Cahier de conception

Groupes D :
Ferhat DADACHE
Tassadit LEROUL
Lydia SASMI

Université Paris Sud
Projet GLA S6 Informatique

Table des matières

1- Introduction :

Après avoir établi le cahier des charges et le cahier d'analyse, nous voici à l'étape de conception.

Dans ce document, on montrera les principales classes de l'application sous forme de diagramme de classes, ainsi que les diagrammes de séquences sous forme de diagrammes UML.

On représentera graphiquement les différentes interactions entre les acteurs et le système selon un ordre chronologique dans la formulation. On le fera sous forme de diagrammes de séquence.

On implémentera également quelques méthodes sous forme de code ou de pseudo-code.

2- Composantes techniques :

Langage de programmation :

Java est complètement orienté objet. Java nous permet de développer notre application d'une façon orientée objet et vous nous d'avoir une application bien structurée, modulable, maintenable très facilement et efficace. Cela augmente notre productivité.

Toute notre application sera donc codée en java en utilisant différentes librairies et modules facilitant la programmation.

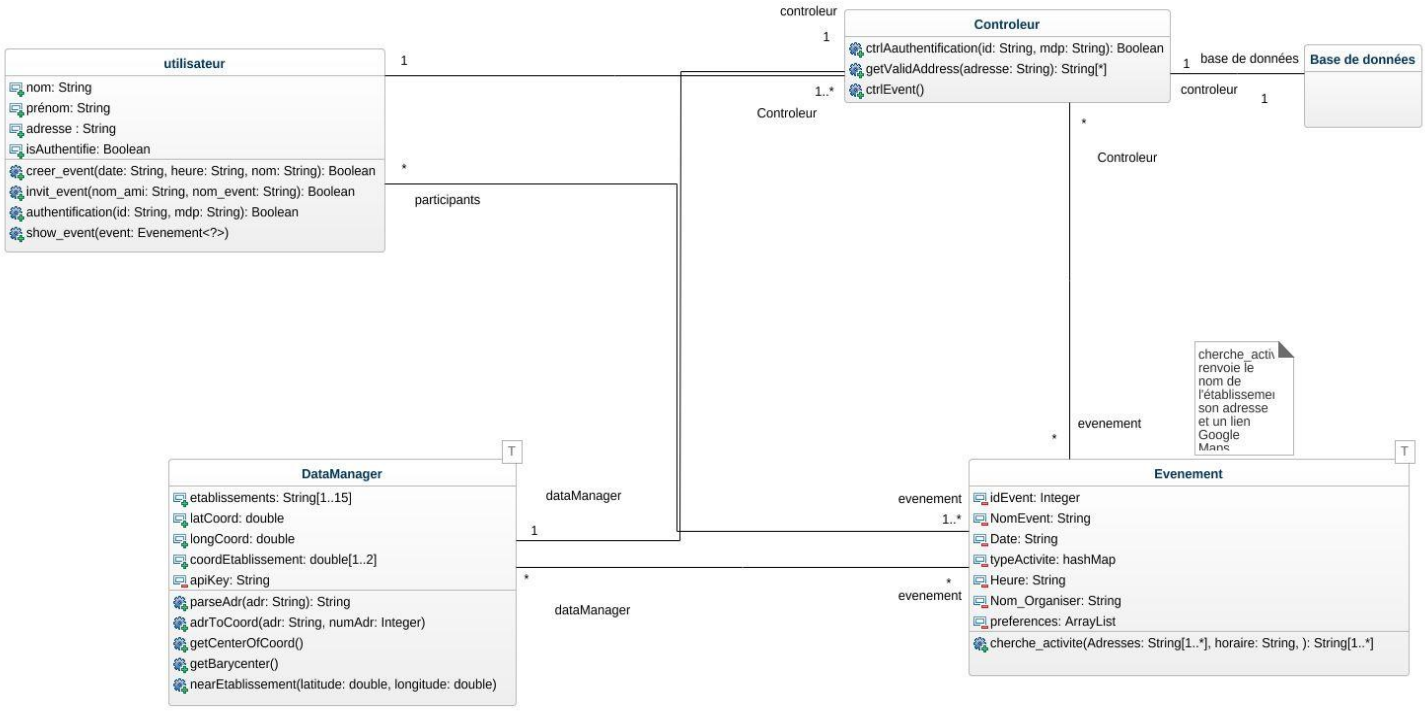
Api Google places

