Équipe 12

Poly Paint
Document d'architecture logicielle

Version 2.2

Historique des révisions

Date	Version	Description	Auteur
2019-02-01	1.0	Rédaction initiale	Chelsy Binet, Olivier Lauzon, Alexis Loiselle, Sébastien Cadorette, Sébastien Labine et William Sévigny
2019-04-01	2.0	Début des corrections de la réponse à l'appel d'offres	Olivier Lauzon
2019-04-02	2.1	Suite des corrections de la réponse à l'appel d'offres	Olivier Lauzon et William Sévigny
2019-04-07	2.2	Relecture et approbation finale	Chelsy Binet, Olivier Lauzon, Alexis Loiselle, Sébastien Cadorette, Sébastien Labine et William Sévigny

Table des matières

Document d'architecture logicielle

1. Introduction

Le document d’architecture logicielle décrit la structure de haut niveau d’une application. Des précisions sur les objectifs et contraintes ainsi que leur impact architectural sont présentés. La structure de l’application est décrite selon quatre points de vue: la vue des cas d’utilisation, la vue logique, la vue des processus et la vue de déploiement. Enfin, les caractéristiques de taille et de performance de l’application qui pourraient affecter l’architecture sont exposées.

2. Objectifs et contraintes architecturaux

La plateforme PolyPaint Pro doit avoir deux clients: le premier dans un environnement Windows et le second dans un environnement iOS. Le serveur doit être réutilisable et multiplateforme pour faciliter la mise en marché. L’architecture des comptes utilisateurs doit être sécurisée avec cryptage pour assurer la sécurité des données personnelles des utilisateurs. Le code doit être simple et facilement maintenable pour respecter le budget et l’échéancier. L'architecture de la plateforme doit être évolutive pour permettre l’ajout de nouvelles fonctionnalités dans l’avenir et doit maximiser la réutilisabilité de code libre de droits pour accélérer le développement.

3. Vue des cas d’utilisation

Afin de bien décrire l’utilisation de la plateforme PolyPaint Pro, nous avons identifié les principaux cas d’utilisation.

Le premier cas d'utilisation est l'utilisation générale de l'application.

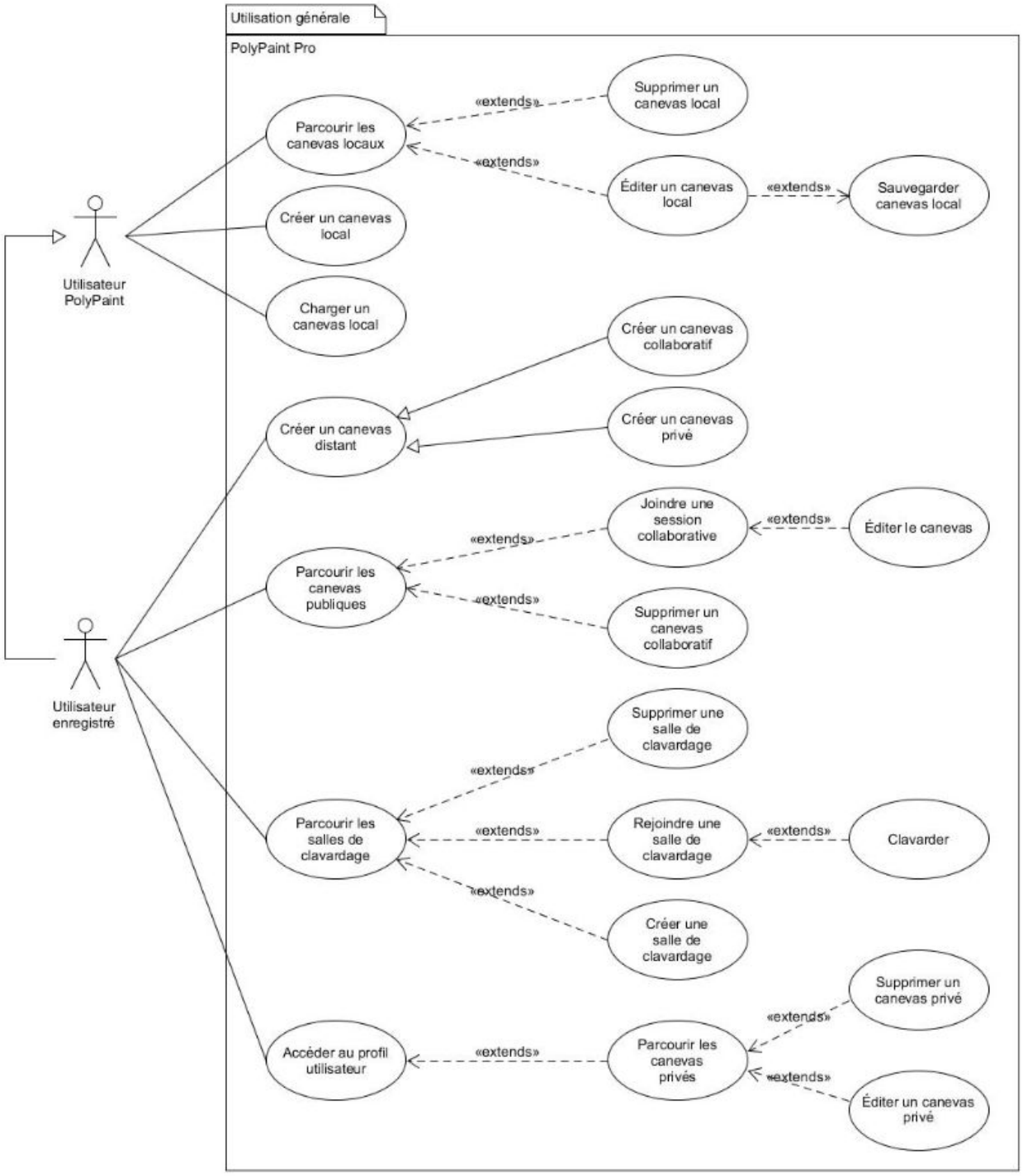

Figure 1: Représentation du cas d'utilisation pour l'utilisation générale de l'application

Le second cas d'utilisation est la création de diagrammes avec l'éditeur PolyPaint Pro.

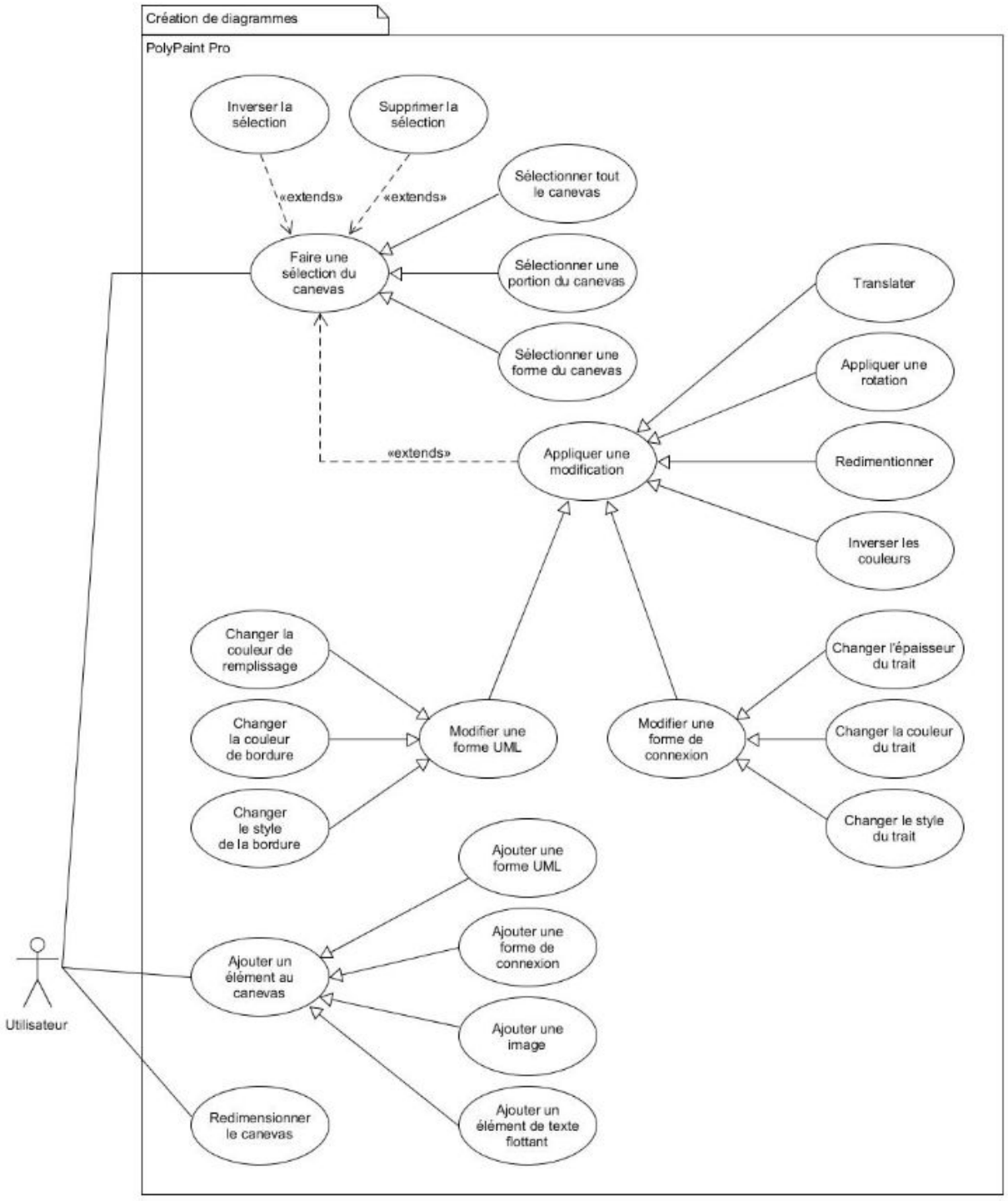

Figure 2: Représentation du cas d'utilisation pour la création de diagrammes

Le troisième cas d'utilisation est l'enregistrement et la connexion à la plateforme.

