

GeoPoppy - Guide d'installation et de configuration

Cette documentation a été créée à la suite d'un projet financé par l'AFD durant lequel CartONG a accompagné la mise en place d'un outil de suivi et d'évaluation des parcelles agricoles basé sur GeoPoppy en Côte d'Ivoire pour l'ONG Etc Terra/Rongead.

Elle reprend en grande partie des éléments tirés des documentations de [GeoPoppy](#), [LizMap](#), [QGIS](#), [pgAdmin](#) et [Docker](#) et a simplement vocation à proposer un pas-à-pas documenté de l'installation et de la configuration de l'outil en insistant sur quelques fonctionnalités importantes pour la répliquabilité de cette solution sur d'autres projets similaires.

/!\ Malgré la volonté de créer une documentation qui permette au plus grand nombre d'utiliser cette solution, une connaissance minimum de **QGIS**, **PostgreSQL/PostGIS**, **Putty** et des **commandes Linux** est fortement recommandée pour être à l'aise dans la création et la customisation de son projet.

I - Matériel

La particularité de GeoPoppy par rapport à une application de collecte classique sur smartphone ou tablette est qu'elle nécessite l'utilisation (et donc l'achat) de RaspberryPi. Toute l'installation se réalise sur ces micro-ordinateurs et le smartphone ou la tablette ne sont que le support (l'écran) qui permet de l'utiliser.

L'avantage d'une telle solution est de profiter de la puissance et des logiciels disponibles sur un ordinateur directement sur une tablette sans être limité par les fonctionnalités des différentes applications classiques.

En terme matériel, il faut donc se doter, pour chaque collecteur de :

1. Un **Raspberry Pi 3** avec son boîtier.
2. Une carte **Micro-SD** (*nous avons utilisé des 64 Go mais une 16Go peut déjà largement suffire en fonction de la taille des jeux de données utilisés*).
3. Un **Power-Pack** pour alimenter le Raspberry Pi sur le terrain (*nous avons utilisé des 10000 mAh qui étaient largement suffisant en terme d'autonomie*).
4. Une **tablette tactile** (*nous avons opté pour des Samsung Galaxy Tab E avec housse de protection*).



II - Installation de GeoPoppy

L'installation de GeoPoppy sur un Raspberry Pi 3 est documentée ici :

https://github.com/jancelin/geo-poppy/blob/master/install/README_install_geopoppy.md. Suivez pas-à-pas les étapes et vous devriez voir apparaître la carte démo "messicoles" en accedant à l'adresse <http://172.24.1.1> .

Si vous êtes sous Windows, nous préconisons l'utilisation de Win32 Disk Imager ou de Etcher pour flasher la Micro SD et de Putty pour la connexion en SSH.

III - Construction de la base de données

III.1 - Créer une base de données vierge et l'ouvrir dans QGIS

Un fois GeoPoppy installé sur un Raspberry Pi, il est temps de construire le modèle de données qui sera utilisé lors de la collecte et dupliqué à tous les Raspberry Pi si vous souhaitez que plusieurs collecteurs travaillent sur le même.

1. Téléchargez et installez [PostgreSQL](#) et [pgAdmin 3](#) sur votre ordinateur.
2. Connectez votre ordinateur au réseau Wi-Fi **GeoPoppy_Pi3** et ouvrez **pgAdmin**.
3. Créez un **Nouvel enregistrement serveur** avec les paramètres suivants:

- Nom : **geopoppy**
- Hôte : **172.24.1.1**
- Port : **5432**
- Nom d'utilisateur : **docker**
- Mot de passe : **docker**

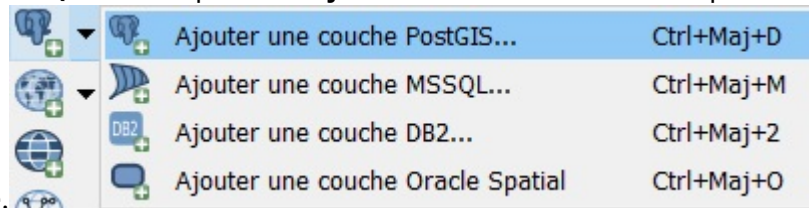
The image shows a 'New Server Registration' window with the following fields and values:

Field	Value
Name	geopoppy
Host	172.24.1.1
Port	5432
Service	
Maintenance DB	postgres
Username	docker
Password	••••••
Store password	<input checked="" type="checkbox"/>
Colour	
Group	Servers

1. Ouvrez cette nouvelle connexion pour voir apparaître les bases de données utilisées dans la démo. Créez un nouveau rôle dans **Roles de connexion** et attribuez lui tous les privilèges (superuser). Ce rôle sera le propriétaire de la base de données de votre projet, notez bien le nom et le mot de passe que vous lui accordez, nous en aurons besoin plus tard.
2. Créez une **nouvelle base de données**, nommez-la comme vous le souhaitez et choisissez comme propriétaire le rôle précédemment créé.
3. Ajoutez les **extensions** *postgis* et *postgis_topology* à l'aide des requêtes SQL suivantes : `CREATE EXTENSION POSTGIS` `CREATE EXTENSION POSTGIS_TOPOLOGY`
4. Créez un **nouveau schéma** portant le nom du rôle précédemment créé et choisissez ce rôle comme propriétaire.

Vous avez ainsi créé une base de donnée vierge dans pgAdmin, nous allons maintenant la remplir directement depuis QGIS.

1. Installez QGIS sur votre ordinateur (choisissez minimum la version 2.0).
2. Une fois installé, ouvrez QGIS et cliquez sur **Ajouter une couche PostGIS** puis sur **Nouveau** dans



la fenêtre qui s'ouvre.

3. Créez une nouvelle connexion en utilisant les paramètres suivants :

- Nom : **geopoppy**
- Hôte : **172.24.1.1**
- Port : **5432**
- Base de données : **Nom que vous avez donné à la base de données dans pgAdmin**
- Mode SSL : **permet**
- Authentification : **rentrez le nom d'utilisateur que vous avez créé dans pgAdmin et son mot de passe. Cochez impérativement *Enregistrer* pour sauvegarder les identifiants de connexion.**

III.2 - Créer les couches dans QGIS

Une fois la connexion créée, rendez-vous dans le menu **Gestionnaire de bases de données** de QGIS. Ouvrez le menu **PostGIS** pour voir votre base de données puis sélectionnez le schéma que vous avez créé. C'est à partir d'ici que vous pouvez commencer à créer les couches que vous voulez utiliser pour votre projet.



*//\ **Rappel concernant LizMap** : Seules les couches et tables que vous souhaitez pouvoir être éditables par l'utilisateur ont besoin d'être créées dans PostGIS. Pour les couches qui ne seront pas éditées (celles qui composent votre fond de carte par exemple), vous pouvez utiliser de simples **shapefiles** pour vous simplifier la tâche.*

Pour créer une nouvelle table dans la base de données, cliquez sur le menu **Table** puis **Créer une table**. La fenêtre qui s'ouvre vous permet de créer une nouvelle table vierge et de renseigner ses différents champs. Construisez les tables dont vous avez besoin en gardant en tête ces conseils :

- Faire attention au **type de champ**. Réfléchissez à chaque fois que vous créez un nouveau champ au type de valeur que l'utilisateur va rentrer (texte, nombre, date etc.)
- Créez toujours un champ **id** de type **serial** et renseignez le comme **clé primaire**.
- Si vous souhaitez mettre en place un système de synchronisation entre plusieurs Raspberry Pi,

ajoutez pour chaque couche un champ **uuid** de type **text** qui servira d'identifiant unique (*voir partie Outils d'edition et Synchronisation*).