Google Places API for Android permettra à notre application la géolocalisation qui renvoient, selon le contexte, les professionnels locaux et autres lieux à proximité de l'appareil. Autrement dit, nous pouvons concevoir grâce à cet api une application complète basée sur les lieux pertinents pour l'utilisateur et qui complètent les services géographiques simples fournis par les services de géolocalisation Android.

Base de données SQLite

SQLite est une bibliothèque qui implémente un moteur de base de données SQL transactionnel autonome sans configuration. SQLite est la base de données la plus largement déployée au monde avec plus d'applications que nous pouvons compter, y compris plusieurs projets de haut niveau.

3- Diagramme de classes :



Classe utilisateur :

Cette classe défini toutes les options auxquelles un utilisateur de l'application a accès.

On ainsi plusieurs méthodes qui sont créer un évènement et inviter à un événement, ces méthodes ne seront possible que pour les organisateurs.

D'autre méthodes sont possibles tel que show évent et l'authentification.

Classe Contrôleur :

Cette classe permet de contrôler la validité des informations saisie par l'utilisateur (exemple vérifie la bonne saisie des adresses et mails), et permet aussi de vérifier les contraintes de la base de données.

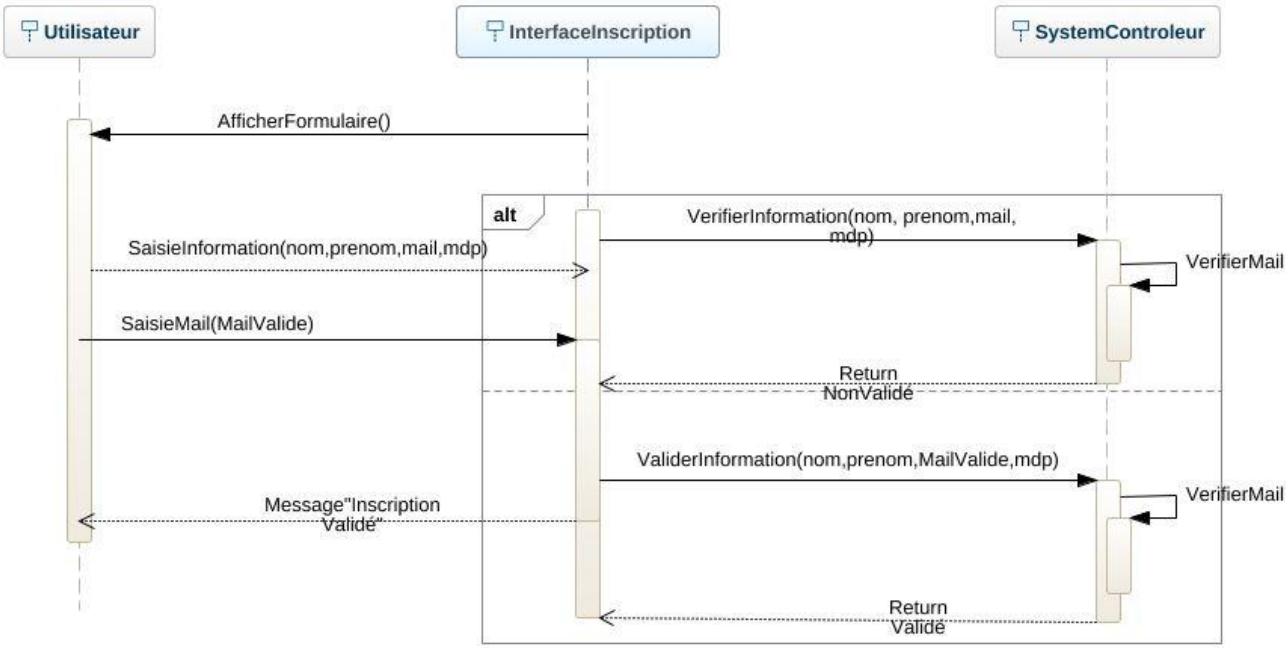
Classe DataManager :

Cette classe permet de gérer les communications entre l'utilisateur et les api google utilisé dans notre application.
Elle permet aussi de calculer la meilleure adresse ou peuvent se rejoindre les amis .