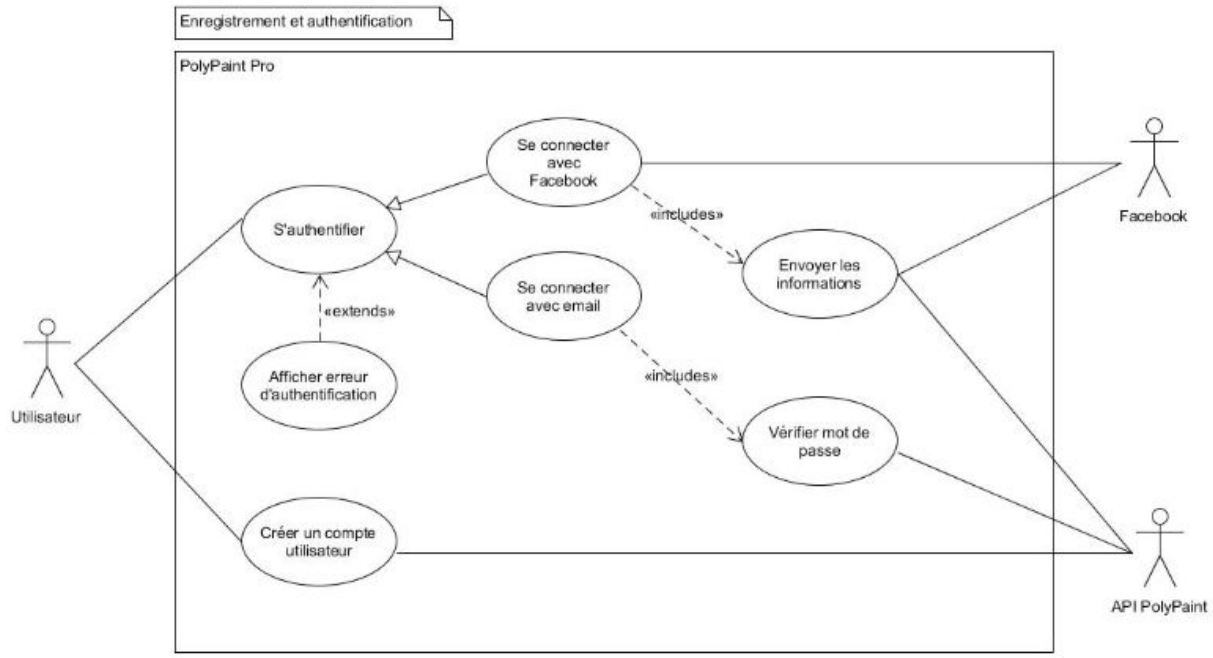

Figure 3: Représentation du cas d'utilisation d'enregistrement et d'authentification

Le dernier cas d'utilisation est clavarder avec d'autres utilisateurs.

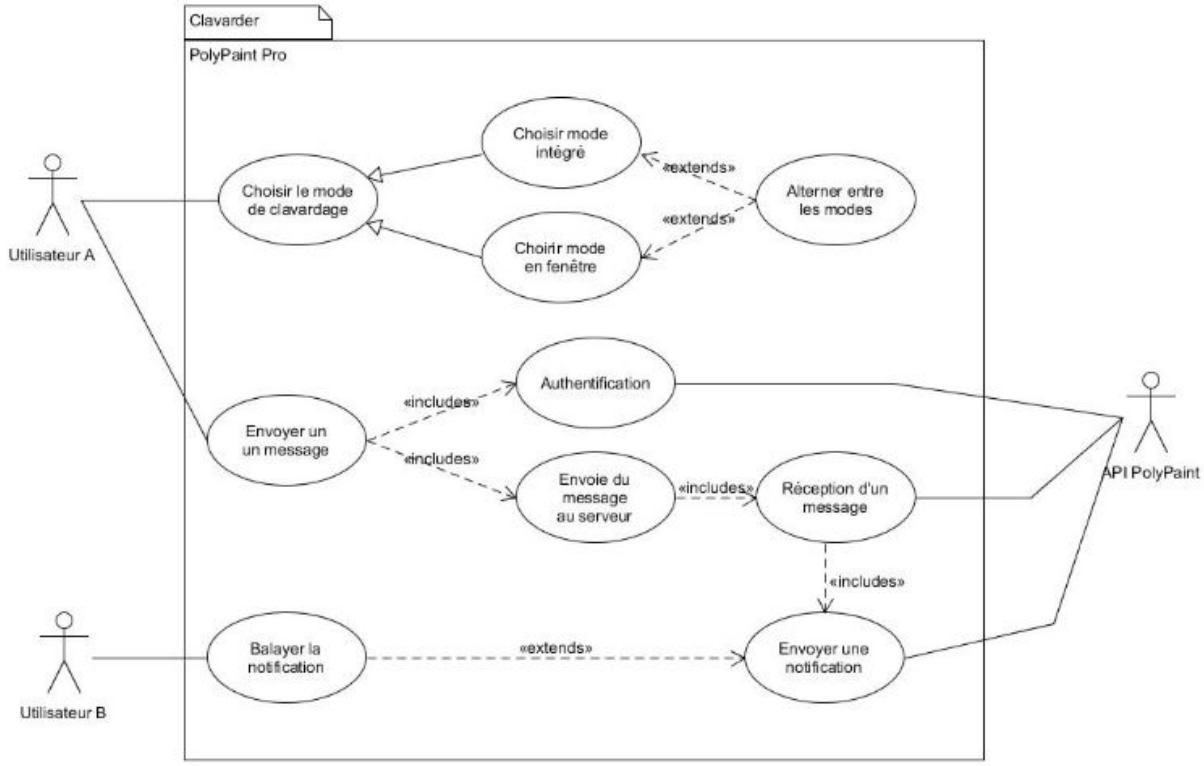

Figure 4: Représentation du cas d'utilisation de clavardage

4. Vue logique

4.1 - Client lourd

Afin de présenter la vue logique du client lourd, nous avons décomposé PolyPaint en trois paquetages principaux, qui contiennent d'autres paquetages. Les trois principaux suivent le modèle architectural du projet, soit MVVM (Modèle, Vue et Vue-Modèle).

Sur l'image ci-dessous, on retrouve, dans l'ordre, View, ViewModel et Model. Chaque paquetage principal est décrit individuellement plus bas, avec des images individuelles afin de mieux voir. L'image suivante sert seulement d'idée générale qui montre les liens entre chaque paquetage principal.