- Si vous souhaitez connaître l'utilisateur et l'heure de création et d'édition, créez pour chaque couche des champs **editepar** et **creepar** de type **text** et des champs **editedate** et **creedate** de type **date** et suivez les instructions énoncées dans la partie **Customisation de Lizmap / Javascript**.
- Choisissez soigneusement les champs qui peuvent être nuls.
- Cochez **Créer une colonne géométrique** si vous souhaitez créer une couche de point, lignes ou polygones. Cochez systématiquement **Créer un index spatial** pour améliorer les performances.
- Pour certains champs, vous voudrez sûrement que l'utilisateur ait le choix parmi une liste de valeurs dans un **menu déroulant** (dans notre cas, pour la couche des parcelles : type de culture, nom du propriétaire etc.). La configuration du menu déroulant ne se fait pas lors de la création de la couche, mais **si vous souhaitez que la liste des choix soit évolutive** (c'est à dire que l'utilisateur puisse ajouter lui même des valeurs, ce qui est notre cas pour les propriétaires des parcelles), il faut penser à **créer une table sans géométrie pour y stocker cette liste**. Pour l'exemple des propriétaires, nous avons donc créé une table nommée **propriétaires** comprenant les champs **id (serial)**, **nom (text)**, les champs de métadonnées expliqués plus haut ainsi que quelques informations supplémentaires comme le genre ou le numéro de téléphone. La couche de parcelle dispose elle d'un champ **proprietaire** qui sert à rentrer le nom du propriétaire, en lien avec la liste de la table **propriétaire**. Ainsi, l'utilisateur doit d'abord rentrer le propriétaire dans la table des propriétaires avant de pouvoir s'en servir dans la table des parcelles ce qui permet à chaque propriétaire d'avoir un identifiant unique et d'avoir un vrai lien entre les deux tables.
- Si vous souhaitez utiliser des couches de polygone comme ce fut notre cas pour les parcelles, nous recommandons de créer une couche de points que vous pouvez appeler **sommets** qui serviront de points GPS temporaires. Lors de la collecte, enregistrez les sommets des parcelles comme des points individuels en les numérotant dans l'ordre puis utilisez les comme repères pour tracer le polygone final. Nous avons choisi cette méthode car le tracage des polygones peut être une opération complexe pour certains utilisateurs sur la tablette et qu'à la moindre erreur il est très difficile de revenir en arrière sans tout retracer. Lorsque certains parcelles peuvent prendre plus de 30min pour en faire le tour, vous voulez être sûrs de ne pas avoir à revenir sur vos pas !

IV - Construction du projet QGIS

Une fois que vous avez créé l'ensemble des couches éditables dans le **Gestionnaire de bases de données**, il est temps de les ouvrir dans QGIS pour créer le projet qui sera ensuite consultable via LizMap.

/!\ La construction d'un projet QGIS pour Lizmap est expliquée en détail dans la documentation officielle de LizMap ici : <https://docs.3liz.com/fr/publish/index.html>. Cette partie détaillera simplement certains points spécifiques comme l'ouverture des couches PostGIS, un supplément d'information sur les outils d'édition et sur le plugin LizMap.

IV.1 - Ouvrir des couches

Dans QGIS, ouvrez un nouveau projet et sauvegardez-le tout de suite sur votre ordinateur dans un dossier qui contiendra tous les fichiers utiles au fonctionnement de l'application (*par exemple, créez un dossier **geopoppy** sur votre bureau et sauvegardez le projet*).

Cliquez sur **Ajouter une couche PostGIS** et choisissez votre connexion dans le menu déroulant des connexions. Pensez à cocher **Lister les tables sans géométrie** si vous souhaitez ajouter des tables qui n'ont pas de géométrie puis cliquez sur **Connecter**. Sélectionnez les tables à ajouter au projet et cliquez sur **Ajouter**.


Vous pouvez également ouvrir des shapefiles dans votre projet mais ils ne seront pas éditables dans l'application. Si vous souhaitez utiliser des shapefiles, placez les dans le dossier de l'application (par exemple le dossier **geopoppy** de votre bureau avant de les ouvrir dans QGIS et vérifiez dans le menu **Projet > Propriétés du projet > Général** que les chemins sont enregistrés en **relatif** et non en absolu)






IV.2 - Les outils d'édition

Les **outils d'édition** de champ dans QGIS sont extrêmement utiles pour construire un formulaire d'édition qui soit le plus simple et intuitif pour l'utilisateur de l'application. Ils permettent notamment de créer des menus déroulants ou des réponses à choix multiple. Ainsi, il est important de passer en revue l'intégralité des champs des couches éditables pour choisir l'outil d'édition qui correspond le mieux.

Les outils d'édition se définissent dans les **Propriétés de la couche**, onglet **Champs**. Par défaut tous les champs sont en **Edition de texte** ce qui signifie que l'utilisateur peut rentrer n'importe quelle valeur dans une zone de texte.

