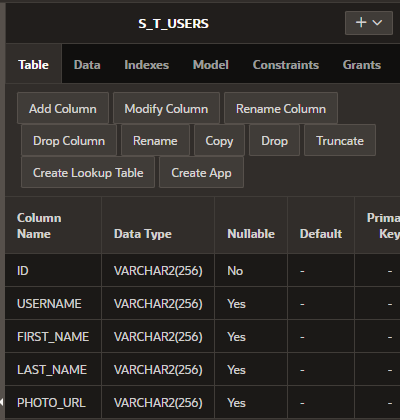
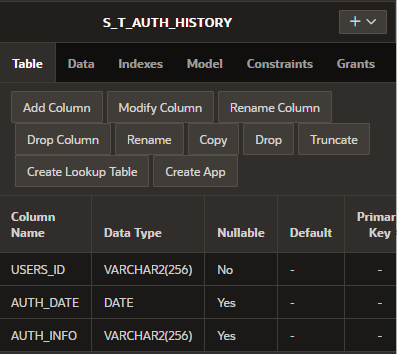
## Аутентификация ORACLE APEX через Wiget Telegram

* Создай бота телеграмм Через @BotFather
* Добавь свой Domain в телеграм бота
* Сгенерируй виджет на <https://core.telegram.org/widgets/login>
* Создай таблицы



* Добавь код на страницу Логина ORACLE APEX

| <script   async src="https://telegram.org/js/telegram-widget.js?19"   data-telegram-login="YOU\_BOT\_NAME\_BOT"   data-size="large"   data-radius="2"   data-onauth="onTelegramAuth(user)"   data-request-access="write"> </script> <script type="text/javascript">  function onTelegramAuth(TG) {  try {  let json = JSON.stringify(TG);  apex.item( "P9999\_USERNAME" ).setValue( TG.id );  apex.item( "P9999\_PASSWORD" ).setValue(json);  apex.submit("LOGIN");   }  catch (e) {  console.log(e);  }   } </script> |
| --- |

