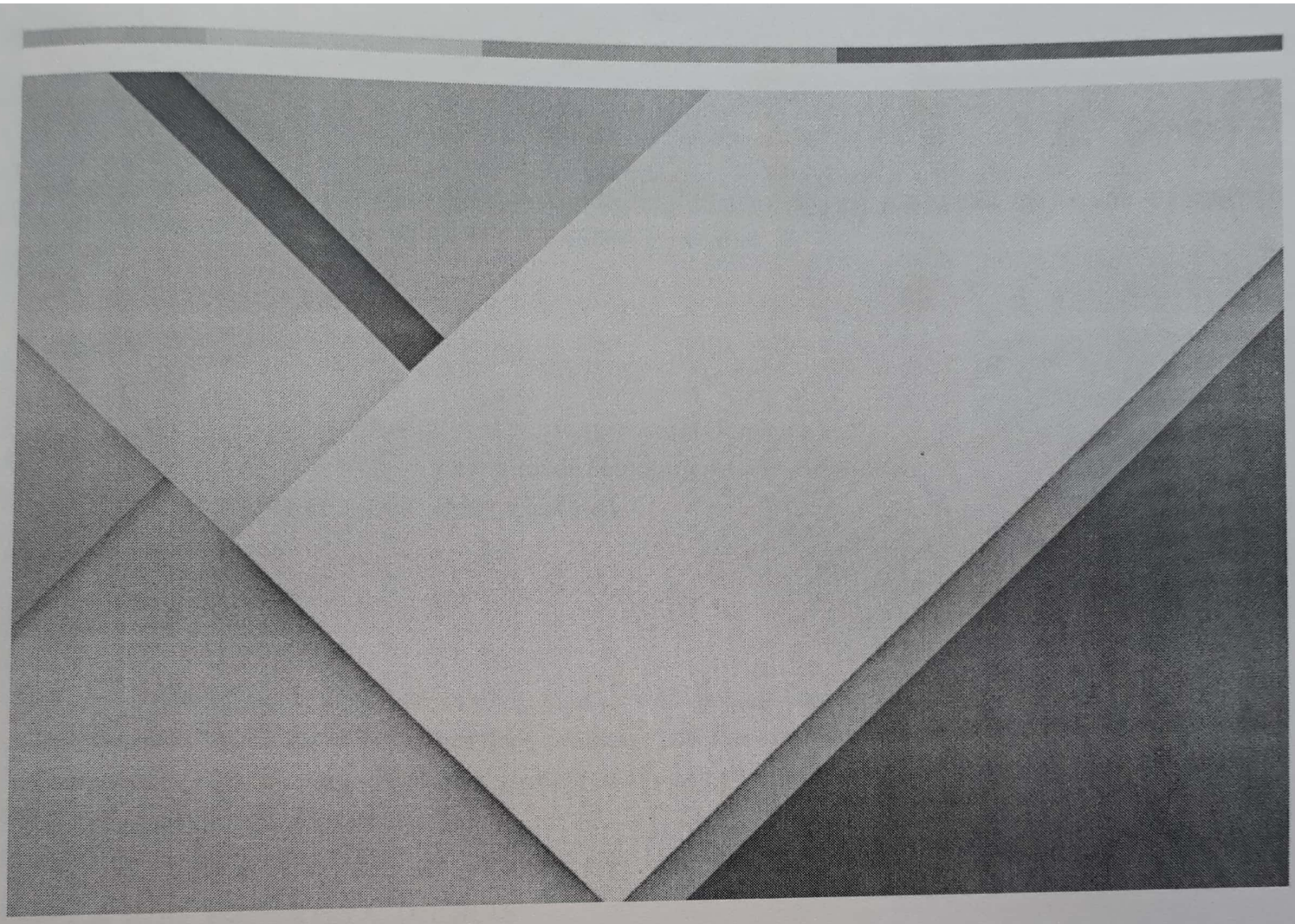


TRUECALLER

Scanned by CamScanner

Overview

This project aims to abolish anonymity from the world by allowing people to look up for any unknown numbers they wish to find by using this app called Truecaller.

Goals

1. Create a database for all global working cellular numbers.
2. Allow users to search for any number belonging to any individual.
3. Make users aware of incoming spam calls.