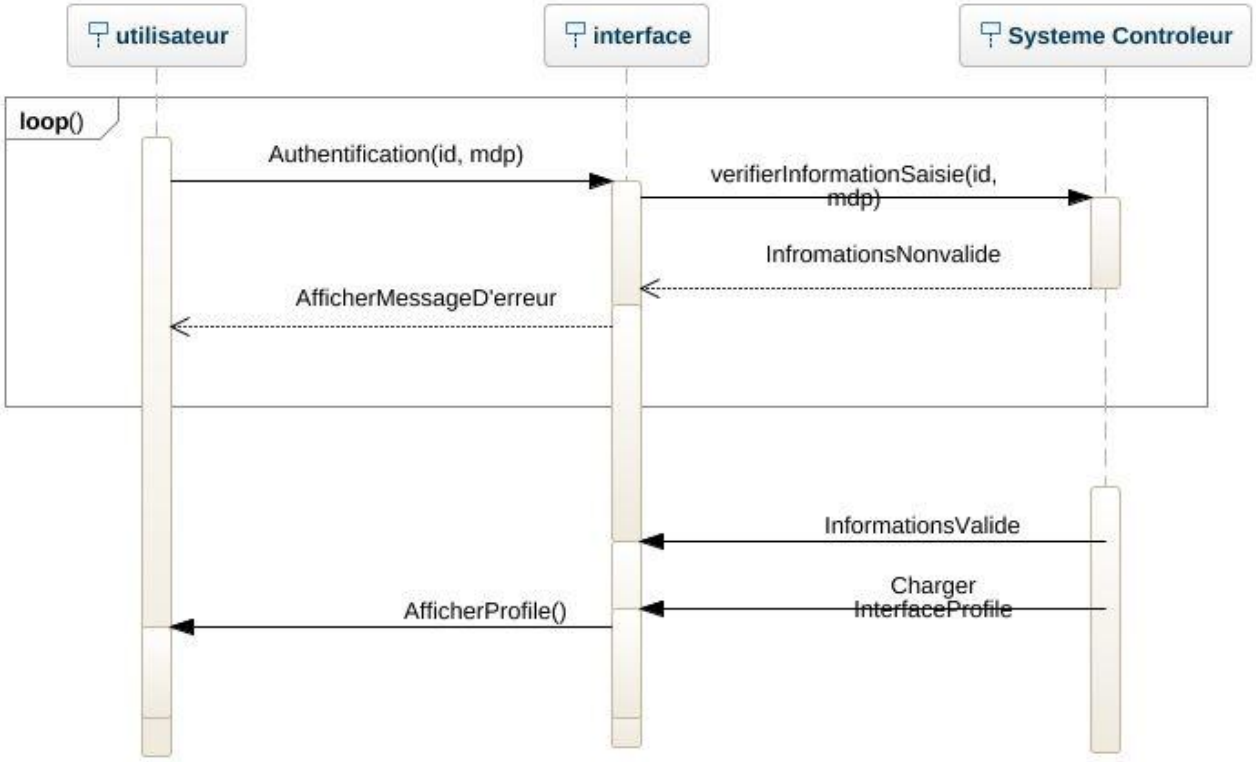
Classe Evènement :

Cette classe permet à l'organisateur de créer un évènement et de le gérer en invitant des amis , elle permet aussi de chercher ls meilleures activités qui correspondent aux envies des participants à l'événement .

