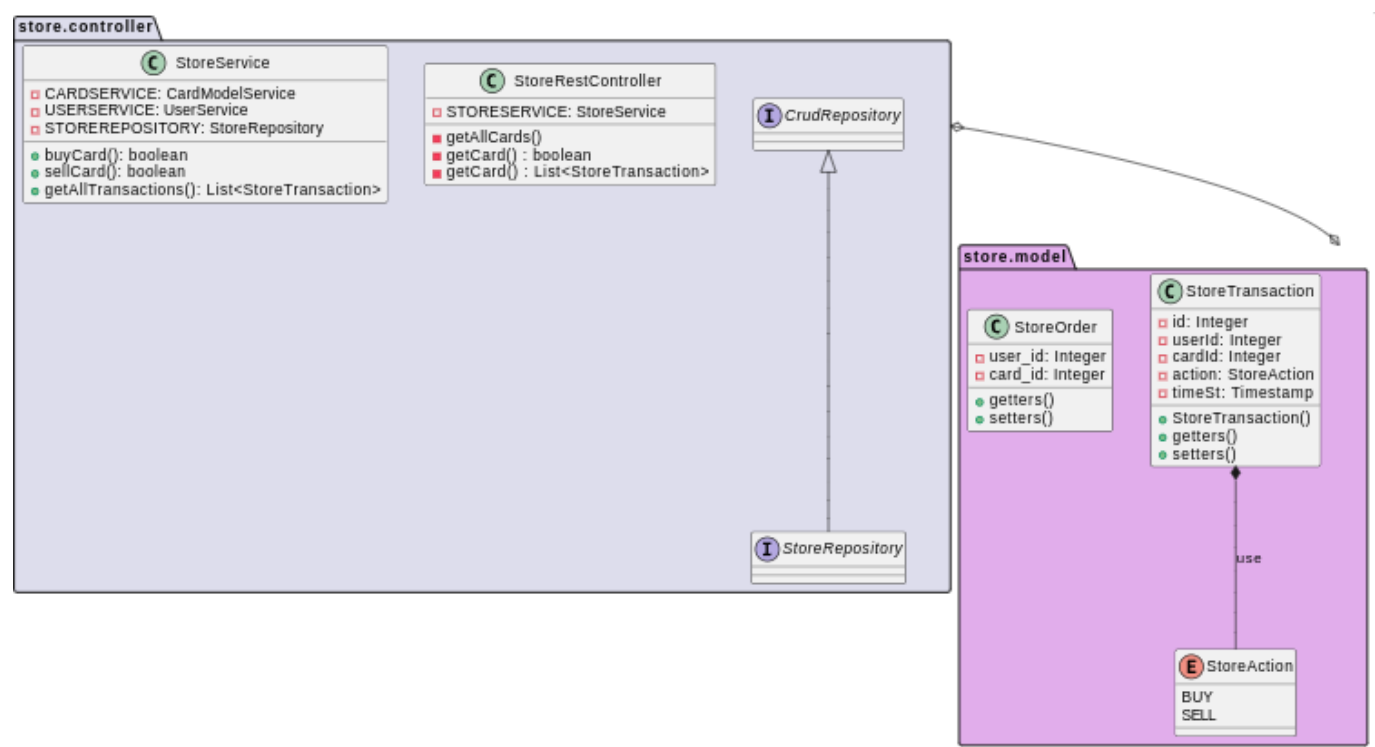
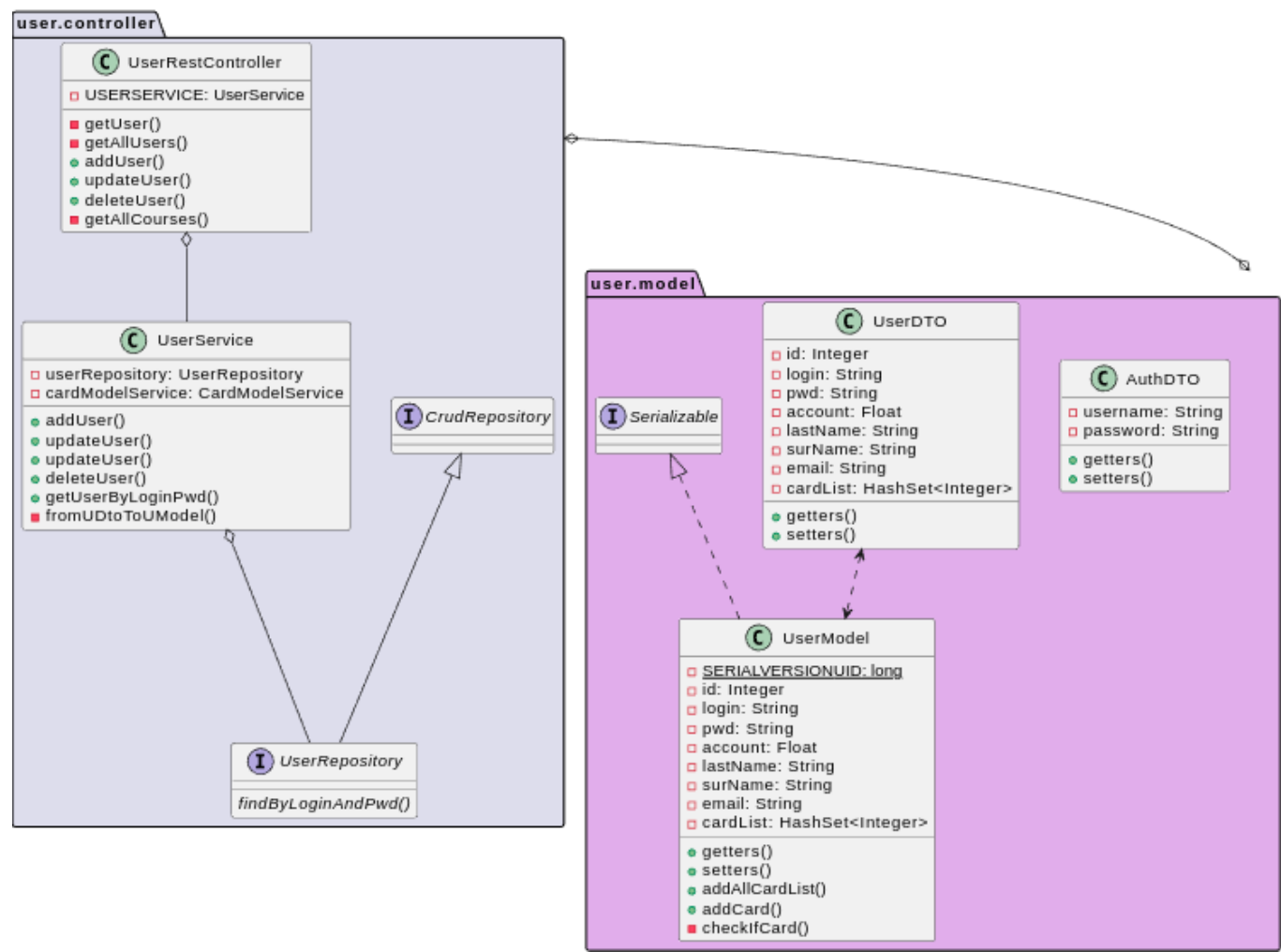


