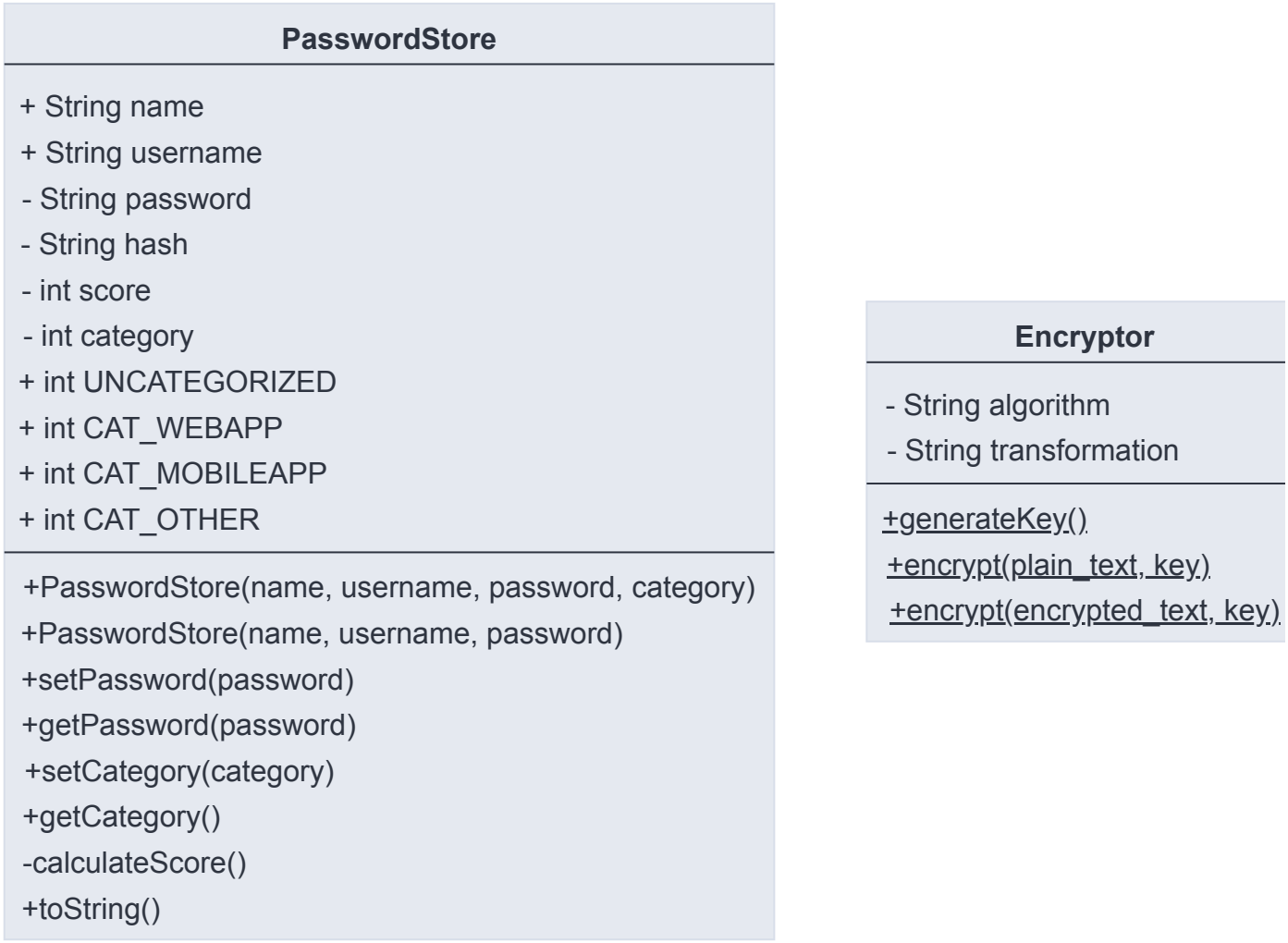
Praktikum PBO : Modul 3

Pada praktikum ini, kita membuat pondasi dari aplikasi penyimpanan password dengan membuat 1 `class Encryptor` untuk melakukan enkripsi terhadap teks dengan menggunakan algoritma AES, serta `class PasswordStore` sebagai class representasi penyimpanan data password.

