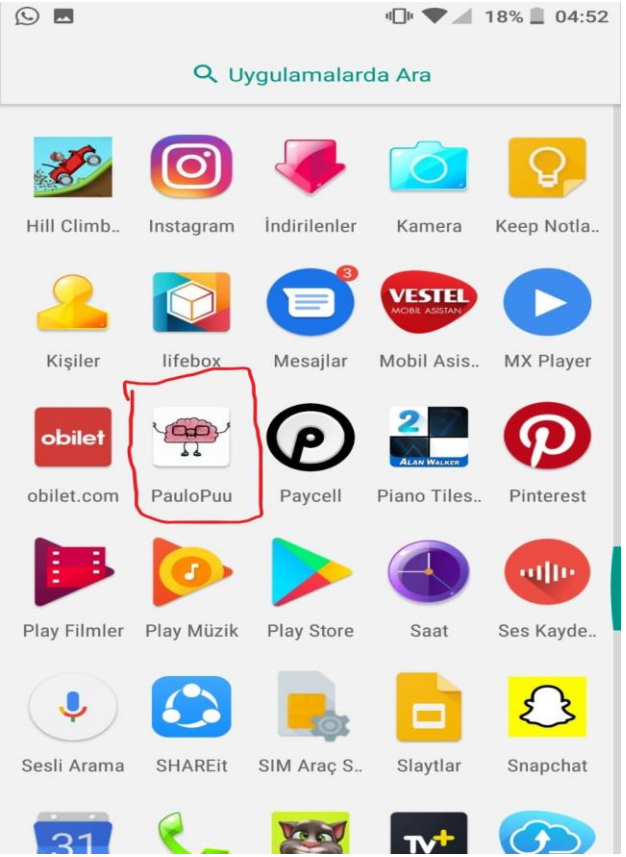
ANDROİD JAVA MOBİL PROGRAMLA

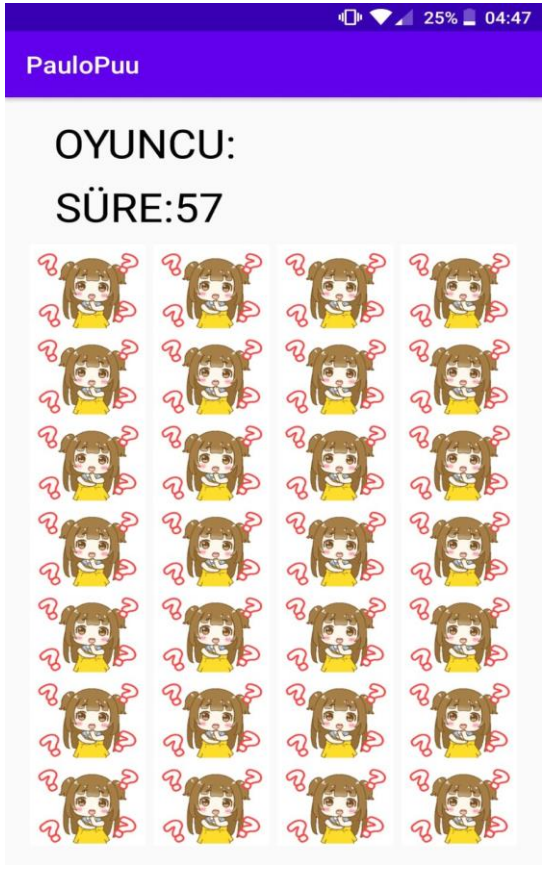
ELİF ÖKSÜZALİ 180303010

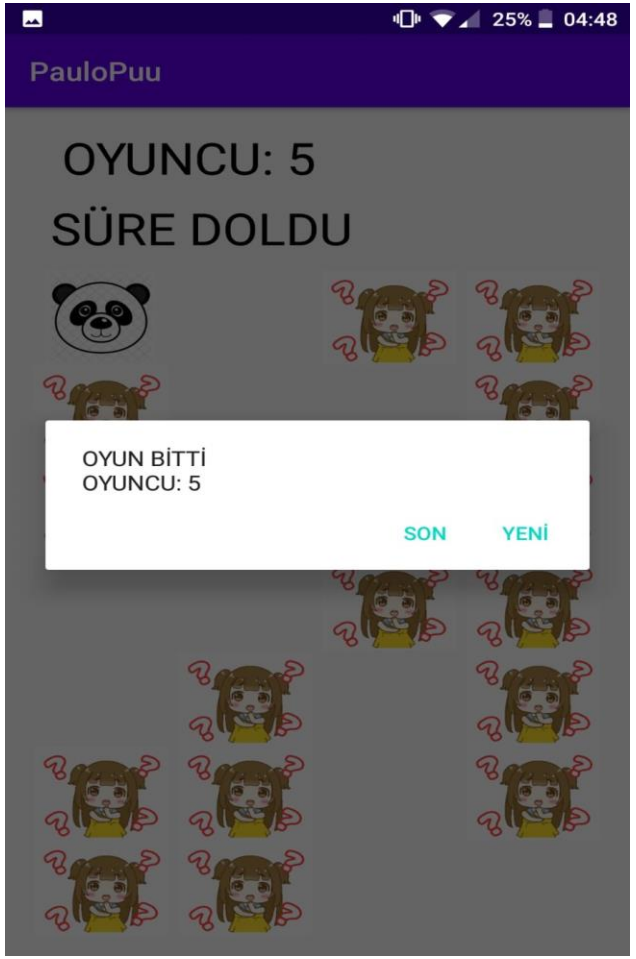
HAFIZA OYUNU

Belirlenen sürede kartları eşleştirip ve oyunu bitirme mantığı ile çalışır.

OYUN GÖRÜNTÜLERİ







```
TextView tv_Süre;
TextView tv_oyuncu1;

ImageView iv_11, iv_12, iv_13, iv_14, iv_21, iv_22, iv_23, iv_24, iv_31, iv_32, iv_33, iv_34, iv_41, iv_42, iv_43, iv_44,
        iv_51, iv_52, iv_53, iv_54, iv_61, iv_62, iv_63, iv_64, iv_71, iv_72, iv_73, iv_74;


// kartlarımız için bir liste yapalım
Integer[] kartList = {101, 102, 103, 104, 105, 106, 107, 108, 109, 110, 111, 112, 113, 114,
        201, 202, 203, 204, 205, 206, 207, 208, 209, 210, 211, 212, 213, 214,};

//mevcut resimler
int image101, image102, image103, image104, image105, image106, image107, image108, image109, image110,
        image111, image112, image113, image114, image201, image202, image203, image204, image205,
        image206, image207, image208, image209, image210, image211, image212, image213, image214;


int firstCart, secondCard;
int clickFirst, clickSecond;