diagramme UML user



Avantages et inconvénients des bus de communication

	ActiveMq	RabbitMq	Kafka	MSMQ
Plateforme	Open Source, Apache	Open source, Mozilla	Open Source, Apache	Microsoft
Protocoles	Middleware orienté message (MOM)	AMPQ (Advanced Message Queuing Protocol)	Protocole binaire basé sur TCP	Protocoles propriétaires non standardisés
Fonctionnement	C'est un système dit push-type, où l'expéditeur envoie les messages au destinataire et il est responsable de leur bonne réception.	Le message est échangé via un serveur central, impossible de le transmettre directement au destinataire. Les queues des messages sont gérées par le serveur.	C'est un système dit pull-type où les destinataires viennent piocher leur message du broker et ils sont responsables de la bonne consommation des messages.	Mode décentralisé, chaque application gère sa propre queue et peut verser dans la queue des destinataires.
Avantages	Facilite la montée en charge (grâce à ses noeuds). Possibilité d'utiliser des queues ou des topics (en point-to-point ou en publish/subscribe). Double sécurité grâce aux doubles couches de la couche SSL/TLS. API Rest est une API autonome au langage.	Messagerie asynchrone, interface de gestion utilisateur, scalable, basé sur le cloud (TLS, LDAP), sécurité (2 authentification (2FA), SSL ...)	Traitement de grand ensemble de données, tolérance aux pannes, évolutivité, gestion en temps réel, durabilité des messages	Natif Windows, queues transactionnelles (possibilité de livraison unique), routage dynamique, pas de perte de message (sauvegarde disque dur), large documentation, scalable

	ActiveMq	RabbitMq	Kafka	MSMQ
Inconvénients	Documentation peu pratique, les messages peuvent arriver désordonnés, prise en main compliquée si non familier avec Apache.	Non transactionnel, Besoin d'Erlang, difficulté de traitement de big data, implémentation des clusters complexe, peu documenté	Topics non supportés, performances pouvant chuter dans des contextes de messages modifiés, ou compressés, comportement maladroit lorsque le nombre de queues augmente	Pas de publish/subscribe natif, gestion de l'espace de stockage sur chaque machine importante, configuration lourde

Avantages et inconvénients des Framework Front-End

	Angular	Vue.js	React
Avantages	<ul style="list-style-type: none"> - Il est possible de mettre à jour les modifications du modèle dans la vue et inversement. - Comme la plupart des fonctionnalités importantes, par exemple la liaison de données bidirectionnelle sont fournies par défaut. La quantité de code est réduite. - Découplage des composants des dépendances en les définissant comme des éléments externes. - L'injection de dépendances permettant la réutilisabilité et la facilité de gestion des composants. - L'apprentissage est facilité par le soutien d'une grande communauté utilisateur 	<ul style="list-style-type: none"> - Large documentation détaillée. - Syntaxe simple. - Structure d'application simple. 	<ul style="list-style-type: none"> - La réutilisabilité des composants. Utilisation du DOM virtuel permettant des performances cohérentes et transparentes. - Possibilité d'utiliser des outils de développement avancés, facilitant le développement
Inconvénients	<ul style="list-style-type: none"> - Les applications dynamiques peuvent manquer de performantes à cause de leur structure complexe et de leur taille. (ceci peut être contrebalancé par l'optimisation de code) 	<ul style="list-style-type: none"> - Les composants manquent de stabilité. - Communauté relativement restreinte. - Barrière de langage avec les plugins et les composants (la plupart des plugins sont rédigés en chinois). 	<ul style="list-style-type: none"> - Les mises à jour multiples et constantes de React, rendent difficile la rédaction de documentation appropriée. - Les complexités de JSX à tendance à rendre difficile le début de l'apprentissage. - React n'est que des solutions front-end.