Berikut adalah class diagram penggambaran kedua class tersebut:



Class `Encryptor`

Terdiri dari 2 attribute private attribute untuk konfigurasi algoritma enkripsi yang digunakan dan 3 static method untuk pemanggilan fungsi enkripsi secara langsung, tanpa instantiasi class. Berikut adalah implementasi koding Java dari class tersebut.

```
package PV.evolution.util;

import javax.crypto.Cipher;
import javax.crypto.KeyGenerator;
import javax.crypto.SecretKey;
import java.nio.charset.StandardCharsets;
import java.security.NoSuchAlgorithmException;
import java.util.Base64;
import javax.crypto.spec.SecretKeySpec;

public class Encryptor {
```

```
    private final static String algorithm = "AES";
    private final static String transformation = "AES/ECB/PKCS5Padding";

    public static String generateKey() throws NoSuchAlgorithmException{
        KeyGenerator key = KeyGenerator.getInstance(algorithm);
        SecretKey secret = key.generateKey();
        String encodedKey = Base64.getEncoder().encodeToString(secret.getEncoded());
        return encodedKey;
    }

    public static String encrypt(String plainText, String key) throws Exception {
        byte[] decodedKey = Base64.getDecoder().decode(key);
        SecretKey originalKey = new SecretKeySpec(decodedKey, 0,
                                                  decodedKey.length, algorithm);

        Cipher cipher = Cipher.getInstance(transformation);
        cipher.init(Cipher.ENCRYPT_MODE, originalKey);
        byte[] encryptedBytes = cipher.doFinal(plainText.getBytes(StandardCharsets.UTF_8));
        String encryptedText = Base64.getEncoder().encodeToString(encryptedBytes);

        return encryptedText;
    }

    public static String decrypt(String encodedText, String key) throws Exception {
        byte[] decodedKey = Base64.getDecoder().decode(key);
        SecretKey originalKey = new SecretKeySpec(decodedKey, 0,
                                                  decodedKey.length, algorithm);

        Cipher cipher = Cipher.getInstance(transformation);
        cipher.init(Cipher.DECRYPT_MODE, originalKey);
        byte[] encryptedBytes = Base64.getDecoder().decode(encodedText);
        byte[] decryptedBytes = cipher.doFinal(encryptedBytes);

        String decryptedText = new String(decryptedBytes,
                                          StandardCharsets.UTF_8);
        return decryptedText;
    }

}
```

Cara penggunaan class tersebut adalah dengan generate key, serta menyimpannya untuk kemudian digunakan pada fungsi enkripsi dan dekripsi. Berikut adalah contoh cara penggunaannya.

```
String hashkey = Encryptor.generateKey();
String plainPass = "KodeRahasiaNegara";
String encryptedPass;
encryptedPass = Encryptor.encrypt(plainPass, hashkey);
System.out.println(Encryptor.decrypt(encryptedPass, hashkey));
```

> **Catatan**:
>
> - Karena fungsi `generateKey`, `encrypt`, dan `decrypt` menimbulkan exception, pada pemanggilannya kita harus membungkus dengan `try` ... `catch` atau melakukan `throw` dengan exception yang sesuai.

Class `PasswordStore`

Class ini merupakan representasi penyimpanan data password yang terdiri dari 6 atribute yaitu:

- `String name` : untuk menyimpan judul akun yang tersimpan
- `String username` : untuk menyimpan username dari akun
- `String password` : untuk menyimpan password terenkripsi dari akun
- `String hashkey` : untuk menyimpan teks key yang digunakan untuk enkripsi dan dekripsi
- `double score` : untuk menyimpan skor keamanan dari password yang tersimpan
- `int category` : untuk memyimpan kategori akun yang tersimpan

> **Catatan:**
>
> - Nilai dari atribut *public* dapat diakses (baca/tulis) langsung melalui *object* yang dibuat
> - Sedangkan akses (baca/tulis) nilai atribut *private* akan dijelaskan pada fungsi getter dan setter-nya

Selain atribute tersebut, class `PasswordStore` juga memiliki beberapa konstanta static untuk memudahkan pengisian nilai `category` yaitu:

```
public static final int UNCATEGORIZED = 0;
public static final int CAT_WEBAPP = 1;
public static final int CAT_MOBILEAPP = 2;
public static final int CAT_OTHER = 3;
```

Class `PasswordStore` juga memiliki 8 (delapan) method yang masing-masIng akan djelaskan fungsinya pada bagian selanjutnya di bawah ini.

