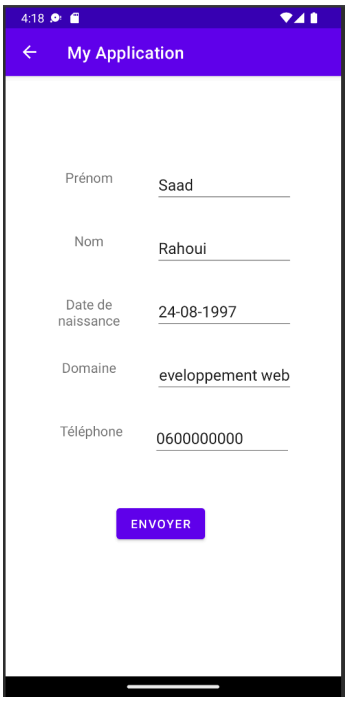
TP1 : Les Bases d'Android

Exercice 3 : Une première application- Interface simple

```
<TextView
    android:id=« @+id/textView"
    android:layout_width="100dp"
    android:layout_height="42dp"
    android:layout_marginStart=« 28dp"
    android:layout_marginTop="108dp"
    android:layout_marginBottom="32dp"
    android:text="@string/firstName"
    android:textAlignment="center"
    android:textSize=« 16sp"
    app:layout_constraintBottom_toTopOf="@+id/
textView4"
    app:layout_constraintEnd_toStartOf="@+id/
firstName"
    app:layout_constraintHorizontal_bias="0.176"
    app:layout_constraintStart_toStartOf="parent"
   app:layout_constraintTop_toTopOf="parent" />
```

```
<EditText
    android:id="@+id/firstName"
    style="@style/Widget.AppCompat.EditText"
    android:layout_width="162dp"
    android:layout_height="42dp"
    android:layout_marginStart="8dp"
    android:layout_marginTop="108dp"
    android:layout_marginEnd="36dp"
    android:ems="10"
    android:inputType="textPersonName"
    android:textAppearance="@style/
TextAppearance.AppCompat.Medium"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toEndOf="@+id/
textView"
    app:layout_constraintTop_toTopOf="parent" />
```



Nous commençons par créer une nouvelle activité et la déclarer dans le manifeste. Nous créons ensuite dans le fichier xml correspondant à l'activité et nous déclarons les propriétés demandées.

Un TextView représentant le label et un EditText pour chaque entrée, et puis un bouton de validation.

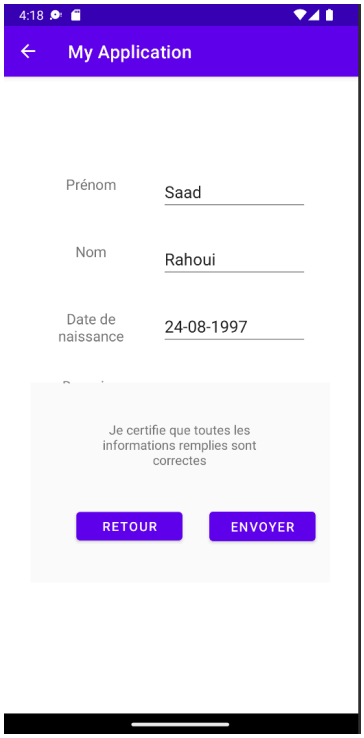
Puis, on ajoute des contraintes de position en fonction des éléments aux alentours dans un *constraint layout. Voici donc le résultat*

Exercice 4 : Internationalisation des interfaces

L'isolation des ressources dans un fichier strings permet de manipuler facilement les textes et la langue d'affichage. Pour ajouter une nouvelle langue. L'environnement Android Studio permet d'y procéder facilement en faisant un click droit sur le dossier strings.xml > *open translator editor.* Cet outil permet de générer un nouveau fichier strings.xml contenant la nouvelle langue, comme l'original avec des ressources clé/valeur.

```
<resources>
    <string name="app_name">My Application</string>
    <string name="button_send">Send</string>
    <string name="main_button">Singup</string>
    <string name="firstName">First Name</string>
    <string name="lastName">Last Name</string>
    <string name="birth_date">Birthday</string>
    <string name="domaine">Domaine</string>
    <string name="phoneNumber">Phone</string>
    <string name="createUserButton" translatable="false">Send</string>
</resources>
```

Exercice 5 : Événements associés aux objets graphiques d'une vue



Tout d'abord nous créons le bouton dans le fichier xml correspondant puis nous en créons une instance dans le code java

```
Button createButton = (Button) findViewById(R.id.createUser);
```