4- Inscription :



5- Authentification :

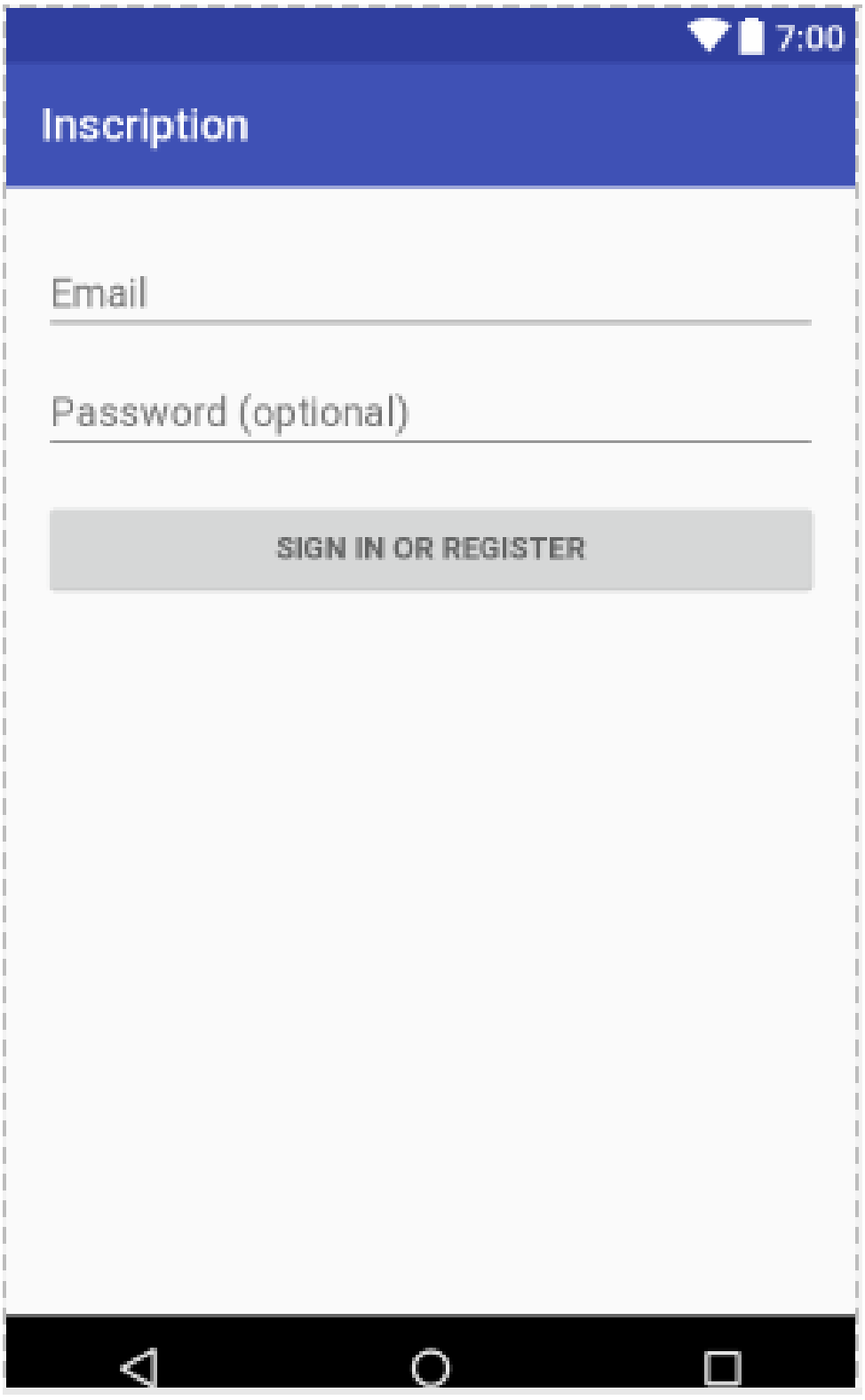


Pseudo code (authentification et inscription) :

