WELCOME\_S

DECLARE

correct\_pw varchar2(20);

username\_count number(3,0);

sta\_type varchar2(20);

Begin

If :WELCOME\_S.STA\_USERNAME\_TXB is null THEN

--SET\_ITEM\_PROPERTY('WELCOME\_S.STA\_USERNAME\_TXB',UPDATE\_ALLOWED, PROPERTY\_FALSE);

MESSAGE('INVALID USERNAME & PASSWORD MATCH');

MESSAGE('INVALID USERNAME & PASSWORD MATCH');

:WELCOME\_S.STA\_USERNAME\_TXB := NULL;

:WELCOME\_S.STA\_PSW\_TXB := NULL;

RAISE FORM\_TRIGGER\_FAILURE;

ELSIF

:WELCOME\_S.STA\_PSW\_TXB is null THEN

MESSAGE('INVALID USERNAME & PASSWORD MATCH');

MESSAGE('INVALID USERNAME & PASSWORD MATCH');

:WELCOME\_S.STA\_USERNAME\_TXB := NULL;

:WELCOME\_S.STA\_PSW\_TXB := NULL;

RAISE FORM\_TRIGGER\_FAILURE;

ELSE

--IF :WELCOME\_S.STA\_USERNAME\_TXB IS NOT NULL THEN

SELECT COUNT(\*) INTO username\_count from STAFF

where STAFF\_UNAME = :WELCOME\_S.STA\_USERNAME\_TXB;

IF username\_count =0 then

MESSAGE('INVALID USERNAME & PASSWORD MATCH');

MESSAGE('INVALID USERNAME & PASSWORD MATCH');

:WELCOME\_S.STA\_USERNAME\_TXB := NULL;

:WELCOME\_S.STA\_PSW\_TXB := NULL;

RAISE FORM\_TRIGGER\_FAILURE;

ELSE

Select STAFF\_PW into correct\_pw from STAFF

where STAFF\_UNAME=:WELCOME\_S.STA\_USERNAME\_TXB;

IF correct\_pw <> :WELCOME\_S.STA\_PSW\_TXB THEN

MESSAGE('INVALID USERNAME & PASSWORD MATCH');

MESSAGE('INVALID USERNAME & PASSWORD MATCH');

:WELCOME\_S.STA\_USERNAME\_TXB := NULL;

:WELCOME\_S.STA\_PSW\_TXB := NULL;

RAISE FORM\_TRIGGER\_FAILURE;

END IF;

Select STAFF\_TYPE INTO sta\_type From STAFF

where STAFF\_UNAME=:WELCOME\_S.STA\_USERNAME\_TXB;

END IF;

END IF;

IF correct\_pw=:WELCOME\_S.STA\_PSW\_TXB AND sta\_type = 'Cashier' THEN

Go\_Block('Main\_S');

Elsif correct\_pw=:WELCOME\_S.STA\_PSW\_TXB AND sta\_type = 'Barista' THEN

Go\_Block('MAIN\_BAR');

ElsiF correct\_pw=:WELCOME\_S.STA\_PSW\_TXB AND sta\_type = 'Manager' THEN

Go\_Block('DASH\_S');

Else

MESSAGE('INVALID USERNAME & PASSWORD MATCH');

MESSAGE('INVALID USERNAME & PASSWORD MATCH');

:WELCOME\_S.STA\_USERNAME\_TXB := NULL;

:WELCOME\_S.STA\_PSW\_TXB := NULL;

RAISE FORM\_TRIGGER\_FAILURE;

END IF;

END;

FORGET\_PW\_S -> RESET

DECLARE

staff\_id number(20,0);

correct\_ans varchar2(100);

staff\_id\_count number(3,0);

RETURN\_VALUE NUMBER(20);

duplicate\_pwd number(3);

Begin

-- Handle not enter all info

IF :STAFF\_ID\_T IS NULL OR :ANSWER\_T IS NULL OR :NEW\_PW\_T IS NULL OR :RE\_NEW\_PW\_T IS NULL THEN

SET\_ALERT\_PROPERTY('ALERT\_ORDERS', ALERT\_MESSAGE\_TEXT, 'Please enter all information');