Ensuite on associe un événement a ce bouton dans la méthode onCreate()

```
createButton.setOnClickListener(new View.OnClickListener(){
    public void onClick(View view) {}
```

La prochaine étape serait de créer le *popUp window* que nous allons afficher lorsqu'on click sur ce bouton.

```
createButton.setOnClickListener(new View.OnClickListener(){
    public void onClick(View view) {
});
```

On crée le popUp window et on y associe un layout.

```
LayoutInflater inflater = (LayoutInflater) getSystemService(LAYOUT_INFLATER_SERVICE);
View popupView = inflater.inflate(R.layout.popup_design, null);

int width = LinearLayout.LayoutParams.WRAP_CONTENT;
int height = LinearLayout.LayoutParams.WRAP_CONTENT;
final PopupWindow popupWindow = new PopupWindow(popupView, width, height, true);
popupWindow.showAtLocation(view, Gravity.CENTER, 0, 0);
```

layout.popup_design :

```
<View
    android:id="@+id/view"
    android:layout_width="330dp"
    android:layout_height="220dp"
    android:layout_marginTop="350dp"
    android:layout_marginBottom="100dp"
    android:background="#FAFAFA"
    android:foregroundGravity="center_vertical|center|center_horizontal"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.0"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:layout_constraintVertical_bias="0.0" />
```

La dernière étape serait de cacher cette vue lorsqu'on click autre part. On peut y parvenir avec la méthode dismiss()

```
  popupView.setOnTouchListener(new View.OnTouchListener() {

      public boolean onTouch(View v, MotionEvent event) {
        popupWindow.dismiss();
         return true;
    }
```

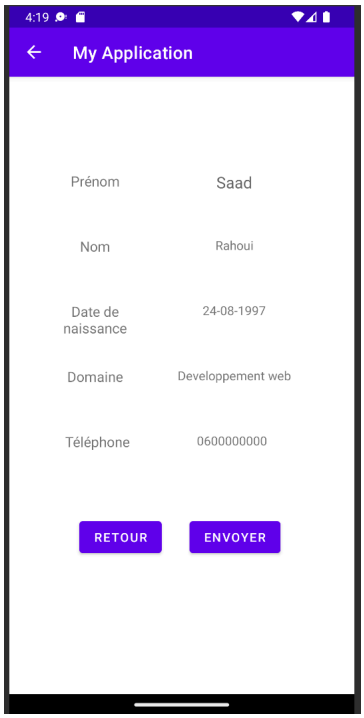
Exercice 6 : Intent explicite

```
public void showUser(View view){
    Intent showUserIntent = new Intent(this, showUserActivity.class);
    showUserIntent.putExtra("firstName", ((EditText) findViewById(R.id.firstName)).getText().toString());
    showUserIntent.putExtra("lastName", ((EditText) findViewById(R.id.lastName)).getText().toString());
    showUserIntent.putExtra("birthday", ((EditText) findViewById(R.id.birthdate)).getText().toString());
    showUserIntent.putExtra("domain", ((EditText) findViewById(R.id.domain)).getText().toString());
    showUserIntent.putExtra("phone", ((EditText) findViewById(R.id.phone)).getText().toString());
    startActivity(showUserIntent);
}
```

Nous allons instancier un intent *showUserIntent* qui vas récupérer les données de l'utilisateur et qui va les envoyer dans une autre activité *showUserActivity* que nous créons également.

Dans la class *showUserActivity*, nous récupérons *l'intent* que nous avons envoyé. Et nous récupérons les informations utilisateur par la méthode getStringExtra() et que nous avons ajouter par la méthode putExtra() auparavant.
Il nous reste qu'afficher ces données dans les *TextViews* correspondants.

```
Intent intent = getIntent();
//show data
TextView firstName = (TextView)findViewById(R.id.show_firstName);
firstName.setText(intent.getStringExtra("firstName"));

TextView lastName = (TextView)findViewById(R.id.show_lastName);
lastName.setText(intent.getStringExtra("lastName"));

TextView birthday = (TextView)findViewById(R.id.show_birthdate);
birthday.setText(intent.getStringExtra("birthday"));

TextView domain = (TextView)findViewById(R.id.show_domain);
domain.setText(intent.getStringExtra("domain"));

TextView phone = (TextView)findViewById(R.id.show_phone);
phone.setText(intent.getStringExtra("phone"));

}
```