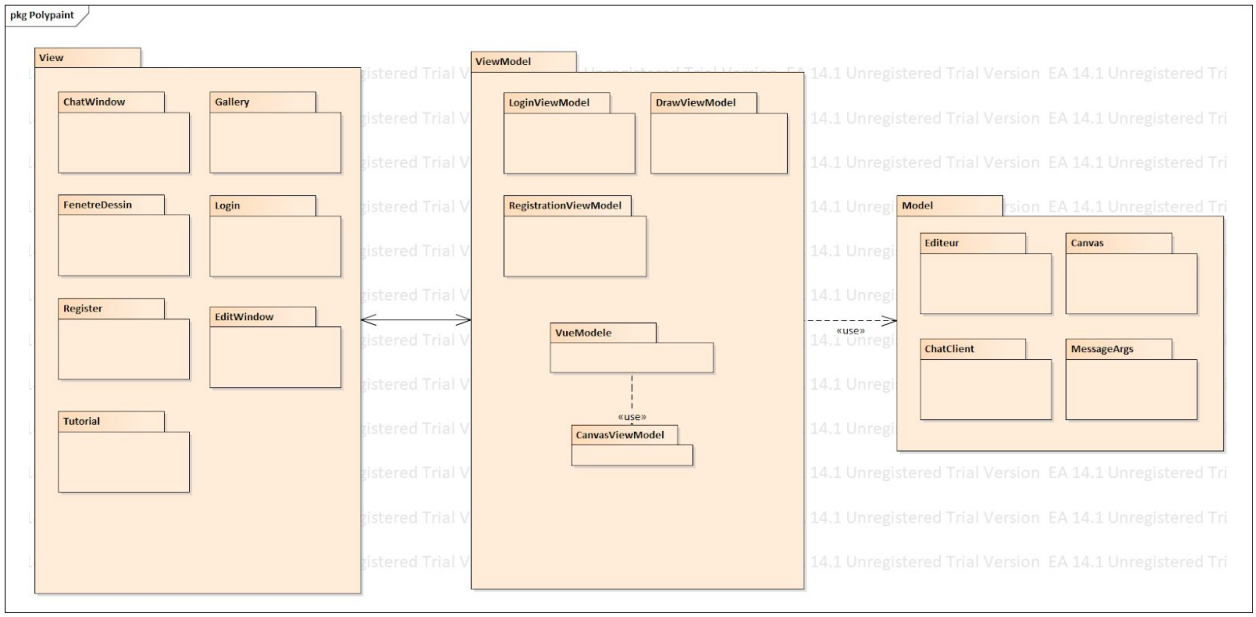

Figure 5: Diagramme de paquetages du client lourd

Le paquetage View regroupe tous les paquetages de vue pour le projet, c'est donc un regroupement de tout ce qui doit être affiché dans le projet.

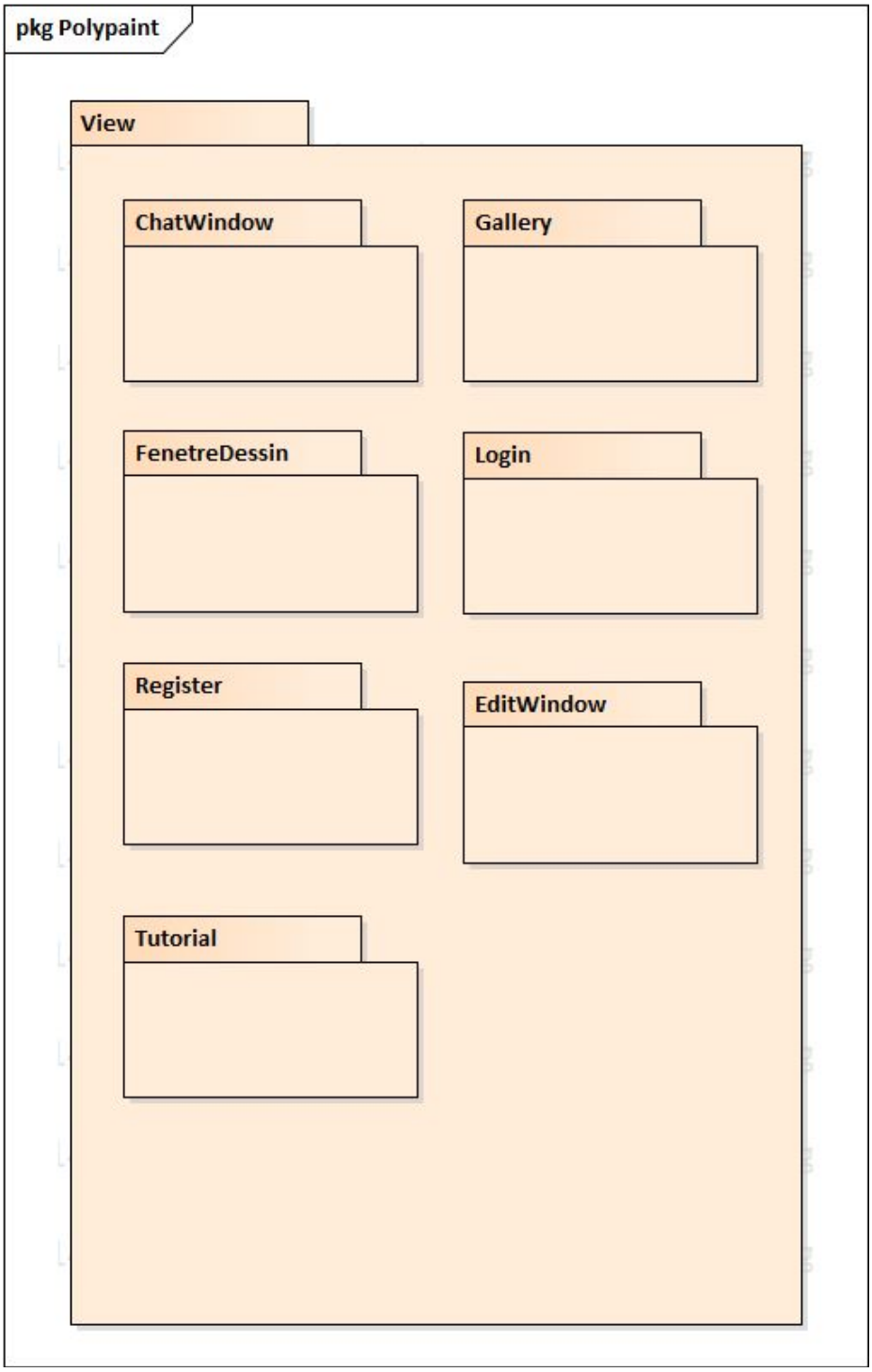

Figure 6 : Diagramme de paquetages de la partie View du client lourd

View
ChatWindow
La vue du ChatWindow est la représentation de la fenêtre de clavardage en mode fenêtré.
FenetreDessin
Le paquetage FenetreDessin est le paquetage responsable de la fenêtre principale, qui contient le canevas et les outils de dessin. Il s'agit du coeur de l'application, puisque l'utilisateur interagit toujours avec celle-ci.
Register
Ce paquetage représente la vue de la fenêtre pour enregistrer un compte sur la base de données.
Login
Ce paquetage représente la vue de la fenêtre permettant de se connecter avec un compte utilisateur.
Gallery
Ce paquetage représente la vue de la galerie d'image qui est hébergée localement ou sur le serveur distant.
Tutorial
La vue du Tutorial représente la fenêtre qui permet d'afficher la séquence d'images formant le tutoriel.
EditWindow
Le paquetage EditWindow représente le bloc qui permet d'éditer du texte sur les divers éléments de dessin.

Figure 7 : Tableau décrivant les divers sous-paquetages du paquetage View

Le second paquetage principal est le ViewModel. Ce dernier regroupe tous les paquetages qui servent à faire le lien entre les données des modèles, et la vue.