int cardnumber = 1;

int turn = 1;
int Puan = 0;
```

Buradaki kod ekranında drawableye yüklediğim resimleri ve oyun esnasında oyun ekranında kullanmam gereken değişkenleri tanımlayıp birde kart listesi oluşturarak kartlarımızı bir listede tuttum. İmages adında açtığım Xmls kısmında Resimlerin isimlerini int tipinde bir değişken türü ile belirttim.

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main2);

    tv_oyuncu1 = (TextView) findViewById(R.id.tv_oyuncu1);
    tv_Süre=(TextView)findViewById(R.id.tv_Süre);



    iv_11 = (ImageView) findViewById(R.id.iv_11);
    iv_12 = (ImageView) findViewById(R.id.iv_12);
    iv_13 = (ImageView) findViewById(R.id.iv_13);
```

Buradaki ekranda ise findViewById kullanarak tanımlamış Olduğum nesneleri bulur ve programa tanıtır.
Aynı şeklide eklemiş olduğum ve xml kısmında İsimlendirdiğim resimleride tanıtmış olduk

```
// kartlara resimleri yükleyelim şimdi
frontOfCardsResources();

// resimleri karıştıralım
Collections.shuffle(Arrays.asList(kartList));


iv_11.setOnClickListener((v) -> {
        int theCard= (int) v.getTag();
        doStuff(iv_11, theCard);
```