▼ Champs



Id	Nom	Outil d'édition
abc 2	localite	Valeur relationnelle
abc 3	beneficiaire	Valeur relationnelle
 4	date_visite	Date/Heure
abc 5	type	Valeur relationnelle
123 6	g_gros_bois	Édition de texte
123 7	g_total	Édition de texte
123 8	h_moyen	Édition de texte
123 9	age	Édition de texte
123 10	densite	Édition de texte
 11	d_eclaircie	Date/Heure
 12	d_elagage	Date/Heure
 13	d_recolte	Date/Heure
abc 14	uuid	Générateur d'UUID
abc 15	identifiant	Cachée
abc 16	editepar	Édition de texte
 17	editedate	Édition de texte

/!\ Bien qu'il soit écrit **Édition de texte**, si le champ est de type **nombre** (decimal interger etc.) ou **date**, l'utilisateur devra tout de même rentrer le bon format de valeur.

Pour changer le type d'outil d'édition, cliquez sur le bouton dans la colonne **Outil d'édition** du champ concerné et choisissez celui qui convient le mieux en fonction des informations suivantes :

- Dans la version de Lizmap qui est celle installée par défaut sur GeoPoppy, les outils d'édition **Couleur**, **Ressource Externe**, **Référence de la relation**, **Valeurs Uniques** et **Vue Web** ne sont pas supportés (documenté ici : https://docs.3liz.com/fr/publish/advanced_lizmap_config.html#editing-data-in-lizmap)
- Choisir **Boîte à cocher** pour les valeurs booléennes. Vous pouvez définir dans *Représentation d'un état coché / décoché* la valeur qui sera donnée au champ en fonction (*par exemple oui / non , 1 / 0 etc.*).
- Choisir **Classification** pour créer un menu déroulant constitué des attributs utilisés pour la classification de la couche. Cette option n'est disponible que si vous avez choisi une symbologie catégorisée pour la couche.

- Choisir **Date/Heure** pour faciliter la saisie des dates via des menu déroulants Jour / Mois / Année. Nous avons constaté l'absence du menu déroulant pour les années ce qui demande de l'écrire en chiffre à chaque fois.
- **Nom de fichier** et **Photo** fonctionnent de la même manière et permettent d'inscrire le chemin vers un fichier ou une photo. Attention, il ne s'agit que du chemin, pas du fichier ou de la photo elle-même.
- Choisir **Edition de texte** pour les champs libres, qu'ils s'agisse de texte ou de chiffres.
- Choisir **Générateur d'UUID** si vous avez créé un champ **uuid**. Les UUIDs sont des identifiants générés aléatoirement qui n'ont théoriquement aucune chance d'apparaître en double, à l'inverse des **id** qui s'incrémentent et qui sont donc dupliqués si les bases de données sont utilisées par plusieurs utilisateurs en mode déconnecté. Retrouvez plus d'information sur les **uuid** et leur utilisation dans la partie **Synchronisation**.
- Choisir **Liste de valeurs** pour créer un menu déroulant et spécifiez les différentes valeurs possibles ainsi que leur description.
- Choisir **Valeur relationnelle** pour créer un menu déroulant comprenant des valeurs provenant d'une autre couche ou table. Choisissez dans **couche** celle qui comprend les valeurs puis dans **colonne clé** et **colonne de valeurs** les champs appropriés. Par exemple, nous avons utilisé cet outil d'édition sur le champ **propriétaire** des parcelles agricoles en le liant à la **table des propriétaires** sur le champ **nom**. Notez qu'il s'agit également du seul widget qui permette les selections multiples, **Liste de valeurs** ne le permettant pas. Si vous souhaitez donner la possibilité pour un utilisateur de sélectionner plusieurs valeurs au sein d'un champ, vous serez obligé d'utiliser cet outil d'édition, que la couche de référence soit éditable ou non.

IV.4 - Le plugin LizMap

Le plugin LizMap sert à construire un fichier de configuration obligatoire à la conversion de votre projet QGIS en une webmap dans l'interface Web de Lizmap. Il se télécharge classiquement dans le **gestionnaire d'extensions** de QGIS.