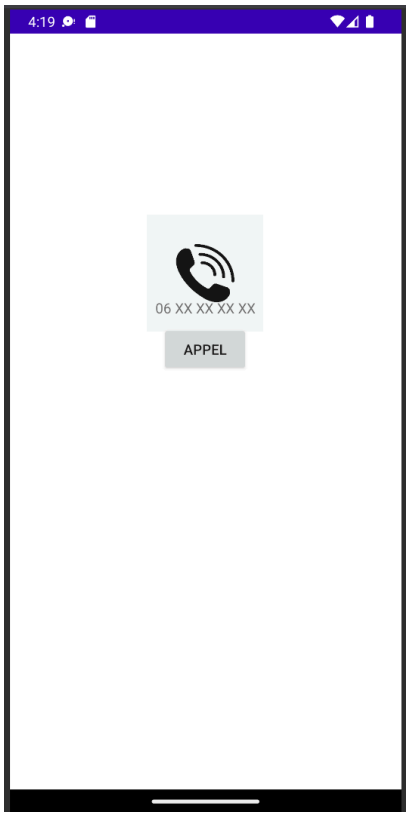
Voici donc le résultat. Les bouton envoyer et retours permettent de nous envoyer vers une autre activité, en passant par *l'intent*.

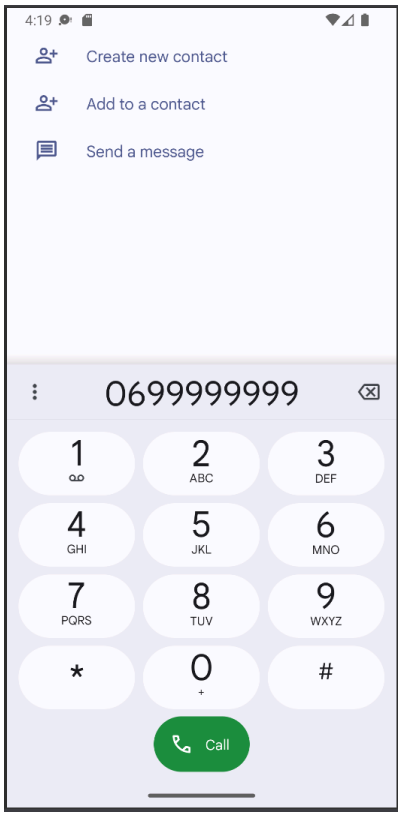
```
Intent intent = new Intent(this, createUserActivity.class);
startActivity(intent);
```

Exercice 7 : Intent implicite

```
public void call(String number){
   Intent callIntent = new Intent(Intent.ACTION_DIAL);
    callIntent.setData(Uri.parse("tel:" + number));

    startActivity(callIntent);
}
```

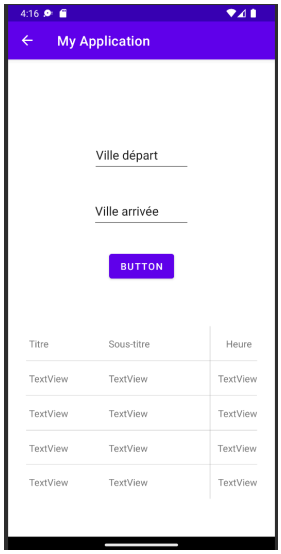
L'intent.ACTION_DIAL est plus facile à implémenter car il nécessite pas l'autorisation utilisateur. Contrairement à intent.ACTION_CALL. Voici donc le resultat





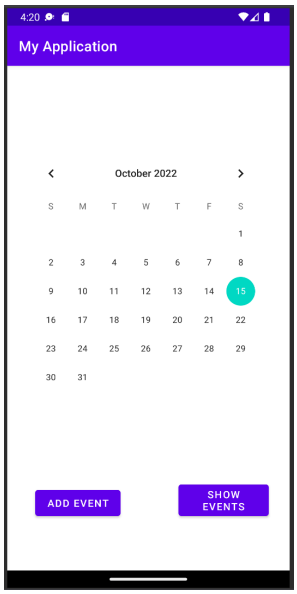

```
public void call(String number){
   Intent callIntent = new Intent(Intent.ACTION_DIAL);
    callIntent.setData(Uri.parse("tel:" + number));

    startActivity(callIntent);
}
```

Exercice 8 : Application simple pour consulter les horaires de train




```
<androidx.recyclerview.widget.RecyclerView
    android:id="@+id/recyclerView"
    android:layout_width="363dp"
    android:layout_height="254dp"
    android:layout_marginTop="32dp"
    android:layout_marginBottom="40dp"
    app:layout_constraintBottom_toBottomOf=« parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/button3" />
```

Exercice 9 : Application simple d’agenda




```
<CalendarView
    android:id="@+id/calendar"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignParentTop="true"
    android:layout_marginLeft="19dp"
    android:layout_marginTop="116dp" />
```

Résultat final : voila donc le point de départ d’application qui permet de naviguer dans toutes les autres activités