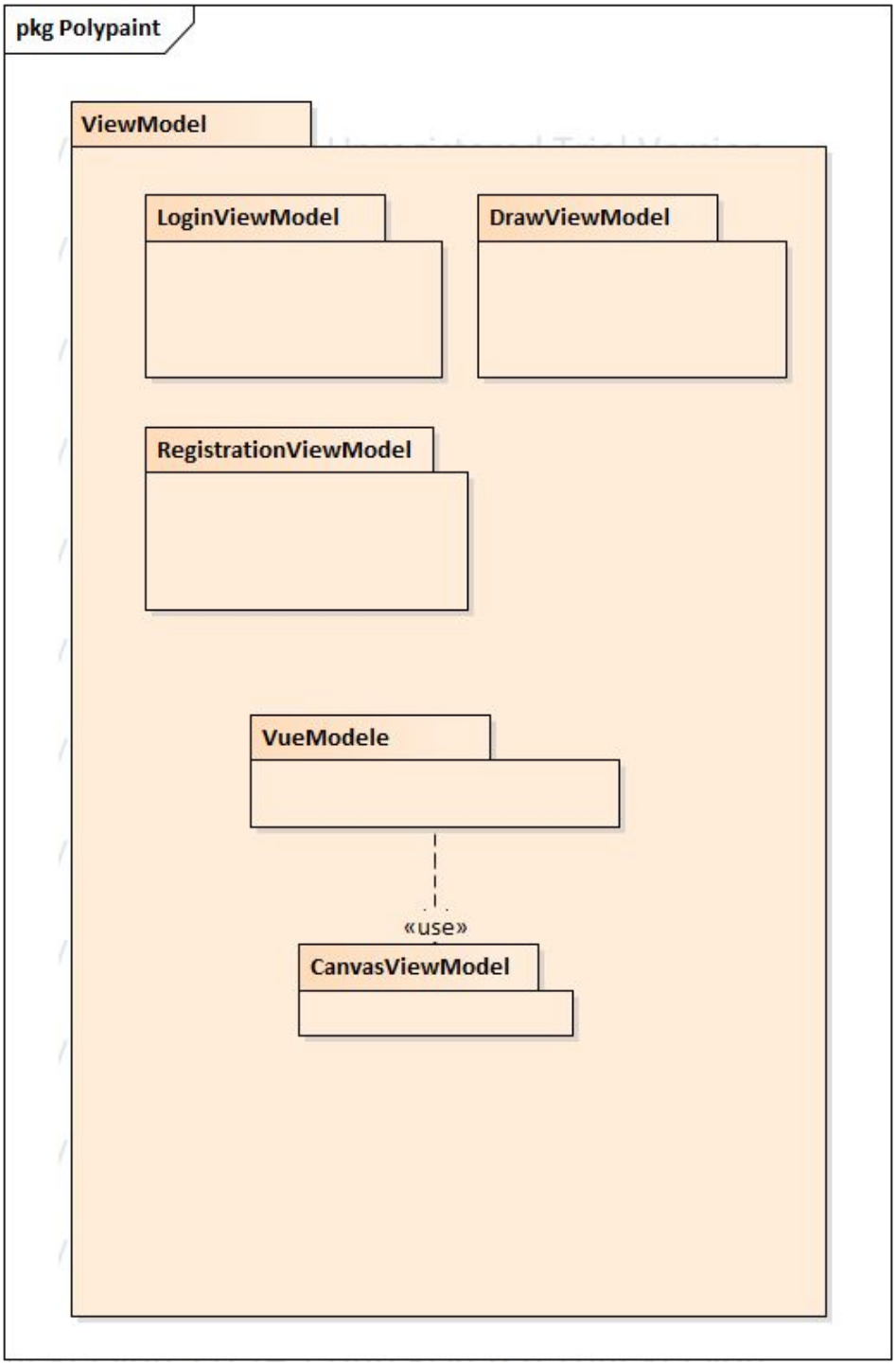

Figure 8 : Diagramme de paquetages de la partie ViewModel du client lourd

ViewModel
LoginViewModel
Ce paquetage sert à utiliser l'information qui permet de se connecter en tant que client.
VueModele
Ce paquetage contrôle l'information de la fenêtre principale, c'est-à-dire la fenêtre de dessin.
CanvasViewModel
Ce paquetage utilise l'information qu'elle récupère du modèle du canevas afin de faire différents affichages par rapport à ceux-ci.
DrawViewModel
Ce paquetage sert à structurer l'information de dessin qui doit circuler entre le client lourd et léger.
TutorialViewModel
Ce paquetage sert à gérer l'utilisation du tutoriel par l'utilisateur et de savoir si l'utilisateur a déjà suivi le tutoriel ou non.

Figure 9 : Tableau décrivant les divers sous-paquetages du paquetage ViewModel

Le troisième paquetage principal est celui de Model. Ce paquetage regroupe toutes les composantes qui nécessitent un modèle afin de transmettre de l'information provenant d'un API.

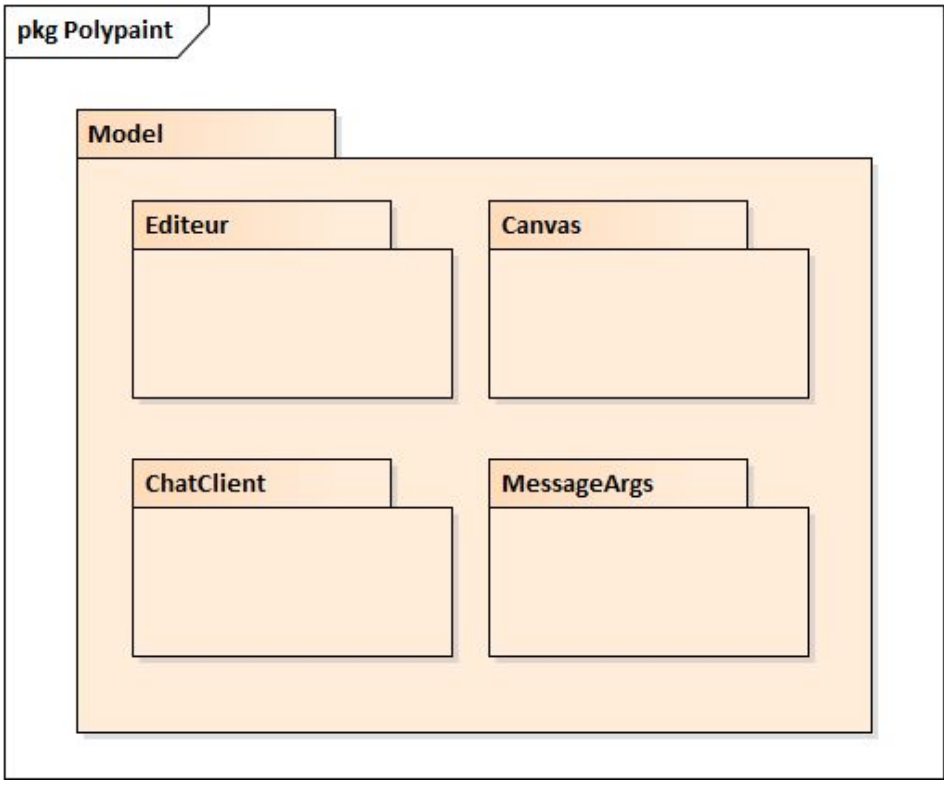

Figure 10 : Diagramme de paquetages de la partie View du client lourd

Model
Editeur
Ce paquetage permet de retenir de l'information par rapport à la fenêtre de dessin principale.
Canvas
Ce paquetage sert à retenir l'information de chaque canevas.
ChatClient
Sert à initialiser et retenir de l'information de connexion de chaque client.
MessageArgs
Ce paquetage permet de retenir les arguments des messages de clavardage (tels que le message, le nom d'utilisateur et le timestamp).

Figure 11 : Tableau décrivant les divers sous-paquetages du paquetage Model

4.2 - Client léger

Un peu comme le client lourd, nous avons décomposé PolyPaint en trois paquetages principaux qui contiennent d'autres paquetages afin de représenter le client léger. Les trois principaux suivent le modèle architectural MVC (soit modèle, vue et contrôleur).

Sur l'image ci-dessous, on retrouve, dans l'ordre, Model, View et Controller. Chaque paquetage principal est décrit individuellement plus bas, avec des images individuelles afin de mieux voir. L'image suivante sert seulement d'idée générale qui montre les liens entre chaque paquetage principal.

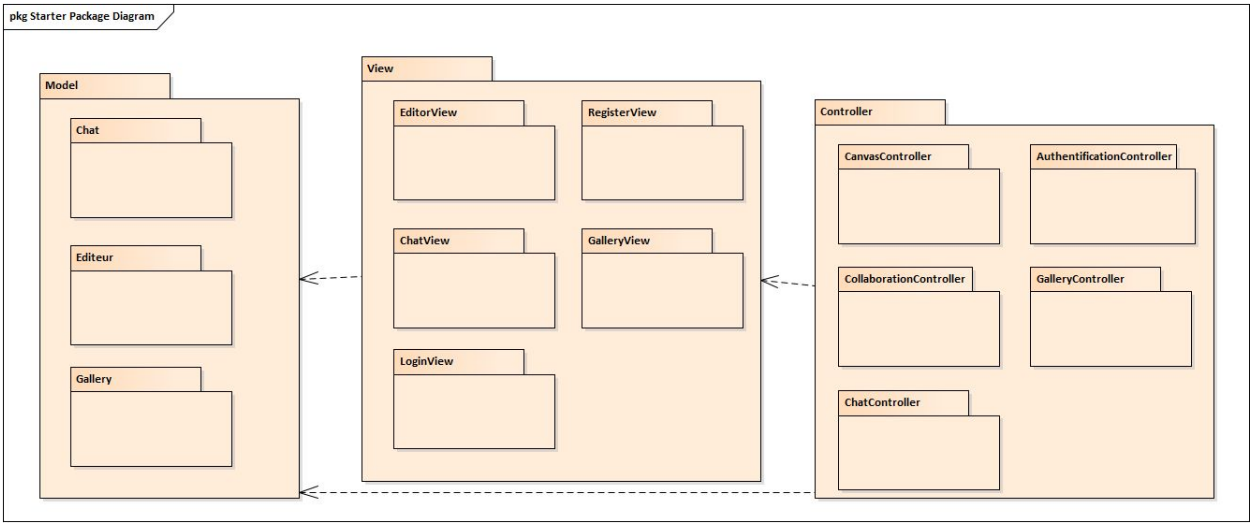

Figure 12: Diagramme de paquetages du client léger

Le premier paquetage principal est celui de Model. Ce paquetage regroupe toutes les composantes qui représentent les états actuels pour les fonctionnalités qui vont être affichées par les vues.

