PBO

TWITTER / X API Oauth 1.0a dengan Java

A11.2022.14795 - NUR IKHSANUDIN

```
private static String CONSUMER_KEY  = "CONSUMER_KEY";
private static String CONSUMER_SECRET  = "CONSUMER_SECRET";
private static String ACCESS_TOKEN  = "ACCESS_TOKEN";
private static String ACCESS_TOKEN_SECRET  = "ACCESS_TOKEN_SE
```

OAuth 1.0 credentials

Didapatkan dengan mendaftar ke https://developer.x.com/ kemudian melakukan regenerate untuk Keys dan Token.

```
private Map<String, String> generateOAuthParams() {
        Map<String, String> params = new HashMap<>();
        params.put("oauth_consumer_key", CONSUMER_KEY);
        params.put("oauth_token", ACCESS_TOKEN);
        params.put("oauth_signature_method", "HMAC-SHA1");
        params.put("oauth_timestamp", String.valueOf(System.
        params.put("oauth_nonce", generateNonce());
        params.put("oauth_version", "1.0");
        return params;
    }

private static String generateNonce() {
        String characters = "abcdefghijklmnopqrstuvwxyzABCDE
        Random rand = new Random();
        StringBuilder nonceBuilder = new StringBuilder();
        for (int i = 0; i < 32; i++) {
            nonceBuilder.append(characters.charAt(rand.nextI
        }
```

```
        return nonceBuilder.toString();
    }
```

Meng-Generate parameter OAuth yang diperlukan untuk otentikasi. Parameter-parameter ini mencakup consumer key, access token, signature method, timestamp, nonce, and version.

```
private String generateOAuthSignature(String requestUrl, Stri
        // Combine the request parameters and OAuth paramete
        Map<String, String> allParams = new HashMap<>(oauthP

        // Sort the parameters alphabetically by key
        String[] sortedKeys = allParams.keySet().toArray(new
        Arrays.sort(sortedKeys);

        // Construct the parameter string
        StringBuilder paramBuilder = new StringBuilder();
        for (String key : sortedKeys) {
            if (paramBuilder.length() > 0) {
                paramBuilder.append("&");
            }
            paramBuilder.append(key).append("=").append(allP
        }
        String parameterString = paramBuilder.toString();

        // Construct the base string
        String baseString = requestMethod + "&" + encode(req

        // Construct the signing key
        String signingKey = encode(CONSUMER_SECRET) + "&" +

        // Generate the HMAC-SHA1 signature
        Mac mac = Mac.getInstance("HmacSHA1");
        SecretKey secretKey = new SecretKeySpec(signingKey.g
        mac.init(secretKey);
        byte[] baseStringBytes = baseString.getBytes("UTF-8"
        byte[] signatureBytes = mac.doFinal(baseStringBytes)
        String signature = new String(Base64.getEncoder().en
```

```
        return signature;
    }
```

Generate OAuth Signature

Menggunakan metode generateOAuthSignature. Ini melibatkan penggabungan request parameters and OAuth parameters, mengurutkannya secara alfabet, membangun base string, dan menghasilkan signature HMAC-SHA1.

```
private String buildOAuthHeader(Map<String, String> oauthPara
        StringBuilder headerBuilder = new StringBuilder();
        headerBuilder.append("OAuth ");

        List<String> encodedParams = new ArrayList<>();

        encodedParams.add("oauth_consumer_key=\"" + encode(o
        encodedParams.add("oauth_token=\"" + encode(oauthPar
        encodedParams.add("oauth_signature_method=\"" + enco
        encodedParams.add("oauth_timestamp=\"" + encode(oaut
        encodedParams.add("oauth_nonce=\"" + encode(oauthPar
        encodedParams.add("oauth_version=\"" + encode(oauthP
        encodedParams.add("oauth_signature=\"" + encode(sign

        String header = String.join(", ", encodedParams);
        headerBuilder.append(header);

        return headerBuilder.toString();
    }

private String encode(String value) {
        try {
            return URLEncoder.encode(value, StandardCharsets
        } catch (UnsupportedEncodingException e) {
            throw new RuntimeException("Failed to encode par
        }
    }
```

Membuat OAuth Authorization Header

Dengan parameter OAuth dan signature, kemudina membangun header otorisasi OAuth menggunakan metode buildOAuthHeader. Header ini mencakup semua parameter OAuth yang telah dienkripsi dan generated signature.

Contoh Header :

```
 --header 'authorization: OAuth oauth_consumer_key="CONSUMER_
```

```
public static void sendTweet(String body) throws IOException
        String tweetUrl = "https://api.twitter.com/2/tweets";

        // Generate the OAuth parameters
        Map<String, String> oauthParams = OAuth.generateOAuth

        // Generate the OAuth signature
        String signature = OAuth.generateOAuthSignature(tweet

        // Build the OAuth authorization header
        String oauthHeader = OAuth.buildOAuthHeader(oauthPara

        // System.out.println(oauthHeader);

                // Membuat HTTP Request
        URL url = new URL(tweetUrl);
        HttpURLConnection connection = (HttpURLConnection) ur
        connection.setRequestMethod("POST");
        connection.setRequestProperty("Authorization", oauthH
        connection.setRequestProperty("Content-Type", "applic
        connection.setDoOutput(true);

        OutputStreamWriter writer = new OutputStreamWriter(co
        writer.write(body);
        writer.flush();

        int responseCode = connection.getResponseCode();
        System.out.println("Response code: " + responseCode);
```

```
            writer.close();
            connection.disconnect();
        }
```

Set up HTTP Connection

Twitter API endpoint : https://api.twitter.com/2/tweets digunakan untuk request upload status ke X

Mengatur metode request sebagai "POST", dan menambahkan header otorisasi OAuth serta jenis konten yang akan dikirim, dalam hal ini adalah JSON.
Data Tweet yang ingin dikirimkan diatur dalam variabel body dan dikirimkan melalui koneksi yang telah dibuka.
Menghandle response, respons dari server Twitter diterima untuk mengetahui apakah Tweet berhasil dikirim atau tidak. Jika mendapatkan kode 200 maka Tweet berhasil dikirim.