RETURN\_VALUE := SHOW\_ALERT('ALERT\_ORDERS');

RAISE FORM\_TRIGGER\_FAILURE;

-- ELSE NULL;

END IF;

IF :STAFF\_ID\_T IS NOT NULL THEN

SELECT COUNT(\*) INTO staff\_id\_count from STAFF

where STAFF\_ID =:STAFF\_ID\_T;

END IF;

IF staff\_id\_count = 0 then

SET\_ALERT\_PROPERTY('ALERT\_ORDERS', ALERT\_MESSAGE\_TEXT, 'Please enter correct staff ID');

RETURN\_VALUE := SHOW\_ALERT('ALERT\_ORDERS');

RAISE FORM\_TRIGGER\_FAILURE;

ELSE

-- Check whether staff ID and answer are matched

Select STAFF\_ID, STAFF\_ANS

INTO staff\_id, correct\_ans

FROM STAFF

WHERE STAFF\_ID = :FORGET\_PW\_S.STAFF\_ID\_T;

If staff\_id = :FORGET\_PW\_S.STAFF\_ID\_T and correct\_ans <> :FORGET\_PW\_S.ANSWER\_T THEN

SET\_ALERT\_PROPERTY('ALERT\_ORDERS', ALERT\_MESSAGE\_TEXT, 'Please fill the correct answer for safety question.');

RETURN\_VALUE := SHOW\_ALERT('ALERT\_ORDERS');

RAISE FORM\_TRIGGER\_FAILURE;

--ELSE NULL;

End if;

IF LENGTH (:FORGET\_PW\_S.NEW\_PW\_T) <5 THEN

SET\_ALERT\_PROPERTY('ALERT\_ORDERS', ALERT\_MESSAGE\_TEXT, 'Password is too short. Please write more than 5.');

RETURN\_VALUE := SHOW\_ALERT('ALERT\_ORDERS');

RAISE FORM\_TRIGGER\_FAILURE;

--ELSE NULL;

End if;

IF LENGTH (:FORGET\_PW\_S.NEW\_PW\_T) >20 THEN

SET\_ALERT\_PROPERTY('ALERT\_ORDERS', ALERT\_MESSAGE\_TEXT, 'Password is too long. Please write less than 20.');

RETURN\_VALUE := SHOW\_ALERT('ALERT\_ORDERS');

RAISE FORM\_TRIGGER\_FAILURE;

--ELSE NULL;

End if;

IF :FORGET\_PW\_S.NEW\_PW\_T != :FORGET\_PW\_S.RE\_NEW\_PW\_T then

SET\_ALERT\_PROPERTY('ALERT\_ORDERS', ALERT\_MESSAGE\_TEXT, 'New Password and Retypre Password should be consistent');

RETURN\_VALUE := SHOW\_ALERT('ALERT\_ORDERS');

RAISE FORM\_TRIGGER\_FAILURE;

--ELSE NULL;

END IF;

SELECT COUNT(\*) INTO duplicate\_pwd FROM STAFF

WHERE STAFF\_PW = :FORGET\_PW\_S.NEW\_PW\_T;

IF duplicate\_pwd >0 THEN

SET\_ALERT\_PROPERTY('ALERT\_ORDERS', ALERT\_MESSAGE\_TEXT, 'New Password should be different from the Old Password');

RETURN\_VALUE := SHOW\_ALERT('ALERT\_ORDERS');

RAISE FORM\_TRIGGER\_FAILURE;

-- ELSE NULL;

END IF;

UPDATE STAFF

SET STAFF\_PW = :FORGET\_PW\_S.NEW\_PW\_T

WHERE STAFF\_ID = :FORGET\_PW\_S.STAFF\_ID\_T;

SET\_ALERT\_PROPERTY('ALERT\_ORDERS', ALERT\_MESSAGE\_TEXT, 'Password is reset successfully');

RETURN\_VALUE := SHOW\_ALERT('ALERT\_ORDERS');

forms\_ddl('Commit');

END IF;

CLEAR\_BLOCK(no\_validate);

END;