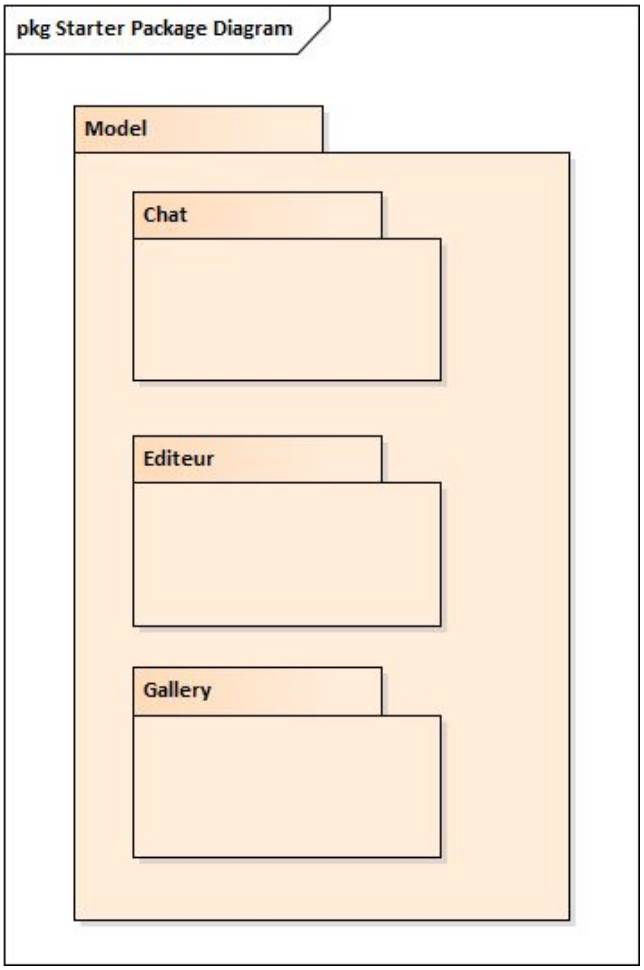

Figure 13 : Diagramme de paquetages de la partie Model du client léger

Model
Chat
Ce paquetage représente les informations de l'état actuel pour le chat.
Editeur
Ce paquetage représente toutes les informations reliées à l'état du canevas.
Gallery
Ce paquetage représente les informations de la galerie qui pourront être affichées par la galerie.

Figure 14 : Tableau décrivant les divers sous-paquetages du paquetage Model

Le second paquetage principal est celui de View. Ce paquetage regroupe toutes les composantes qui représentent les vues de l'application.

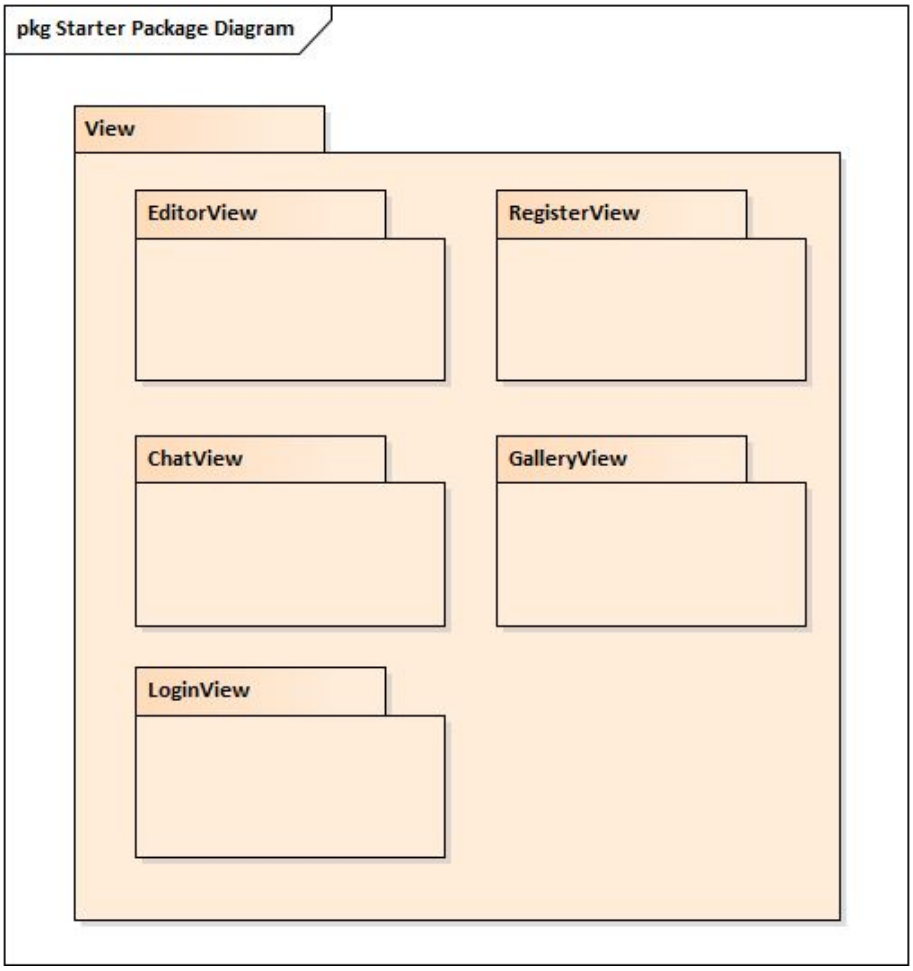

Figure 15 : Diagramme de paquetages de la partie View du client léger

View
EditorView
Ce paquetage représente les informations liées à la vue de l'éditeur (donc du canevas).
RegisterView
Ce paquetage représente la vue de la fenêtre d'enregistrement.
ChatView
Ce paquetage représente les informations pour la vue du clavardage.
GalleryView
Ce paquetage sert à représenter la galerie d'images (et le profil utilisateur).
LoginView
Ce paquetage représente la fenêtre liée à la connexion d'un utilisateur.

Figure 16 : Tableau décrivant les divers sous-paquetages du paquetage View

Le troisième paquetage principal est celui de Controller. Ce paquetage regroupe toutes les composantes qui servent d'intermédiaire entre les vues et les modèles.

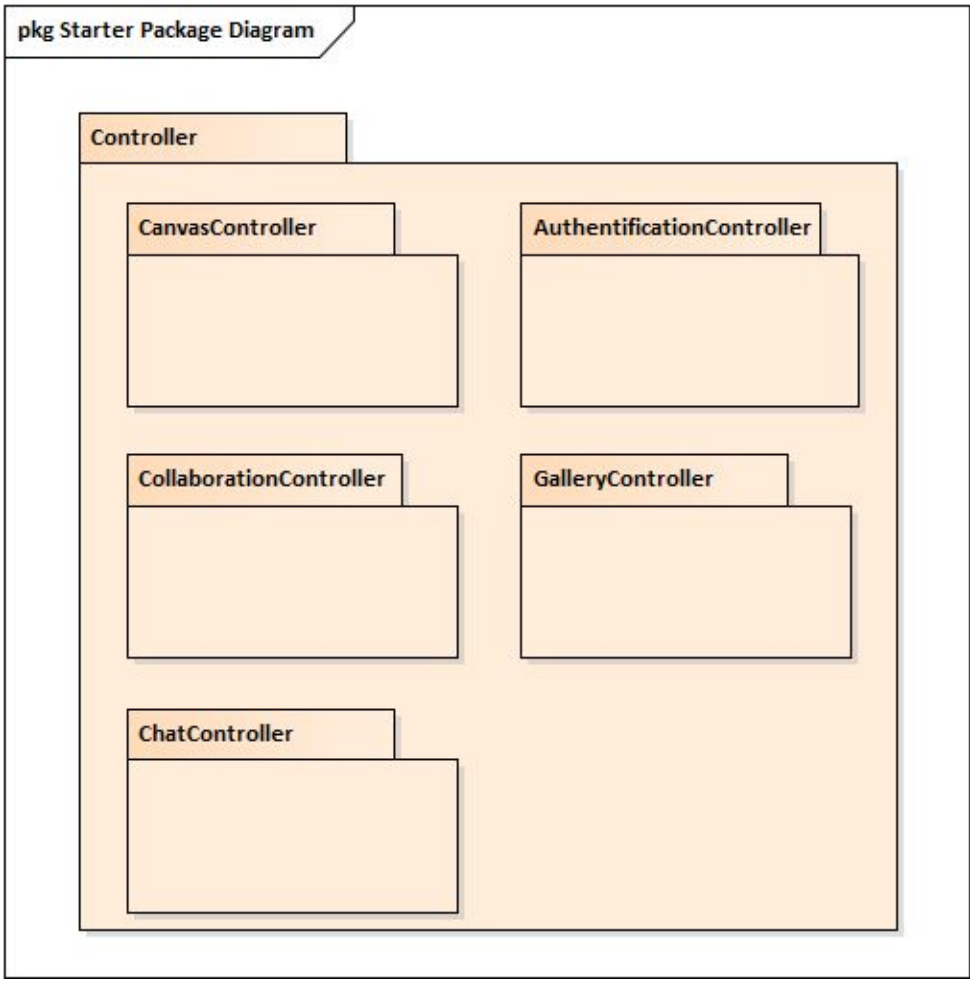

Figure 17 : Diagramme de paquetages de la partie Controller du client léger

Controller
CanvasController
Le contrôleur du canevas sert d'intermédiaire entre les modèles qui servent à l'éditeur et leurs vues associées.
AuthentificationController
Le paquetage d'authentification sert à communiquer de l'information avec la vue d'enregistrement et la vue de connexion.
CollaborationController
Le paquetage de collaboration sert à gérer et envoyer l'information servant à la collaboration entre le client lourd et le client léger.
GalleryController
Le paquetage de galerie sert à gérer l'information de celle-ci et sert d'intermédiaire entre sa vue et son modèle.
ChatController
Le paquetage de clavardage sert à gérer l'information de celui-ci et sert d'intermédiaire entre ses vues et son modèle.