Specifications

The app works by consent of user to share their as well as their contacts details to the app. Fundamentally it hold records for all users as well as their contacts. The user can then search for any number available in this database. The app uses user's approximate locations to locate the same. Users are allowed to set up their personal profile information.

Spam and Blocked Contacts

The annoying repeating incoming calls to advert can be minimized or blocked by this feature.

Milestones

I. Connect to the world

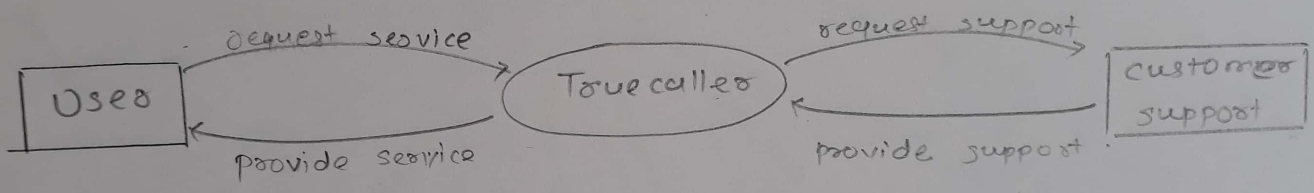
It's like maintaining transparency among all the users of the world if the app is prevalent in most parts.

II. No anonymity

Receive no more unknown calls from anyone. Relieve yourself from it if any.