* Добавь код аутентификации

| FUNCTION TELEGRAM\_AUTEN (p\_username IN VARCHAR2, p\_password IN NVARCHAR2) RETURN BOOLEAN AS  R\_SECRET\_KEY RAW (256) := 'YOU SECRET KEY SHA256 BOT TOKEN';   L\_TG\_AUTH\_DATE VARCHAR2(256) ;  L\_TG\_FIRST\_NAME VARCHAR2(256) ;  L\_TG\_ID VARCHAR2(256) ;  L\_TG\_LAST\_NAME VARCHAR2(256) ;  L\_TG\_PHOTO\_URL VARCHAR2(256) ;  L\_TG\_USERNAME VARCHAR2(256) ;  L\_TG\_HASH RAW (256) ;  L\_USER\_COUNT NUMBER :=0 ;   V\_TG\_AUTH\_DATE VARCHAR2(256) ;  V\_TG\_FIRST\_NAME VARCHAR2(256) ;  V\_TG\_ID VARCHAR2(256) ;  V\_TG\_LAST\_NAME VARCHAR2(256) ;  V\_TG\_PHOTO\_URL VARCHAR2(256) ;  V\_TG\_USERNAME VARCHAR2(256) ;   V\_TG\_DATA VARCHAR2(512) ;  V\_HMAC\_TG\_DATA RAW (256) ; BEGIN *-- Парсим JSON из p\_password*  APEX\_JSON.parse(p\_password) ;  L\_TG\_AUTH\_DATE := APEX\_JSON.get\_varchar2(p\_path => 'auth\_date') ;  L\_TG\_FIRST\_NAME := APEX\_JSON.get\_varchar2(p\_path => 'first\_name') ;  L\_TG\_ID := APEX\_JSON.get\_varchar2(p\_path => 'id') ;  L\_TG\_LAST\_NAME := APEX\_JSON.get\_varchar2(p\_path => 'last\_name') ;  L\_TG\_PHOTO\_URL := APEX\_JSON.get\_varchar2(p\_path => 'photo\_url') ;  L\_TG\_USERNAME := APEX\_JSON.get\_varchar2(p\_path => 'username') ;  L\_TG\_HASH := APEX\_JSON.get\_varchar2(p\_path => 'hash') ;   *-- Проверяем актуальность данных что L\_TG\_AUTH\_DATE не старше 172800 секунд (48 часов) и p\_username = L\_TG\_ID* *-- В случае актуальности данных формируем строку проверки*  IF (((CURRENT\_DATE - DATE '1970-01-01') \* 60 \* 60 \* 24) - 172800) < L\_TG\_AUTH\_DATE AND p\_username = L\_TG\_ID  THEN    *--\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* НАЧАЛО: ДАННАЯ ЧАСТЬ КОДА ТРЕБУЕТ ОПТИМИЗАЦИИ \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*--*  V\_TG\_AUTH\_DATE := 'auth\_date=' || L\_TG\_AUTH\_DATE ;  V\_TG\_ID := CHR(10) || 'id=' || L\_TG\_ID ;  *-- Проверяем и выкидываем пустые значения из списка*  IF L\_TG\_FIRST\_NAME IS NULL THEN V\_TG\_FIRST\_NAME := NULL ;  ELSE  V\_TG\_FIRST\_NAME := CHR(10) || 'first\_name=' || L\_TG\_FIRST\_NAME ;  END IF ;  IF L\_TG\_LAST\_NAME IS NULL THEN V\_TG\_LAST\_NAME := NULL ;  ELSE  V\_TG\_LAST\_NAME := CHR(10) || 'last\_name=' || L\_TG\_LAST\_NAME ;  END IF ;  IF L\_TG\_PHOTO\_URL IS NULL THEN V\_TG\_PHOTO\_URL := NULL ;  ELSE  V\_TG\_PHOTO\_URL := CHR(10) || 'photo\_url=' || L\_TG\_PHOTO\_URL ;  END IF ;  IF L\_TG\_USERNAME IS NULL THEN V\_TG\_USERNAME := NULL ;  ELSE  V\_TG\_USERNAME := CHR(10) || 'username=' || L\_TG\_USERNAME ;  END IF ;  *-- Формируем строчку для проверки HMAC*   V\_TG\_DATA := V\_TG\_AUTH\_DATE || V\_TG\_FIRST\_NAME || V\_TG\_ID || V\_TG\_LAST\_NAME || V\_TG\_PHOTO\_URL || V\_TG\_USERNAME ;  dbms\_output.put\_line(V\_TG\_DATA) ;  *--\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* КОНЕЦ: ДАННАЯ ЧАСТЬ КОДА ТРЕБУЕТ ОПТИМИЗАЦИИ \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*--*   *-- Формируем HMAC из V\_TG\_DATA с секретным ключем R\_SECRET\_KEY (sha256 токена бота)*  V\_HMAC\_TG\_DATA := dbms\_crypto.mac(utl\_raw.cast\_to\_raw(V\_TG\_DATA), 3, R\_SECRET\_KEY) ;    *-- Проверям сходятся ли HASH и HMAC, если сходятся, аутентифицируем пользователя под p\_username*  IF V\_HMAC\_TG\_DATA = L\_TG\_HASH   THEN  SELECT count(\*) INTO L\_USER\_COUNT  FROM S\_T\_USERS  WHERE S\_T\_USERS.ID = p\_username;   IF L\_USER\_COUNT > 0   THEN  UPDATE S\_T\_USERS  SET USERNAME = L\_TG\_USERNAME,  FIRST\_NAME = L\_TG\_FIRST\_NAME,  LAST\_NAME = L\_TG\_LAST\_NAME,  PHOTO\_URL = L\_TG\_PHOTO\_URL  WHERE ID = p\_username;    INSERT INTO S\_T\_AUTH\_HISTORY (  USERS\_ID,   AUTH\_DATE,   AUTH\_INFO  )  VALUES (  L\_TG\_ID,   CURRENT\_DATE,   owa\_util.get\_cgi\_env('X-FORWARDED-FOR') || CHR(10) || OWA\_UTIL.GET\_CGI\_ENV('HTTP\_USER\_AGENT')   );  ELSE   INSERT INTO S\_T\_USERS (  ID,  USERNAME,   FIRST\_NAME,   LAST\_NAME,   PHOTO\_URL  )  VALUES (  L\_TG\_ID,   L\_TG\_USERNAME,   L\_TG\_FIRST\_NAME,   L\_TG\_LAST\_NAME,   L\_TG\_PHOTO\_URL  );   INSERT INTO S\_T\_AUTH\_HISTORY (  USERS\_ID,   AUTH\_DATE,   AUTH\_INFO  )  VALUES (  L\_TG\_ID,   CURRENT\_DATE,   owa\_util.get\_cgi\_env('X-FORWARDED-FOR') || CHR(10) || OWA\_UTIL.GET\_CGI\_ENV('HTTP\_USER\_AGENT')   );   RETURN TRUE;  END IF ;   RETURN TRUE;   ELSE  RETURN FALSE;  END IF ;  ELSE  RETURN FALSE;  END IF ;  END TELEGRAM\_AUTEN; |
| --- |
|  |
|  |
|  |

ps. Отладочный код

DECLARE

p\_username VARCHAR2(512) := '5170982684' ;

p\_password VARCHAR2(512) := '{"id":5170982684,"first\_name":"Максим","photo\_url":"https://t.me/i/userpic/320/1PD\_ERtqfoS\_qBVAx2SNYM-AtdClJGU7rv6ssoze2F2Erg3bbQhBGXx8TvndTuvh.jpg","auth\_date":1651710189,"hash":"5f836e640a4f9651eb3a26c134878105fbdecf0c0729bfe4fdc69d2d41beaecb"}';