L'utilisation de ce plugin est documentée sur le site de Lizmap ici :

https://docs.3liz.com/fr/publish/lizmap_configuration.html. Configurez votre projet en fonction de vos besoins en suivant ces conseils :

- Faites bon usage des **Utilisateurs** et **Groupes** de Lizmap pour restreindre l'utilisation des projets et des couches aux personnes concernées (documentation ici : <https://docs.3liz.com/fr/admin/users-groups.html>). C'est particulièrement important si vous avez plusieurs utilisateurs travaillant sur le même projet mais avec des droits d'édition différents sur les couches. En configurant proprement les droits, vous pourrez facilement dupliquer le projet sur tous les Raspberry Pi sans avoir à ajouter / enlever des couches et se retrouver avec un projet différent par utilisateur.
- Suivez impérativement les conseils donnés ici : https://docs.3liz.com/fr/publish/advanced_lizmap_config.html#optimizing-lizmap sur **l'optimisation du rendu WMS** pour gagner en performance lors de l'affichage des couches.
- Pensez à donner l'accès à la **table attributaire** pour les tables éditables sans géométrie qui est le

seul moyen pour l'utilisateur d'accéder à un enregistrement pour le consulter ou l'éditer (les tables n'apparaissant pas sur la carte, impossible de cliquer dessus !).

- Pensez à activer **Positionnement automatique** dans **Option de carte** pour pouvoir utiliser le GPS de la tablette.

Nous avons rencontré des soucis à faire fonctionner le GPS si le projet est défini avec des projections UTM (dans notre cas le EPSG:32630 pour la Cote d'Ivoire), nous recommandons de laisser le projet en EPSG:3857 pour être sûr du bon fonctionnement du GPS. De même, le GPS ne fonctionnera pas si vous accédez la carte en **http**, donc connectez vous toujours en **https** sur la tablette et acceptez l'exception de sécurité.

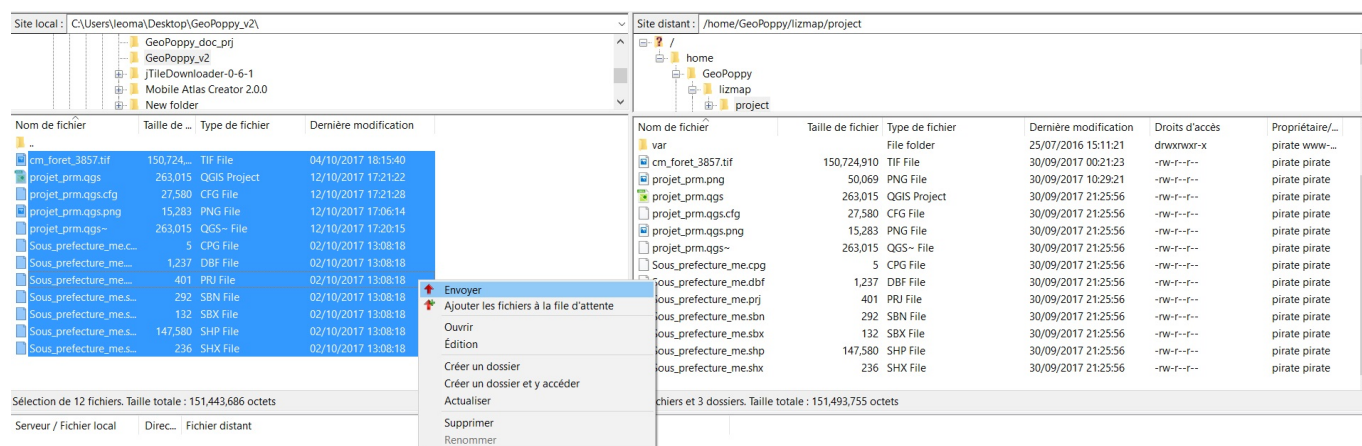
V - Publication du projet

Une fois votre projet fini et le fichier de configuration Lizmap créé grâce au plugin, il ne reste plus qu'à l'envoyer via FTP sur le RaspberryPi pour le tester. La documentation suivante s'appuie sur le logiciel FileZilla mais il est possible d'utiliser n'importe quel autre client FTP.

1. Téléchargez, installez et ouvrez [FileZilla](#).
2. Créez une nouvelle connexion avec les paramètres suivants :

- Hôte : **172.24.1.1**
- Identifiant : **pirate**
- Mot de passe : **hypriot**
- Port : **22**

1. Dans la navigation **Site local**, naviguez jusqu'au dossier qui contient votre projet et les ressources associées (miniature, shapefiles, rasters ...).
2. Dans la navigation **Site distant**, naviguez jusqu'au dossier **/home/GeoPoppy/lizmap/project**.
3. Sélectionnez l'ensemble des fichiers dans le répertoire Site Local, faites un clic droit dessous puis cliquez sur **Envoyer**.