Figure 18 : Tableau décrivant les divers sous-paquetages du paquetage Controller

4.3 - Serveur

Le diagramme de paquetage suivant représente la vue logique du serveur.

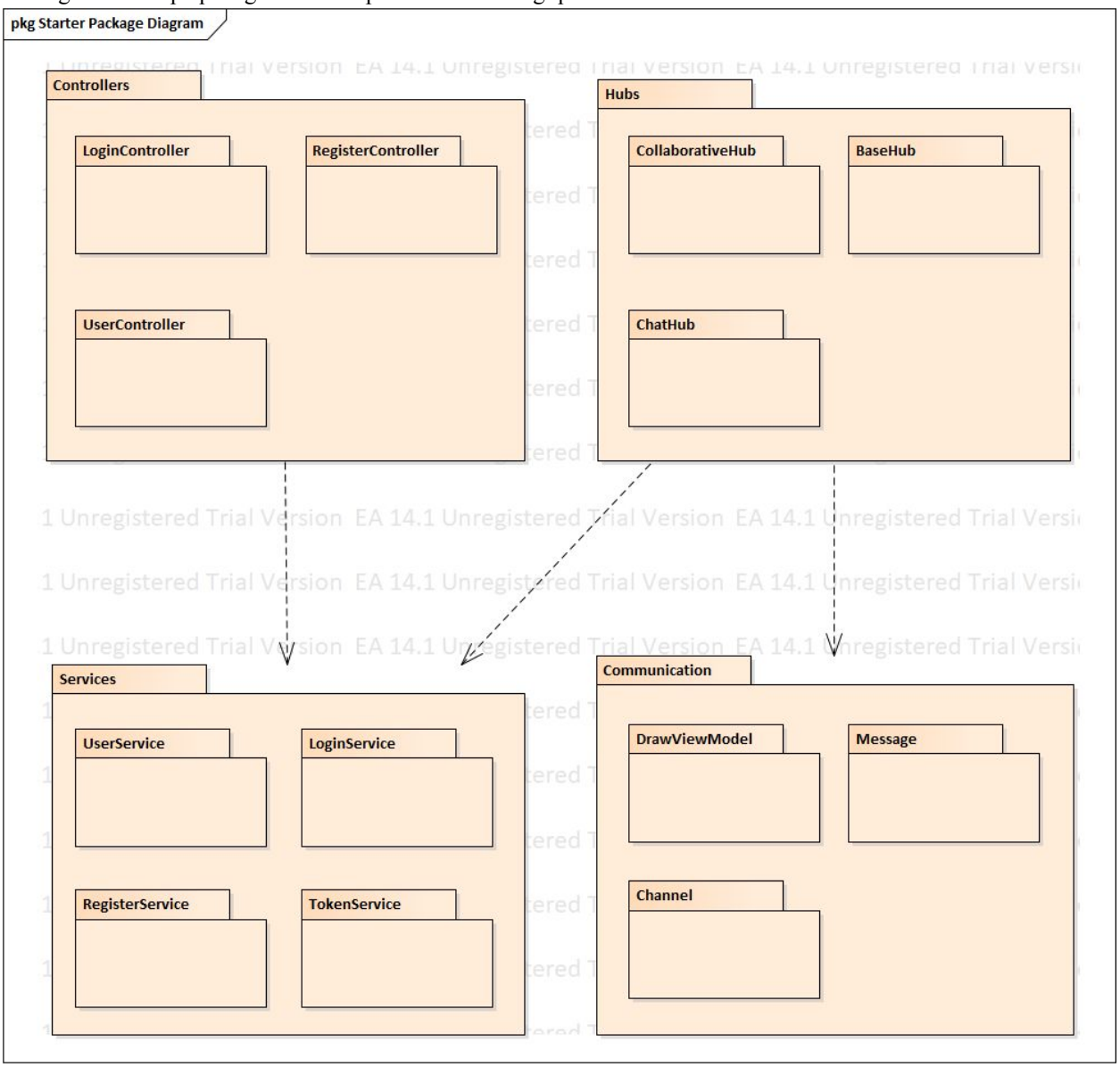

Figure 19 : Diagramme de paquetages pour le serveur

Controllers
Ce paquetage regroupe divers paquetages servant à faire les requêtes HTTP.
LoginController
Ce paquetage sert à faire les requêtes HTTP liées au service de connexion d'utilisateurs.
RegisterController
Ce paquetage sert à faire les requêtes HTTP liées au service de création de nouveaux comptes usagers.
UserController
Ce paquetage sert à faire les requêtes HTTP liées au service qui gère l'information des utilisateurs.
Services
Ce paquetage regroupe divers paquetages de services du serveur.
LoginService
Ce paquetage représente un service permettant de gérer la connexion d'un utilisateur.
TokenService
Ce paquetage représente un service qui permet de gérer la génération de jetons pour les utilisateurs.
RegisterService
Ce paquetage représente un service qui gère la création de nouveaux comptes usagers.
UserService
Ce paquetage représente un service qui permet de gérer toute l'information liée à un utilisateur.
Hubs
Ce paquetage regroupe divers paquetages qui servent à gérer des requêtes.
CollaborativeHub
Ce paquetage représente la partie qui gère les requêtes de la collaboration entre divers clients.
BaseHub
Ce paquetage représente la partie qui sert à gérer les requêtes pour les fonctionnalités de base de l'application.
ChatHub
Ce paquetage représente la partie qui sert à gérer les requêtes pour le chat.

Communication
Ce paquetage regroupe plusieurs paquetages servant à gérer la communication entre les clients lourds et légers.
DrawViewModel
Ce paquetage sert à structurer l'information de dessin qui doit circuler entre le client lourd et léger.
Message
Ce paquetage représente les différents types de messages qui peuvent circuler entre les clients.
Channel
Ce paquetage représente les canaux d'échange d'informations.

Figure 20 : Tableau décrivant les divers paquetages du serveur

5. Vue des processus

Afin de décrire le système en termes d'interactions entre les différents processus significations, nous avons déterminé les tâches principales, et avons construit un diagramme de séquence pour chacune de ses tâches. Sur ces diagrammes, nous retrouvons les interactions entre un utilisateur quelconque, son application locale, le serveur distant ainsi que les autres utilisateurs en ligne.

Le premier processus représenter par un diagramme de séquence est l'ajout d'un élément sur le canevas (en ligne ou hors-ligne).