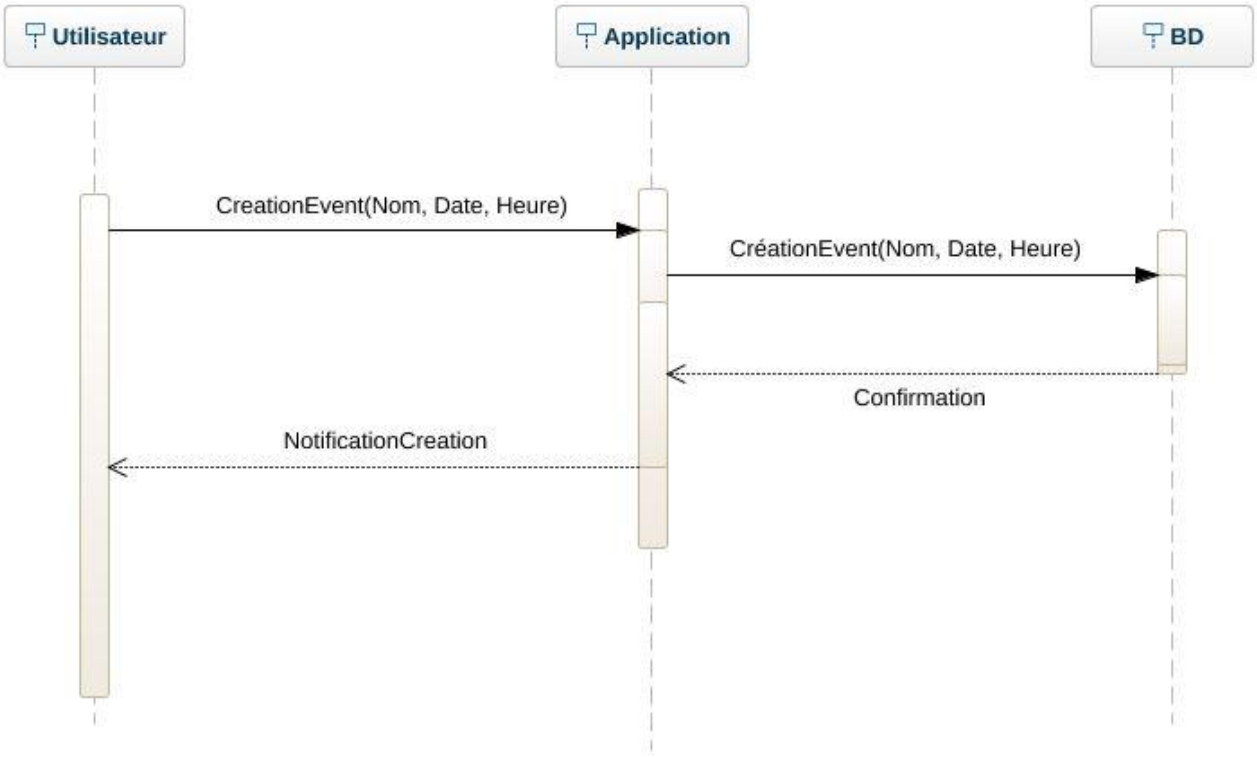
```
public class UserLoginTask extends AsyncTask<Void, Void, Boolean> {

    private final String mEmail;
    private final String mPassword;

    UserLoginTask(String email, String password) {
        mEmail = email;
        mPassword = password;
    }

    @Override
    protected Boolean doInBackground(Void... params) {
        try {
            // Simulate network access.
            Thread.sleep(2000);
        } catch (InterruptedException e) {
            return false;
        }

        for (String credential : DUMMY_CREDENTIALS) {
            String[] pieces = credential.split(":");
            if (pieces[0].equals(mEmail)) {
                // Account exists, return true if the password matches.
                return pieces[1].equals(mPassword);
            }
        }
        return true;
    }
      // TODO: register the new account here.
      return true;
  }

  @Override
  protected void onPostExecute(final Boolean success) {
      mAuthTask = null;
      showProgress(false);

      if (success) {
          finish();
      } else {
          mPasswordView.setError("This password is incorrect");
          mPasswordView.requestFocus();
      }
  }

  @Override
  protected void onCancelled() {
      mAuthTask = null;
      showProgress(false);
  }
```

```
private void attemptLogin() {
        if (mAuthTask != null) {
            return;
        }

        mEmailView.setError(null);
        mPasswordView.setError(null);

        String email = mEmailView.getText().toString();
        String password = mPasswordView.getText().toString();

        boolean cancel = false;
        View focusView = null;

        if (!TextUtils.isEmpty(password) && !isPasswordValid(password)) {
            mPasswordView.setError(getString(R.string.error_invalid_password));
            focusView = mPasswordView;
            cancel = true;
        }
        if (TextUtils.isEmpty(email)) {
            mEmailView.setError(getString(R.string.error_field_required));
            focusView = mEmailView;
            cancel = true;
        } else if (!isEmailValid(email)) {
            mEmailView.setError(getString(R.string.error_invalid_email));
            focusView = mEmailView;
            cancel = true;
        }
        if (cancel) {
            focusView.requestFocus();
        } else {
            showProgress(true);
            mAuthTask = new UserLoginTask(email, password);
            mAuthTask.execute((Void) null);
        }
    }
```

```
private boolean isEmailValid(String email) {
    return email.contains("@");
}

private boolean isPasswordValid(String password) {
    return password.length() > 6;
}
```