1. Accédez à la page <http://172.24.1.1> depuis un navigateur depuis votre ordinateur ou votre tablette pour vérifier que le projet a bien été envoyé.

VI - Utilisation du projet LizMap

L'utilisation de LizMap coté client est documentée ici : <https://docs.3liz.com/fr/user/index.html>.

Quelques conseils supplémentaires :

- Faites attention aux **relations qui existent entre les couches lors de l'édition**. Si votre couche de parcelle demande de renseigner le propriétaire, ajoutez d'abord le propriétaire dans la liste.
- Utilisez la fonction **Rester centré** de la géolocalisation pour forcer la carte à se positionner toujours sur votre localisation lors de la collecte de points GPS.
- Apprenez à bien utiliser la fonction de **zoom sur une zone dessinée par l'utilisateur**, il s'agit de la meilleure manière de zoomer rapidement sur un lieu donné.

VII - Customisation de l'application en Javascript

LizMap permet l'ajout de scripts personnalisés pour customiser l'application à son gré. Les différents événements Javascript sont documentés ici :

https://docs.3liz.com/fr/publish/advanced_lizmap_config.html#adding-your-own-javascript ainsi que la méthode pour activer les scripts. Les exemples suivants ont été conçus lors de la mission :

creedate.js

Ce script permet, pour toutes les couches éditables disposant d'un champ **creedate** de type **date** de rentrer automatiquement la date de création de l'enregistrement.

```
function formatDate(date) { // function to format date to yyyy-mm-dd
    var d = new Date(date),
        month = '' + (d.getMonth() + 1),
        day = '' + d.getDate(),
        year = d.getFullYear();

    if (month.length < 2) month = '0' + month;
    if (day.length < 2) day = '0' + day;

    return [year, month, day].join('-');
}

lizMap.events.on({
    'lizmapeditionformdisplayed': function(e) {
        var fi = $('#jforms_view_edition_credate');
        if (fi.length) { // if there's a field creedate
            fi.removeClass("hasDatepicker").removeAttr('id')
            //destroying jquery datepicker
            fi.datepicker('destroy')
            $('#jforms_view_edition_credate_label').hide()
            $('.ui-datepicker-reset').hide()
            $('.ui-datepicker-trigger').hide()
            fi.hide() // we hide the label and the input
            if (fi.val().length) {} // if the feature already has a creedate, nothing
            happens
            else { //otherwise we put today's date
```

```

        fi.val(formatDate(new Date())) //set today date
    }
}
});

```

editedate.js

Ce script permet, pour toutes les couches éditables disposant d'un champ **editedate** de type **date** de rentrer automatiquement la date de la dernière édition de l'enregistrement.

```

function formatDate(date) { // function to format date to yyyy-mm-dd
    var d = new Date(date),
        month = '' + (d.getMonth() + 1),
        day = '' + d.getDate(),
        year = d.getFullYear();
    if (month.length < 2) month = '0' + month;
    if (day.length < 2) day = '0' + day;
    return [year, month, day].join('-');
}

lizMap.events.on({
    'lizmapeditionformdisplayed': function(e){
        var fi = $('#jforms_view_edition_editedate');
        if (fi.length){ // if there's a field editedate
            //destroying datepicker
            fi.removeClass("hasDatepicker").removeAttr('id')
            fi.datepicker('destroy')
            $('#jforms_view_edition_editedate_label').hide()
            $('.ui-datepicker-reset').hide()
            $('.ui-datepicker-trigger').hide()
            fi.hide() // we hide the label and the input
            fi.val(formatDate(new Date())) //set today date
        }
    }
});

```

creepar.js

Ce script permet, pour toutes les couches éditables disposant d'un champ **creepar** de type **text** de rentrer automatiquement le nom de l'utilisateur qui a créé l'enregistrement.

```

lizMap.events.on({
    'lizmapeditionformdisplayed': function(e){
        var fi = $('#jforms_view_edition_creepar');
        if (fi.length){ // if there's a field creepar
            $('#jforms_view_edition_creepar_label').hide()
            fi.hide() // we hide the label and the input
            if( $('#info-user-login').length ){ // If user is logged in
                if (fi.val().length){ // if the feature already has a creepar, nothing
happens
                }
            }
        }
    }
});