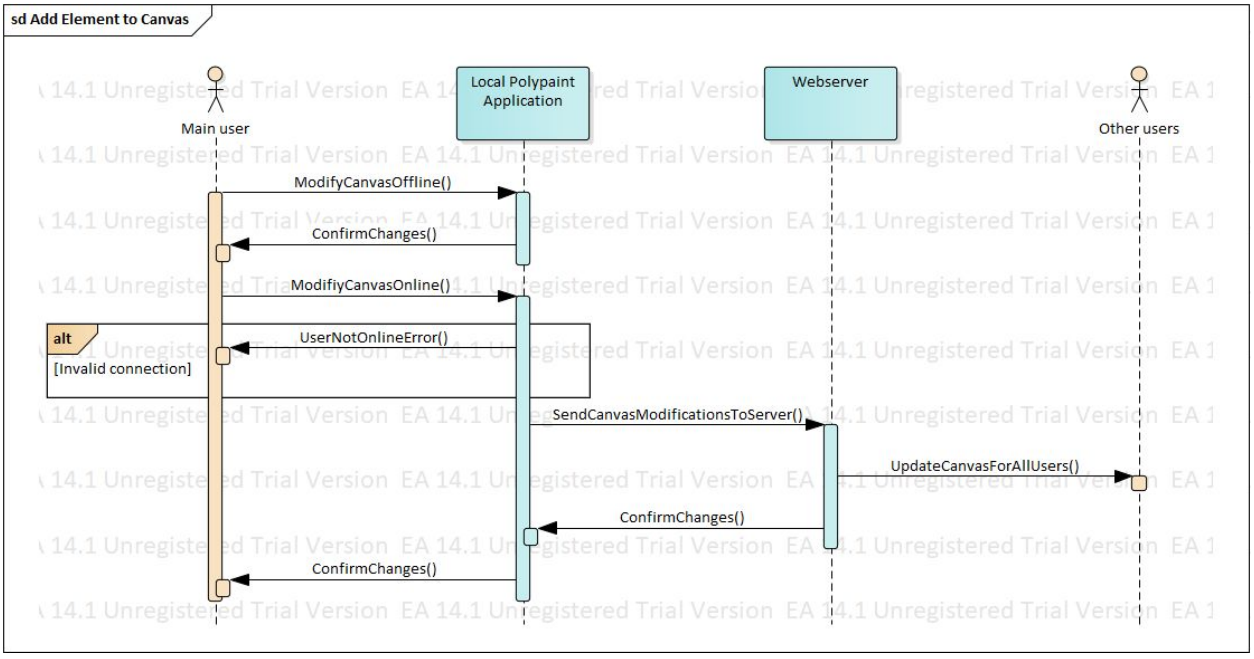

Figure 21 : Diagramme de séquence pour l'ajout d'un élément au canevas

Le second processus important est l'authentification sur le serveur et la connexion grâce à ce dernier.

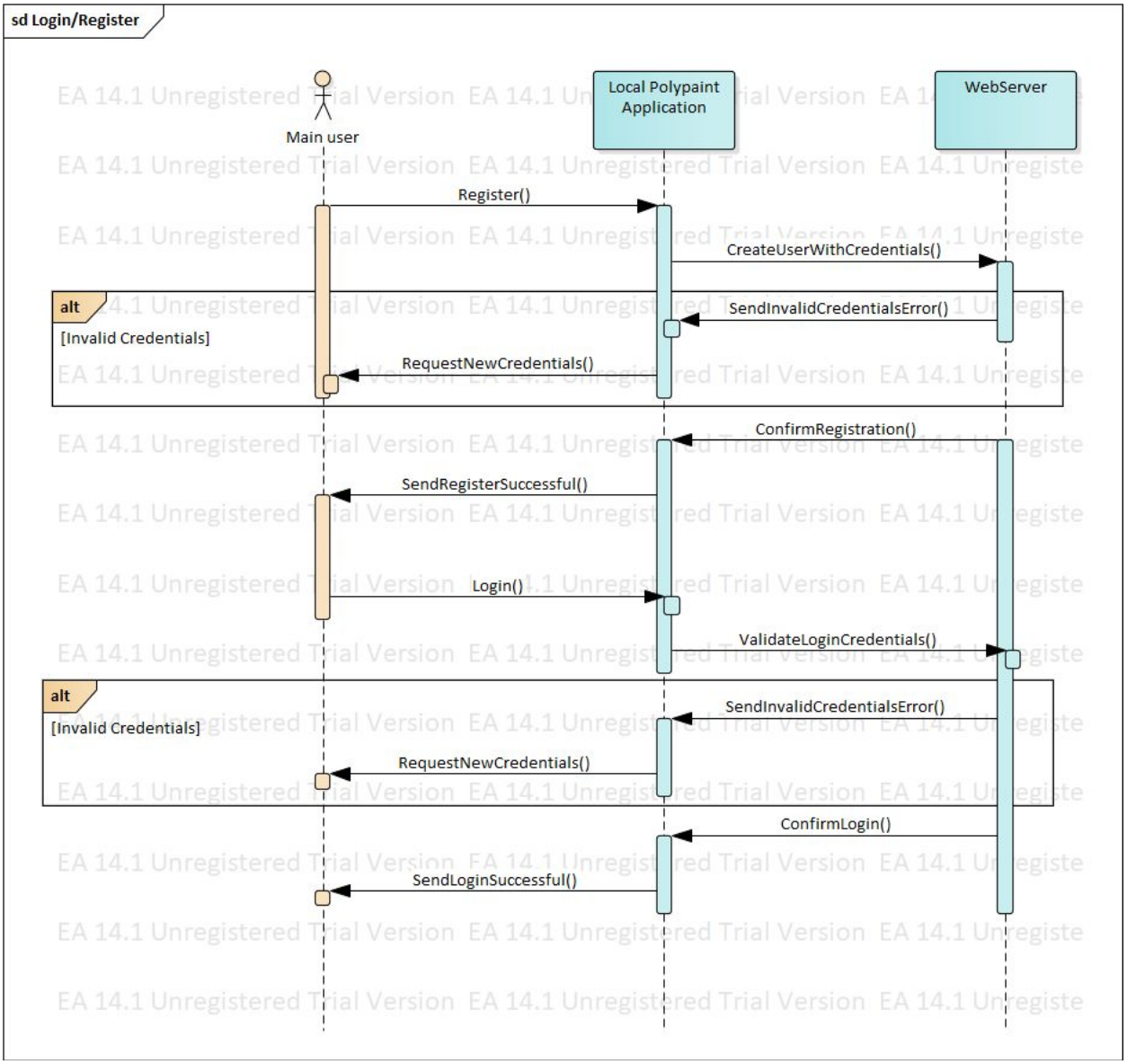

Figure 22 : Diagramme de séquence pour l'enregistrement d'un compte et la connexion à partir de celui-ci

Le troisième processus que nous avons jugé pertinent à présenter est le processus de clavardage.

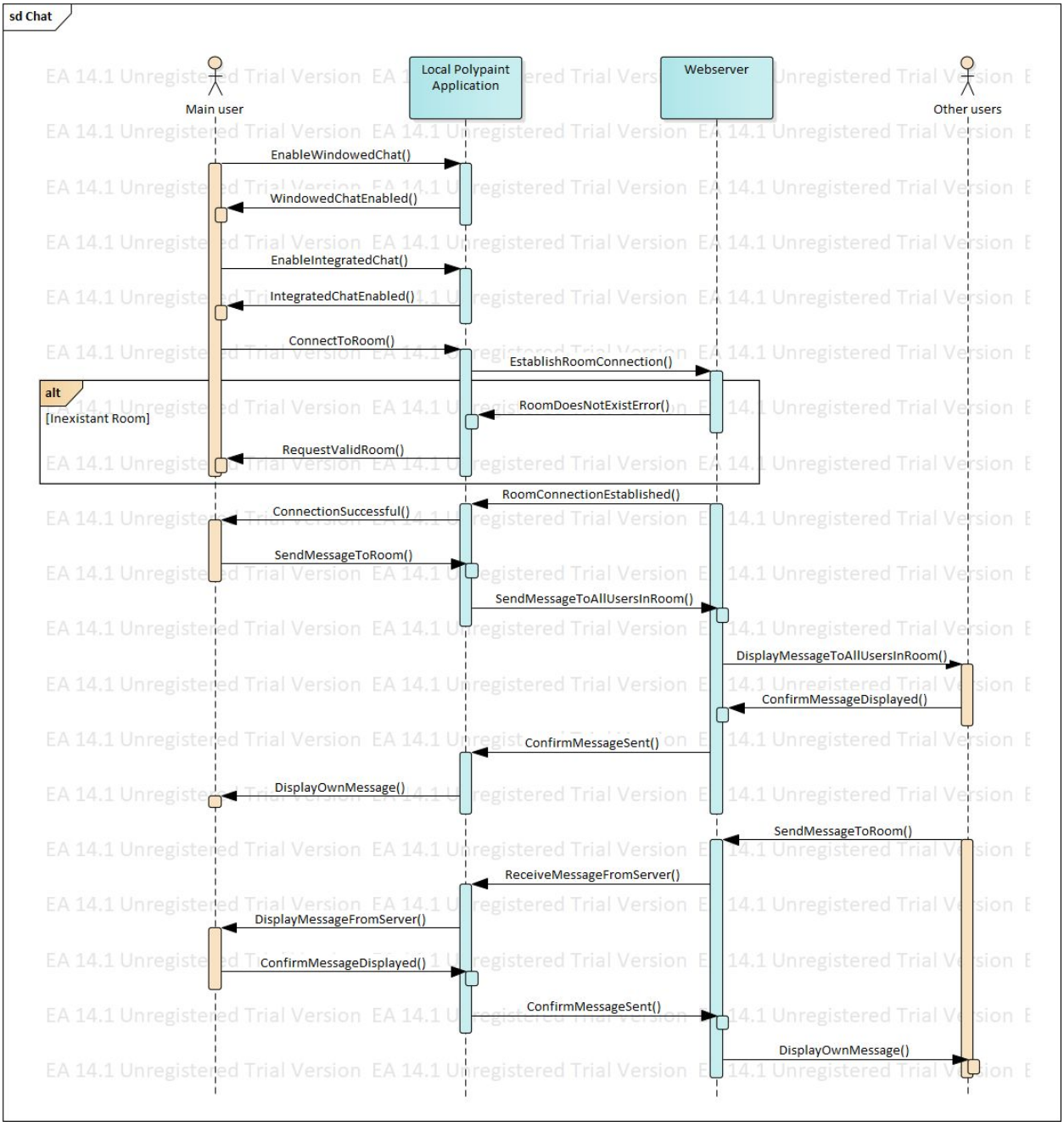

Figure 23 : Diagramme de séquence pour le clavardage

Le quatrième processus important est la sauvegarde et le chargement de fichier (de manière locale et distante).