```
private void doStuff(ImageView iv, int card) {
    // Doğru resim seçim kısmı (stuff = karıştırıcı anlamında kullandım )
    if (kartList[card] == 101) {
        iv.setImageResource(image101);
    } else if (kartList[card] == 102) {
        iv.setImageResource(image102);
```

```
    // Oyun bitimini kontrol eden kısım
    FinishGame();
}
private void FinishGame(){
    if(iv_11.getVisibility() == View.INVISIBLE &&
            iv_12.getVisibility() == View.INVISIBLE &&
            iv_13.getVisibility() == View.INVISIBLE &&
```

```
iv_21.setTag(4);
iv_22.setTag(5);
iv_23.setTag(6);
iv_24.setTag(7);
```

```
private void frontOfCardsResources()
{
    image101 = R.drawable.ic_image101;
    image102 = R.drawable.ic_image102;
```

Buradaki kod satırlarını açıklarsak:

setTag: Herhangi bir nesneyi tek bir Görüntü ile ilişkilendirebilmek için kullanılır bende image dosyasında isim vermiş olduğum görüntü ile bir sayıyı ilişkilendirdim bunu yapmasaydım onClik kullanmam gerekirdi. Diyelim ki benzer bir sürü görüş oluşturduk Her görünüm için ayrı ayrı bir onClik yaptık bu yaptığımız şeyler daha sonra benzer şey yapsa bile her görünüm için benzersiz bir onClik oluşturmamız gerekirdi onun yerine bunu kullansak daha iyi .

imageResourse:

Aktarma için tanımladığımız metot ne işe yarar:
İmage xml tanımladığım resimleri id'lerini main deki tanımladığım diğer değişkenlere eşitliyorum

Collections.suffle:

Resimleri karıştırma metodudur karıştırma işlemi yapar

getVisibility: Pencereye eklendiğinde görünümün olmasını istediğiniz görünürlüğü döndürür. Yani arkası dönükken istediğim resim ne ise onu döndürme işlemi yapar.

```
tv_Süre = ( TextView) findViewById(R.id.tv_Süre);
new CountDownTimer( millisInFuture: 60000,  countDownInterval: 1000) {
    @Override
    public void onTick(long millisUntilFinished) {
        tv_Süre.setText(" SÜRE:" + millisUntilFinished/1000);
    }
    @Override
    public void onFinish() { // Oyun bitiminde kişiye yeniden mi değil mi sorusu için yapılan dialog kısmı
        AlertDialog.Builder alertDialogBuilder = new AlertDialog.Builder( context: com.example.paulopuu.Main2Activity.this);
        alertDialogBuilder
                .setMessage("OYUN BİTTİ \nOYUNCU: " +Puan )
                .setCancelable(false)
                .setPositiveButton( text: " YENİ", (dialog, which) → {
                        Intent intent = new Intent(getApplicationContext() , com.example.paulopuu.Main2Activity.class);
                        startActivity(intent);
                })
                .setNegativeButton( text: "SON", (dialog, which) → {
                        finish();
                });
        // AlertDialog alertDialog = alertDialogBuilder.create();
        AlertDialog alertDialog = alertDialogBuilder.create();
        alertDialog.show();
        tv_Süre.setText("SÜRE DOLDU  ");
        Toast.makeText( context: Main2Activity.this, text: "SÜRE DOLDU",Toast.LENGTH_LONG ).show();
    }
}.start();
```

Burada kullanılan **Intent metodu:** uygulama bileşenleri arasında veri akışını sağlayan bir yapıdır burda kullanmamdaki amaç ise kişi yeni oyun isterse tekrardan oyun sayfasına veya oyunu bitirmek isterse başlangıç sayfasına geçebilsin diye kullandım.

AlertDialogBuilder : İletişim Kutusu, kullanıcıdan karar vermesini veya ek bilgi girmesini isteyen küçük bir penceredir.
Oyun süresi boyunca süre bittiğinde yeni mi son mu diye sorması için bende kullandım.

countDownTimer : Zamanlayıcı olarak kullandım belirli bir sürede oyunun bitmesi gerektiği için