FORGET\_PW\_S -> GO TO MAIN PAGE

GO\_BLOCK('FORGET\_PW\_S');

CLEAR\_BLOCK(no\_validate);

GO\_BLOCK('WELCOME\_S');

REGISTER\_S → INSERT

DECLARE

MEM\_UNAME\_COUNT NUMBER(3);

MEM\_Phone\_COUNT NUMBER(3);

current\_id number(20); --current\_id is the var. that stores the current number of records in the table

new\_id number(20); --new\_id is the var. that stores the 'current\_id +1'

BEGIN

SELECT COUNT(\*) INTO MEM\_UNAME\_COUNT FROM MEMBER

WHERE MEM\_UNAME = :REGISTER\_S.MEM\_UNAME;

IF MEM\_UNAME\_COUNT >0 THEN

MESSAGE('This username had already been occupied! Please use another one !');

MESSAGE('This username had already been occupied! Please use another one !');

RAISE FORM\_TRIGGER\_FAILURE;

:REGISTER\_S.MEM\_UNAME := NULL;

END IF;

If :REGISTER\_S.MEM\_NAME is NULL or :REGISTER\_S.MEM\_UNAME is NULL or :REGISTER\_S.MEM\_PW is NULL

or :REGISTER\_S.MEM\_PHONE is NULL or :REGISTER\_S.MEM\_ANS is NULL Then

Message('Please fill in the missing information!');

MESSAGE('Please fill in the missing information!');

RAISE FORM\_TRIGGER\_FAILURE;

-- Else NULL;

End if;

IF LENGTH (:REGISTER\_S.MEM\_PW) >20 THEN

message('Password is too long. Please write less than 20 character/number');

message('Password is too long. Please write less than 20 character/number');

RAISE FORM\_TRIGGER\_FAILURE;

-- ELSE NULL;

End if;

IF LENGTH (:REGISTER\_S.MEM\_PW) <5 THEN

message('Password is too short. Please write more than 5 character/number');

message('Password is too short. Please write more than 5 character/number');

RAISE FORM\_TRIGGER\_FAILURE;

-- ELSE NULL;

End if;

IF LENGTH (TRIM(TRANSLATE(:REGISTER\_S.MEM\_PHONE, '+0123456789', ' '))) IS NOT NULL THEN

message('Phone number should not contain any non-numeric value.');

message('Phone number should not contain any non-numeric value.');

RAISE FORM\_TRIGGER\_FAILURE;

-- ELSE NULL;

END IF;

SELECT COUNT(\*) INTO MEM\_Phone\_COUNT FROM MEMBER

WHERE MEM\_PHONE =:REGISTER\_S.MEM\_PHONE;

IF MEM\_Phone\_COUNT >0 THEN

MESSAGE('This phone number had already been occupied! Please use another one !');

MESSAGE('This phone number had already been occupied! Please use another one !');

RAISE FORM\_TRIGGER\_FAILURE;

-- ELSE NULL;

END IF;

select max(MEM\_ID) into current\_id from MEMBER;

new\_id := current\_id +1;

insert into MEMBER (MEM\_ID,MEM\_NAME,MEM\_UNAME,MEM\_PW,MEM\_PHONE,MEM\_ANS,MEM\_STAR,MEM\_FAV)

VALUES (new\_id,

:REGISTER\_S.MEM\_NAME,

:REGISTER\_S.MEM\_UNAME,

:REGISTER\_S.MEM\_PW,

:REGISTER\_S.MEM\_PHONE,

:REGISTER\_S.MEM\_ANS,

0,

:REGISTER\_S.MEM\_FAV);

--Display message after inserted

message('The Member ' ||:REGISTER\_S.MEM\_NAME || ' has been inserted to the database sucessfully!');

message('The Member ' ||:REGISTER\_S.MEM\_NAME || ' has been inserted to the database sucessfully!');

--save the change to the database

FORMS\_DDL('COMMIT');

CLEAR\_BLOCK(no\_validate);

END;

REGISTER\_S → BACK TO MAIN PAGE

BEGIN

CLEAR\_BLOCK(no\_validate);

go\_block('MAIN\_S');

END;