R\_SECRET\_KEY RAW (512) := '\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*';

L\_TG\_AUTH\_DATE VARCHAR2(512) ;

L\_TG\_FIRST\_NAME VARCHAR2(512) ;

L\_TG\_ID VARCHAR2(512) ;

L\_TG\_LAST\_NAME VARCHAR2(512) ;

L\_TG\_PHOTO\_URL VARCHAR2(512) ;

L\_TG\_USERNAME VARCHAR2(512) ;

L\_TG\_HASH RAW (512) ;

L\_USER\_COUNT NUMBER :=0 ;

V\_TG\_AUTH\_DATE VARCHAR2(512) ;

V\_TG\_FIRST\_NAME VARCHAR2(512) ;

V\_TG\_ID VARCHAR2(512) ;

V\_TG\_LAST\_NAME VARCHAR2(512) ;

V\_TG\_PHOTO\_URL VARCHAR2(512) ;

V\_TG\_USERNAME VARCHAR2(512) ;

V\_TG\_DATA VARCHAR2(512) ;

V\_HMAC\_TG\_DATA RAW (512) ;

BEGIN

-- Парсим JSON из p\_password

APEX\_JSON.parse(p\_password);

L\_TG\_AUTH\_DATE := APEX\_JSON.get\_varchar2(p\_path => 'auth\_date') ;

L\_TG\_FIRST\_NAME := APEX\_JSON.get\_varchar2(p\_path => 'first\_name') ;

L\_TG\_ID := APEX\_JSON.get\_varchar2(p\_path => 'id') ;

L\_TG\_LAST\_NAME := APEX\_JSON.get\_varchar2(p\_path => 'last\_name') ;

L\_TG\_PHOTO\_URL := APEX\_JSON.get\_varchar2(p\_path => 'photo\_url') ;

L\_TG\_USERNAME := APEX\_JSON.get\_varchar2(p\_path => 'username') ;

L\_TG\_HASH := APEX\_JSON.get\_varchar2(p\_path => 'hash') ;

-- Проверяем актуальность данных что L\_TG\_AUTH\_DATE не старше 172800 секунд (48 часов) и p\_username = L\_TG\_ID

-- В случае актуальности данных формируем строку проверки

IF (((CURRENT\_DATE - DATE '1970-01-01') \* 60 \* 60 \* 24) - 172800) < L\_TG\_AUTH\_DATE AND p\_username = L\_TG\_ID

THEN

dbms\_output.put\_line('ДАННЫЕ АКТУАЛЬНЫ: TRUE') ;

--\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* НАЧАЛО: ДАННАЯ ЧАСТЬ КОДА ТРЕБУЕТ ОПТИМИЗАЦИИ \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*--

V\_TG\_AUTH\_DATE := 'auth\_date=' || L\_TG\_AUTH\_DATE ;

V\_TG\_ID := CHR(10) || 'id=' || L\_TG\_ID ;

-- Проверяем и выкидываем пустые значения из списка

IF L\_TG\_FIRST\_NAME IS NULL THEN V\_TG\_FIRST\_NAME := NULL ;

ELSE

V\_TG\_FIRST\_NAME := CHR(10) || 'first\_name=' || L\_TG\_FIRST\_NAME ;

END IF ;

IF L\_TG\_LAST\_NAME IS NULL THEN V\_TG\_LAST\_NAME := NULL ;

ELSE

V\_TG\_LAST\_NAME := CHR(10) || 'last\_name=' || L\_TG\_LAST\_NAME ;

END IF ;

IF L\_TG\_PHOTO\_URL IS NULL THEN V\_TG\_PHOTO\_URL := NULL ;

ELSE

V\_TG\_PHOTO\_URL := CHR(10) || 'photo\_url=' || L\_TG\_PHOTO\_URL ;

END IF ;

IF L\_TG\_USERNAME IS NULL THEN V\_TG\_USERNAME := NULL ;

ELSE

V\_TG\_USERNAME := CHR(10) || 'username=' || L\_TG\_USERNAME ;

END IF ;

-- Формируем строчку для проверки HMAC

V\_TG\_DATA := V\_TG\_AUTH\_DATE || V\_TG\_FIRST\_NAME || V\_TG\_ID || V\_TG\_LAST\_NAME || V\_TG\_PHOTO\_URL || V\_TG\_USERNAME ;

dbms\_output.put\_line(V\_TG\_DATA) ;

--\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* КОНЕЦ: ДАННАЯ ЧАСТЬ КОДА ТРЕБУЕТ ОПТИМИЗАЦИИ \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*--