```
/ Hangi görüntünün seçildiğini kontrol edip
        if (cardnumber == 1) {
            firstCart = kartList[card];
            if (firstCart > 500) {
                firstCart = firstCart - 400;
            }
            cardnumber = 2;
            clickFirst = card;

            iv.setEnabled(false);
        } else if (cardnumber == 2) {
            secondCard = kartList[card];
            if (secondCard > 200) {
                secondCard = secondCard - 100;
            }
            cardnumber = 1;
            clickSecond = card;

            iv_11.setEnabled(false);
            iv_12.setEnabled(false);
            iv_13.setEnabled(false);
```

1

```
            Handler handeler = new Handler();
            handeler.postDelayed(() -> {
                    // Seçilen görüntünün eşit olup olmadığını anlamak için bir method
                    hesap();
            }, delayMillis: 500);
        }
    }
    private void hesap() {

        // görüntüler eşitse kartı çevir ve puan ekle
        if (firstCart == secondCard) {
            if (clickFirst == 0) {
                iv_11.setVisibility(View.INVISIBLE);
```

2

```
        // Oyuncu doğru yaparsa o oyuncuya puan ekleme işlemi yap
        if (turn == 1) {
            Puan++;
            tv_oyuncu1.setText(" OYUNCU: " + Puan);
        }
```

3

Buradaki kod bloklarında 1 numaralı olan kartların hangisinin döndüğünü kontrol eder bu sayede sadece açmak istediğimizin dönmesini sağlarız diğer kartları devre dışı bırakmak içinde **setEnabled(false)** diye tanımladım.
2 numaralı kısımda Handler : Belirli aralıklar ile tekrarlanmasını istediğimiz olayları yönetmemizi sağlayan esnek yapıdır .Benim kodumda süre olduğu için kullandım ve ne kadar süre kullanacağımı da belirttim (500 ms) kısacası ben varsam ne kadar süre çalışacağımı yaz ben o kadar süre çalışacağım demektir aslında.
3 numaralı kısımda doğru yaparsa metodu var bu metot eğer ki kart döndüğünde aynı ise oyuncuya puan eklemesi için yani **2 numaralı kısımdaki** hesap metodunu ekrana verme işlevini görür

```java
public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        ((Button) findViewById(R.id.button)).setOnClickListener((v) -> {
                Intent a = new Intent( packageContext: MainActivity.this, Main2Activity.class);
                startActivity(a);
        });
    }
    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.main_menu,menu);
        return true;
    }
    @Override
    public boolean onOptionsItemSelected(@NonNull MenuItem item) {
       if(item.getItemId()==R.id.ayarlar){
           Toast.makeText( context: this, text: "İNTERNET YOK",Toast.LENGTH_SHORT).show();
       }else  if(item.getItemId()==R.id.yardım){
           Toast.makeText( context: this, text: "YARDIMA İHTİYACIN YOK :D",Toast.LENGTH_SHORT).show();
       }else  if(item.getItemId()==R.id.hakkında){
        Toast.makeText( context: this, text: "VERSİYON 6.91",Toast.LENGTH_SHORT).show();
       }else{
           return super.onOptionsItemSelected(item);
       }
       return true;
```

Toplam ödevde iki adet main kısmı vardır bu main kısımların başlangıç sayfasıdır.

Bu sayfada bir **buton(start isminde)** Tanımladım itent metodu ile start butonuna basınca kullanıcı oyun ekranına gider.

Bir tane de menü tanımladım **menüler** bize oyun veya kullandığımız uygulamada ayarlar, hakkında veya oyuna göre eklenmiş olan internet bağlantısı gibi ikonların ve bağlanmak için ulaşmamız gereken bir seçenek diyebiliriz

getMenuInfalter ile bir tane menü oluşturuyoruz **optionSeleceted** ile neyi seçmek istiyorsak onu karar verip **toast mesajını** oyun ekranına veya işlevi ne ise onu yapıyor