7:00
Inscription
Email
Password (optional)
SIGN IN OR REGISTER

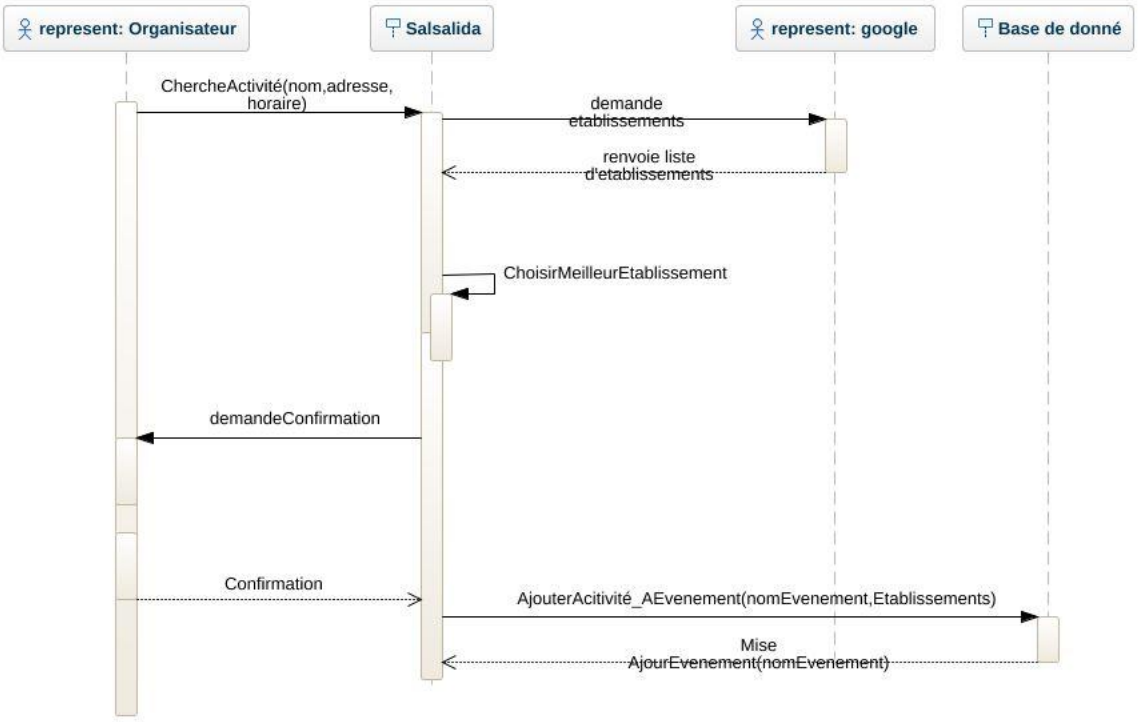
6- Création évènement :