-- Формируем HMAC из V\_TG\_DATA с секретным ключем R\_SECRET\_KEY (sha256 токена бота)

V\_HMAC\_TG\_DATA := dbms\_crypto.mac(utl\_raw.cast\_to\_raw(V\_TG\_DATA), 3, R\_SECRET\_KEY) ;

dbms\_output.put\_line('HMAC: ' || V\_HMAC\_TG\_DATA) ;

dbms\_output.put\_line('HASH: ' || L\_TG\_HASH) ;

-- Проверям сходятся ли HASH и HMAC, если сходятся, проверяем наличие пользователя если нет создаем если есть то обновляем и аутентифицируем пользователя под p\_username

IF V\_HMAC\_TG\_DATA = L\_TG\_HASH

THEN

SELECT count(\*) INTO L\_USER\_COUNT

FROM S\_T\_USERS

WHERE S\_T\_USERS.ID = p\_username;

dbms\_output.put\_line('L\_USER\_COUNT: ' || L\_USER\_COUNT) ;

IF L\_USER\_COUNT > 0

THEN

UPDATE S\_T\_USERS

SET USERNAME = L\_TG\_USERNAME,

FIRST\_NAME = L\_TG\_FIRST\_NAME,

LAST\_NAME = L\_TG\_LAST\_NAME,

PHOTO\_URL = L\_TG\_PHOTO\_URL

WHERE ID = p\_username;

INSERT INTO S\_T\_AUTH\_HISTORY (

USERS\_ID,

AUTH\_DATE,

AUTH\_INFO

)

VALUES (

L\_TG\_ID,

CURRENT\_DATE,

owa\_util.get\_cgi\_env('X-FORWARDED-FOR') || CHR(10) || OWA\_UTIL.GET\_CGI\_ENV('HTTP\_USER\_AGENT')

);

ELSE

INSERT INTO S\_T\_USERS (

ID,

USERNAME,

FIRST\_NAME,

LAST\_NAME,

PHOTO\_URL

)

VALUES (

L\_TG\_ID,

L\_TG\_USERNAME,

L\_TG\_FIRST\_NAME,

L\_TG\_LAST\_NAME,

L\_TG\_PHOTO\_URL

);

INSERT INTO S\_T\_AUTH\_HISTORY (

USERS\_ID,

AUTH\_DATE,

AUTH\_INFO

)

VALUES (

L\_TG\_ID,

CURRENT\_DATE,

owa\_util.get\_cgi\_env('X-FORWARDED-FOR') || CHR(10) || OWA\_UTIL.GET\_CGI\_ENV('HTTP\_USER\_AGENT')

);

dbms\_output.put\_line('ТАКОГО ПОЛЬЗОВАТЕЛЯ НЕ БЫЛО. СОЗДАЛИ НОВОГО: ' || p\_username ) ;

END IF ;

dbms\_output.put\_line('ВЫ АВТОРИЗОВАЛИСЬ ПОД: ' || p\_username ) ;

ELSE

dbms\_output.put\_line('ОШИБКА СРАВНЕНИЯ HASH и HMAC ') ;

END IF ;

ELSE

dbms\_output.put\_line('ДАННЫЕ АКТУАЛЬНЫ: FALSE') ;

END IF ;

END;

<script

async src="https://telegram.org/js/telegram-widget.js?19"

data-telegram-login="\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"

data-size="large"

data-radius="2"

data-onauth="onTelegramAuth(user)"

data-request-access="write">

</script>

<script type="text/javascript">

function onTelegramAuth(TG) {

try {

apex.item( "P9999\_TG\_ID" ).setValue( TG.id );

apex.item( "P9999\_TG\_USERNAME" ).setValue( TG.username );

apex.item( "P9999\_TG\_FIRST\_NAME" ).setValue( TG.first\_name );

apex.item( "P9999\_TG\_LAST\_NAME" ).setValue( TG.last\_name );

apex.item( "P9999\_TG\_PHOTO\_URL" ).setValue( TG.photo\_url );

apex.item( "P9999\_TG\_AUTH\_DATE" ).setValue( TG.auth\_date );

apex.item( "P9999\_TG\_HASH" ).setValue( TG.hash );

let json = JSON.stringify(TG);

apex.item( "P9999\_TG\_DATA2" ).setValue(json);

apex.item( "P9999\_USERNAME" ).setValue( TG.id );

apex.item( "P9999\_PASSWORD" ).setValue(json);

let input\_hash = TG.hash;

delete TG.hash;

const TG\_DATA = Object.keys(TG)

.sort()

.map(k => (`${k}=${TG[k]}`))

.join('\n');

apex.item( "P9999\_TG\_DATA" ).setValue(TG\_DATA);

}

catch (e) {

console.log(e);

}

}

</script>