Constructor `PasswordStore(name, username, plainPass, category)`

- Melakukan generateKey dan menyimpan hasil generate-nya ke dalam attribut `hashkey`
- Mengisi atribut `name` dan `username` berdasarkan parameter yang diberikan
- Mengeset password menggunakan method `setPassword()` dari parameter `plainPass`
- Mengeset kategori menggunakan method `setCategory()` dari parameter `category`

Constructor `PasswordStore(name, username, plainPass)`

- Melakukan aksi yang sama persis dengan constructor sebelumnya
- Namun pada pemanggilan `setCategory()` diberikan nilai `UNCATEGORIZED`

Method `setPassword(plainPass)`

- Melakukan enkripsi terhadap parameter `plainPass` dengan key dari `hashkey` menggunakan method static `Encryptor.encrypt()`, serta menyimpan hasilnya pada atribut `password`.
- Memanggil method `calculateScore()` untuk mengkalkulasi tingkat keamanan password.

Method `getPassword()`

- Memanggil method static `Encryptor.decrypt()` menggunakan atribut `password` dan `hashkey` serta mengembalikan hasilnya.

Method `setCategory(category)`

- Hanya ada 4 kategori yang tersedia, sehingga method harus mengecek apakah kategori yang diinputkan merupakan nilai antara `0 -- 3`, jika iya makan atribut `category` diset sesuai parameter tersebut.
- Jika tidak makan atribut `category` akan diset nilanya = `0`

Method `getCategory()`

- Mengembalikan teks dari kategori dengan ketentuan sebagai berikut:

- `0` = `"Belum terkategori"`
- `1` = `"Aplikasi web"`
- `2` = `"Aplikasi mobile"`
- `3` = `"Akun lainnya"`

Method `calculateScore(String plainPass)`

- Menghitung score kamanan password secara sederhana dengan ketentuan:
  - Jika panjang password lebih dari 15 maka skornya adalah `10`
  - Jika panjang password kurang dari maka skornya adalah `(panjang / 15) * 10`

Method `toString()`

- Mengembalikan representasi String dari object PasswordStore dengan ketentuan harus mengandung beberapa atribut berikut:
  - Username
  - Password (encrypted)
  - Hashkey
  - Kategori
  - Score

Deklarasi class `PasswordStore` tanpa implementasi method-nya adalah sebagai berikut:

```
package PV.evolution.models;

import PV.evolution.util.Encryptor;
import java.security.NoSuchAlgorithmException;
import java.util.logging.Level;
import java.util.logging.Logger;

public class PasswordStore {
    public String name, username;
    private String password, hashkey;
    private double score;
    private int category;

    public static final int UNCATEGORIZED = 0;
    public static final int CAT_WEBAPP = 1;
    public static final int CAT_MOBILEAPP = 2;
    public static final int CAT_OTHER = 3;

    public PasswordStore(String name, String username, String plainPass){

    }

    public PasswordStore(String name, String username, String plainPass, int category){

    }

    public void setPassword(String plainPass){

    }

    public String getPassword(){
```

```
        return null;
    }

    public void setCategory(int category){

    }

    public String getCategory(){
        return "";
    }

    private void calculateScore(String plainPass){

    }

    @Override
    public String toString() {
        return "";
    }
}
```

Adapun contoh penggunaan class tersebut / pemanggilannya adalah sebagai berikut:

```
PasswordStore pass1 = new PasswordStore("Akun BCA", "1122334455", "RahasiaNegara");
pass1.setCategory(PasswordStore.CAT_WEBAPP);
System.out.println(pass1);
System.out.println("USERNAME: "+pass1.username+" PASSWORD: "+pass1.getPassword());
```

Buatlah implementasi instantiasi class `PasswordStore` beserta method yang relevan pada fungsi `main(String [] args)`, selanjutkan jalankan program dan pastikan tidak ada error yang terjadi.