Pseudo code (Création évènement) :

```
for(int i=0;i<listeE.size();i++){
    Map<String,Object> info = new HashMap<>();

    info.put("eventname",listeE.get(i).name);
    info.put("nom",listeE.get(i).hisOrganiser.getName());
    info.put("prenom",listeE.get(i).hisOrganiser.getFirstname());

    Date datedeb = dateFormatInput.parse(listeE.get(i).dateDeb);
    info.put("hdeb",dateFormatOutput.format(datedeb));

    Date datefin = dateFormatInput.parse(listeE.get(i).dateFin);
    info.put("hfin",dateFormatOutput.format(datefin));

    info.put("adresse",Address.getAddressFromId(listeE.get(i).hisOrgan:
    info.put("own",true);
    info.put("ide",listeE.get(i).ide);

    events.add(info);
}
```

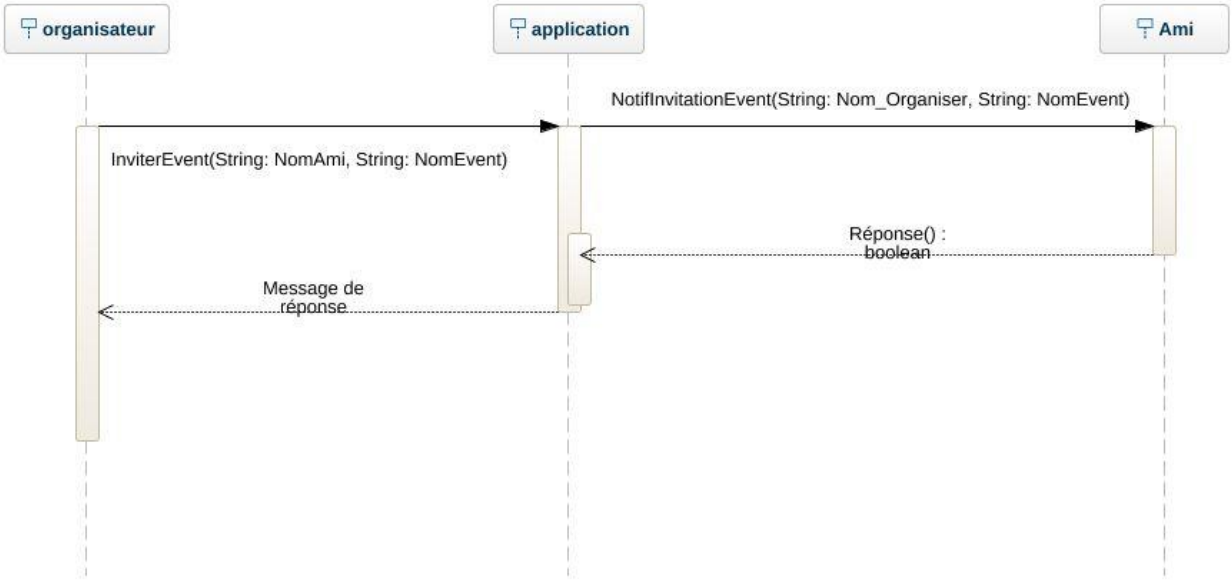
7- Ajout d’activité :