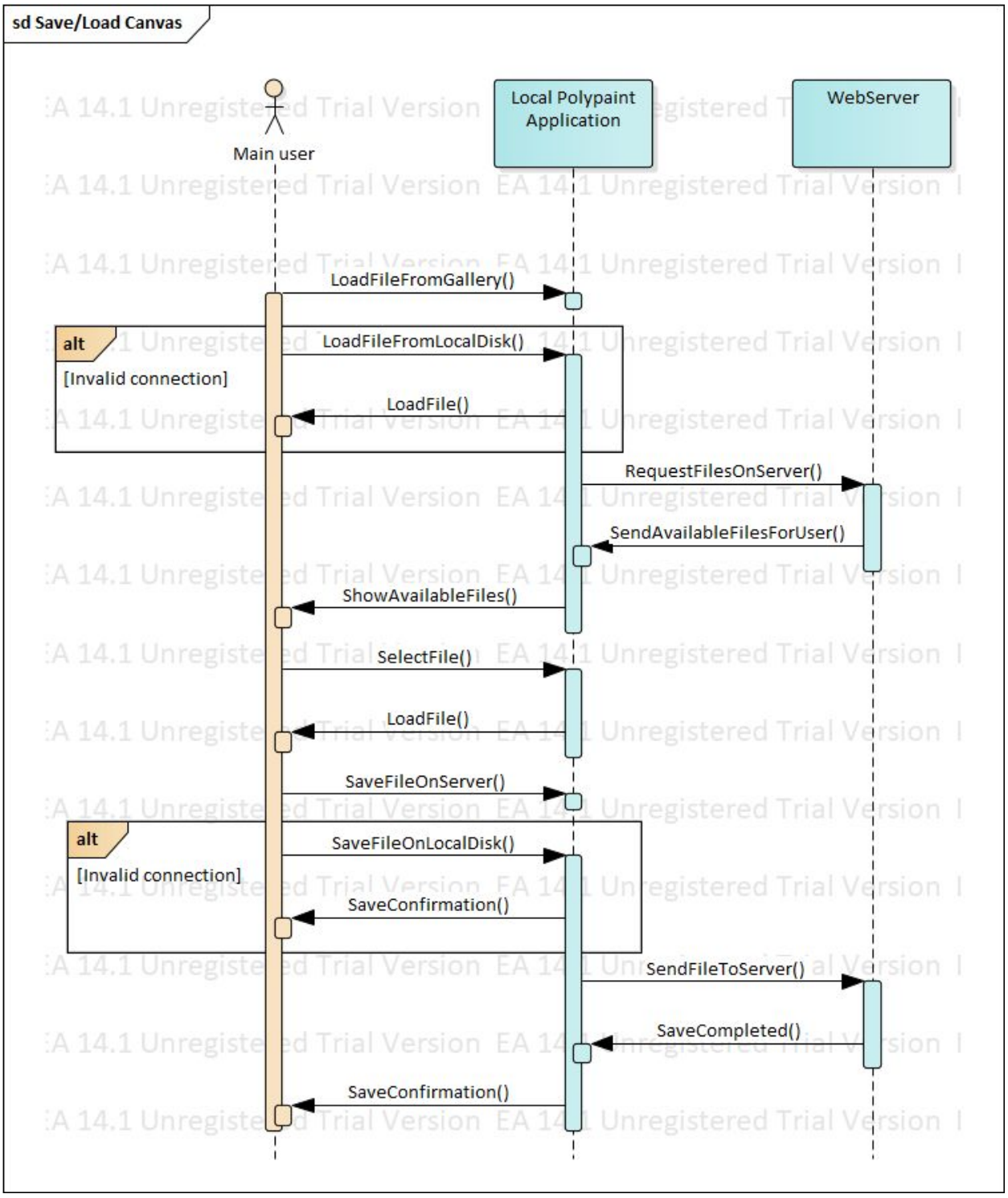

Figure 24 : Diagramme de séquence pour la sauvegarde et le chargement d'un canevas

6. Vue de déploiement

Cette section décrit une configuration de matériel physique complète pour le déploiement de Poly Paint.

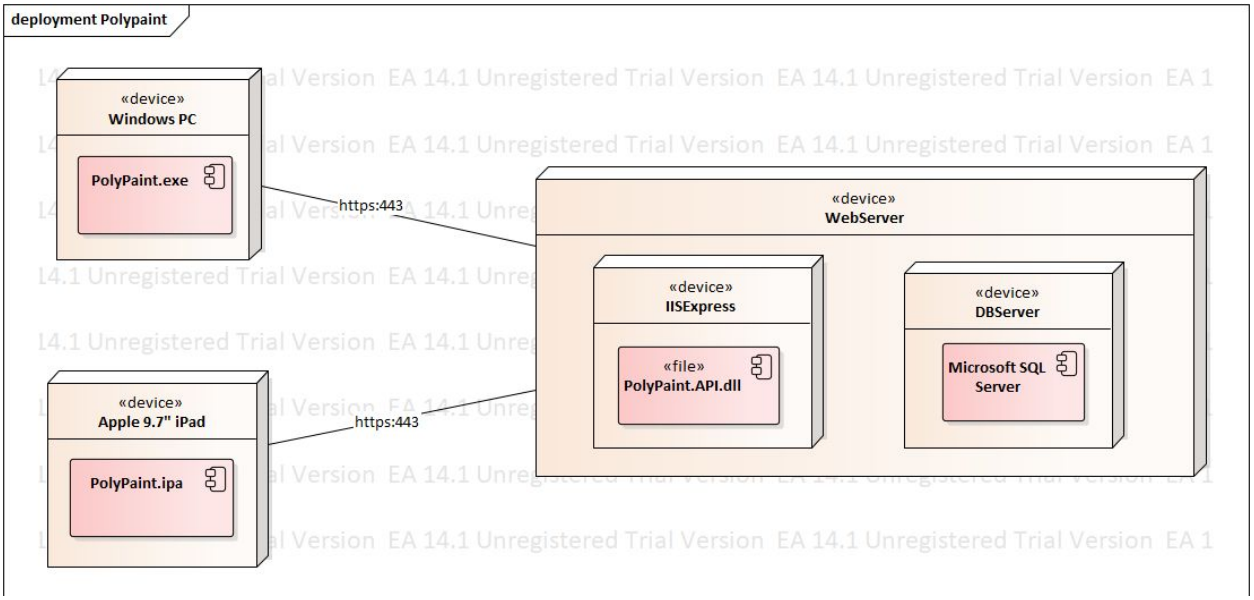

Figure 25 : Diagramme de déploiement de l'application

7. Taille et performance

Tout d'abord, la performance doit respecter les critères énoncés dans le SRS. Ces critères sont que le délai d'authentification doit être inférieur à 2 secondes. De plus, les messages échangés dans la messagerie instantanée doivent avoir un délai inférieur à 1 seconde, alors que le temps réel des dessins devrait avoir un délai inférieur à 2 secondes. La dernière exigence de performance pour le SRS est que l'ouverture d'un dessin lors du chargement devrait être inférieur à 5 secondes. Ces exigences mises toutes à minimiser les délais afin de réduire la latence entre les utilisateurs, et de rendre l'expérience utilisateur plus agréable de manière globale.

Pour ce qui est des ressources matérielles, le client lourd doit minimiser celles-ci afin que les tâches en arrière-plan sur l'appareil soient en mesure de continuer leur déroulement normal (on ne s'attend pas à ce que l'ordinateur s'exécute parfaitement lors de l'exécution de PolyPaint s'il y avait des difficultés avant le lancement de l'application). Ces conditions s'appliquent de manière encore plus importante sur le client léger, puisque ce dernier possède moins de mémoire vive que la majorité des ordinateurs de nos jours. Même si le stockage n'est plus vraiment un problème de nos jours, on souhaite minimiser la taille de nos applications, puisqu'il s'agit d'une application relativement simple, on devrait avoir des fichiers de tailles très raisonnables. On souhaite donc que la mémoire vive utilisée n'excède jamais 500 Mo, et que l'application nécessite moins de 200 Mo une fois installés (en excluant la taille des canevas sauvegardés localement).