```

```

        else{ //otherwise we put the login name
            fi.val($('#info-user-login').text())
        }
    }
}

});

```

editepar.js

Ce script permet, pour toutes les couches éditables disposant d'un champ **editepar** de type **text** de rentrer automatiquement le nom de l'utilisateur qui a édité pour la dernière fois l'enregistrement.

```

lizMap.events.on({
    'lizmapeditionformdisplayed': function(e){
        var fi = $('#jforms_view_edition_editepar');
        if (fi.length){ // if there's a field createdby
            $('#jforms_view_edition_editepar_label').hide()
            fi.hide() // we hide the label and the input
            if( $('#info-user-login').length ){ // If user is logged in
                fi.val($('#info-user-login').text())
            }
        }
    }
});

```

surface.js

Ce script permet, pour toutes les couches éditables de type polygone disposant d'un champ **surface** de type **float** d'inscrire automatiquement son aire en hectare. Attention, le script ne fonctionne qu'à l'ouverture d'une entité existante donc si vous venez de tracer une parcelle, vous devez re-ouvrir son formulaire pour mettre à jour son aire la première fois. *Ce script fonctionne mais est très certainement améliorable ...*

```

lizMap.events.on({
    'lizmapeditionformdisplayed': function(e){
        var fi = $('#jforms_view_edition_surface');
        if (fi.length){

$.get('http://172.24.1.1/websig/lizmap/www/index.php/lizmap/edition/editFeature?project=projet_prm&repository=geopoppy&layerId='+e.layerId+'&featureId='+e.featureId+',function(data) {
            data = JSON.stringify(data)
            data = data.substring(data.indexOf("POLYGON") + 9);
            data = data.substring(0, data.indexOf('))'));
            data = data.replace(/,/g,'],[')
            data = data.replace(/ /g,','')
            data = '['+data+']'
            var points1 = JSON.parse(data);
            var points2 =[]

```

```

        for (var i in points1){
            var p = new
OpenLayers.LonLat(points1[i][0],points1[i][1]).transform('EPSG:3857', 'EPSG:4326')
            points2.push(new OpenLayers.Geometry.Point(p.lon, p.lat))
        }
        var rings2 = new OpenLayers.Geometry.LinearRing(points2);
        var polygon2 = new OpenLayers.Geometry.Polygon([rings2]);
        var surface = polygon2.getGeodesicArea()
        surface = surface/10000
        surface = Math.round(surface * 100) / 100
        fi.val(surface)
    })
}
}
});

```

VIII - Troubleshooting

Lors de l'utilisation de l'application, nous avons rencontré deux principaux problèmes. Cette section (qui sera amenée à évoluer) présente les solutions temporaires que nous avons mis en place.

VIII.1 Rafraichissement des couches

Après plusieurs dizaines de minutes d'utilisation continue de l'application, il arrive que les couches WMS ne se rafraichissent plus lorsque l'utilisateur navigue sur la carte. La solution trouvée pour l'instant est simplement de rafraichir la page du navigateur pour relancer l'application.

VIII.2 "Service non disponible"

De manière régulière, le chargement de la carte n'aboutissait pas et un message d'erreur "SERVICE NON DISPONIBLE" apparaissait en fond de l'application. Ce problème a pour l'instant été résolu en replaçant le fichier **docker-compose.yml** situé dans le dossier /home/pirate du RaspberryPi par le fichier suivant :

https://github.com/CartONG/geopoppy_documentation/blob/master/docker-compose/docker-compose.yml ce qui aura pour effet de fixer les adresses IP des différents containers et de rentrer http://172.18.0.7/cgi-bin/qgis_mapserv.fcgi comme URL du server WMS de LizMap dans l'interface admin de LizMap.

Attention, tant que vous ne rencontrez pas ce problème, ne touchez pas au fichier docker-compose.yml. De plus, cette solution est pour l'instant temporaire, suivez l'évolution de la résolution de ce bug ici :

<https://github.com/jancelin/geo-poppy/issues/18>