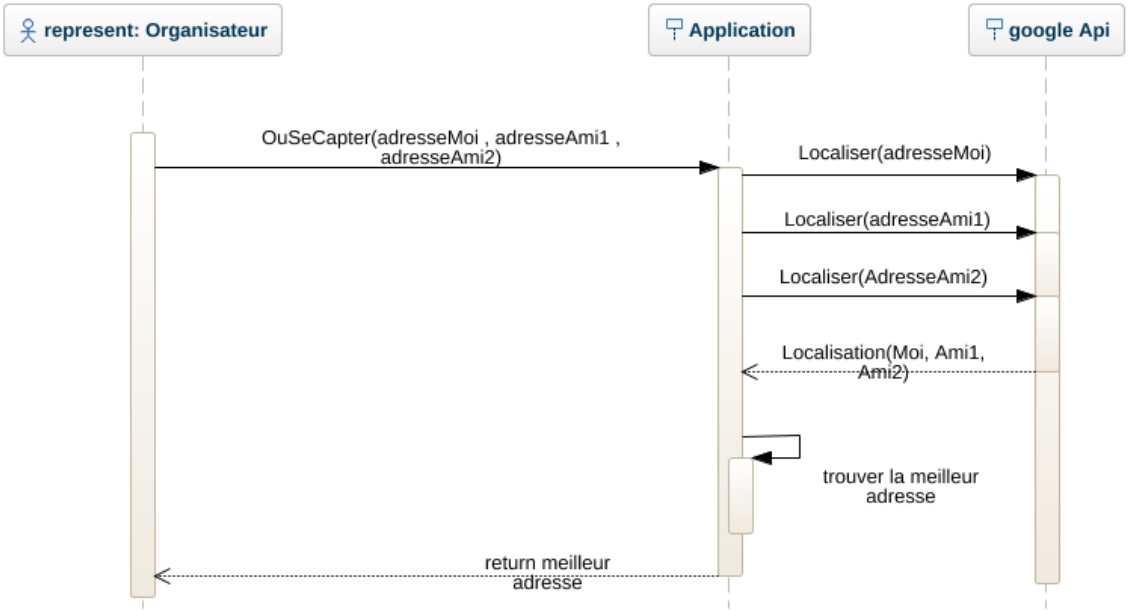
Pseudo code (recherche activité) :

```
public static void NearbySearchEtablissement(double Lat, double Lng) throws Exception {
    String s = "https://maps.googleapis.com/maps/api/place/nearbysearch/json?location=" + Lat + "," + Lng
            + "+&radius=" + u.perimetre + "&type=bar&key=" + GooglePlacesKey;
    URL url = new URL(s);
    // lire l'URL
    Scanner scan = new Scanner(url.openStream());
    String str = new String();
    while (scan.hasNext())
        str += scan.nextLine();
    scan.close();

    // JSON contenant les résultats.
    JSONObject obj = new JSONObject(str);
    if (obj.getString("status").equals("ZERO_RESULTS")) {
        Etablissement.name = "Aucun etablissement n'est trouvé";
        coordEtablissement[0] = centerLat;
        coordEtablissement[1] = centerLng;
        return;
    } else if (!obj.getString("status").equals("OK")) {
        Etablissement.name= "Erreur";
        coordEtablissement[0] = centerLat;
        coordEtablissement[1] = centerLng;
        return;
    }
    // renvoie le nom de l'etablissement, son adresse courte et
    // le lien google maps
    JSONObject lieu = (obj.getJSONArray("results")).getJSONObject(compteurRecherche % obj.length());
    Etablissement.name = lieu.getString("name");
    System.out.println(Etablissement.name);
    Etablissement.vicinity = lieu.getString("vicinity");
    String etablissementVicinityParse = parseAddr(Etablissement.vicinity);
    System.out.println(Etablissement.vicinity);
    coordEtablissement[0] = lieu.getJSONObject("geometry").getJSONObject("location").getDouble("lat");
    coordEtablissement[1] = lieu.getJSONObject("geometry").getJSONObject("location").getDouble("lng");
    Etablissement.Adresse = "https://www.google.com/maps/?q=" + etablissementVicinityParse;
    compteurRecherche++;
}
```

8- Invitation d'ami à un événement :



9- Trouver la meilleure adresse pour se retrouver



Pseudo code :

Pour trouver le Barycentre des adresses il nous faut transformer les adresses écrite en chaines de caractère en coordonnées de localisation .Notre classe s'occupera de faire cela et ensuite on recherchera le barycentre des points de localisations

```java
// trouver le barycentre de plusieurs adresses
    public static void getBarycentre() {
        int numAddr = 1;
        for (String addr : u.adresses) {
            try {
                GestionDonnees.addrToCoord(addr, numAddr);
            } catch (Exception e1) {
                e1.printStackTrace();
            }
            numAddr++;
        }
        for (int i = 0; i < GestionDonnees.coordLat.length; i++) {
            System.out.println("Lat : " + coordLat[i]);
            System.out.println("Lng : " + coordLng[i]);
        }
        getCenterCoord();
    }
```

10- Conclusion :

Nous veillerons à respecter les contraintes d'entrées de l'utilisateur et du cahier des charges.

Ainsi, grâce à ces différents outils, nous réaliserons l'ensemble des demandes de l'utilisateurs.