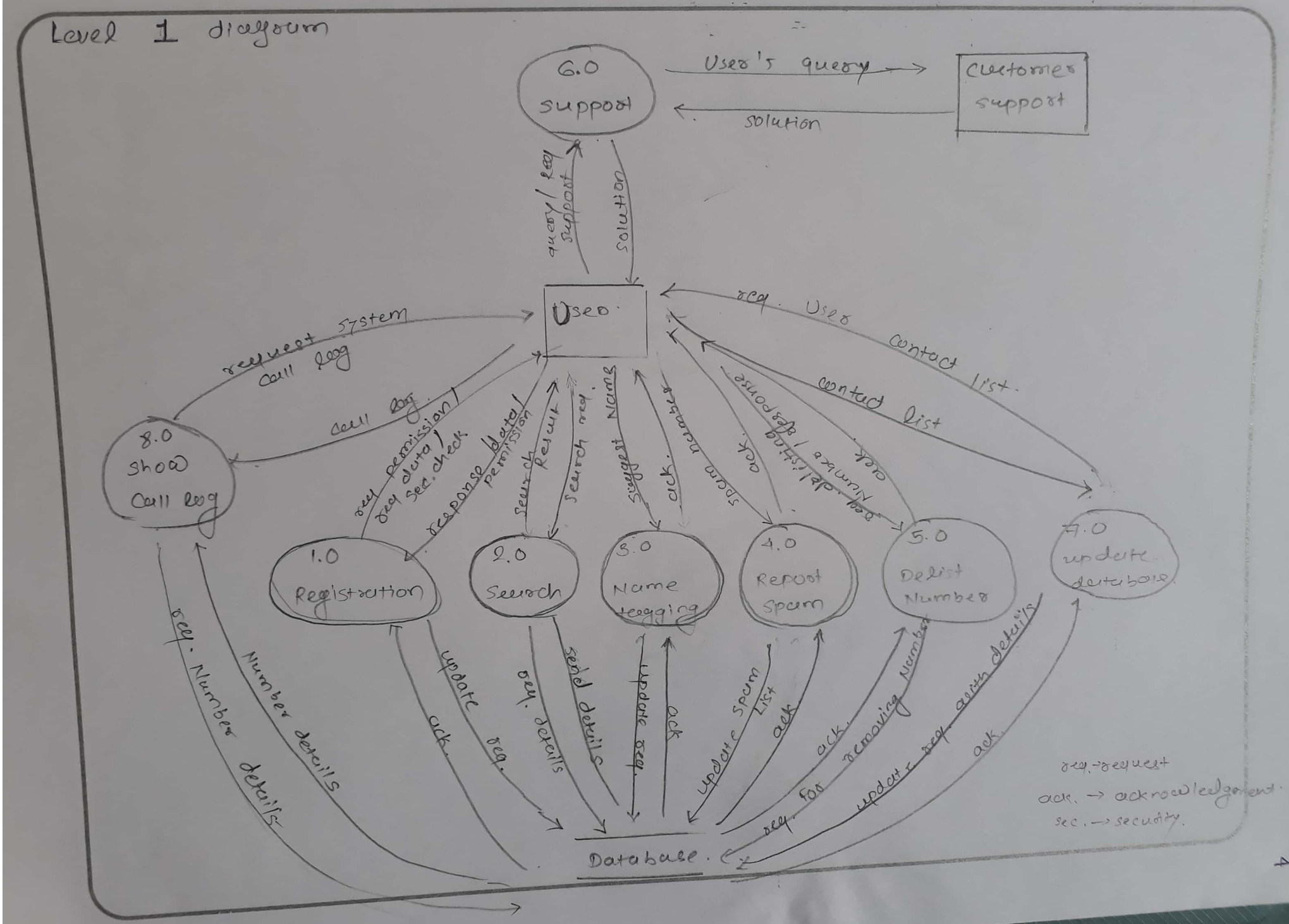
Scanned by CamScanner

Context diagram :- Level - 0



Scanned by CamScanner

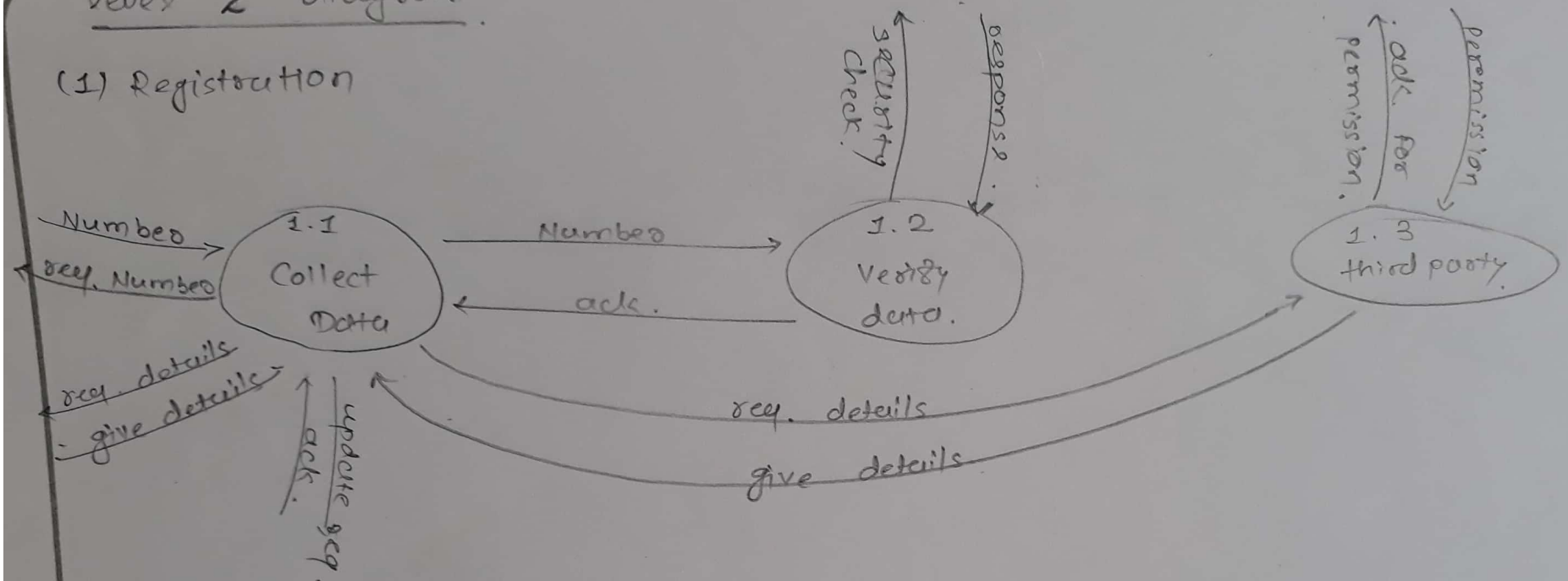
Level 1 diagram



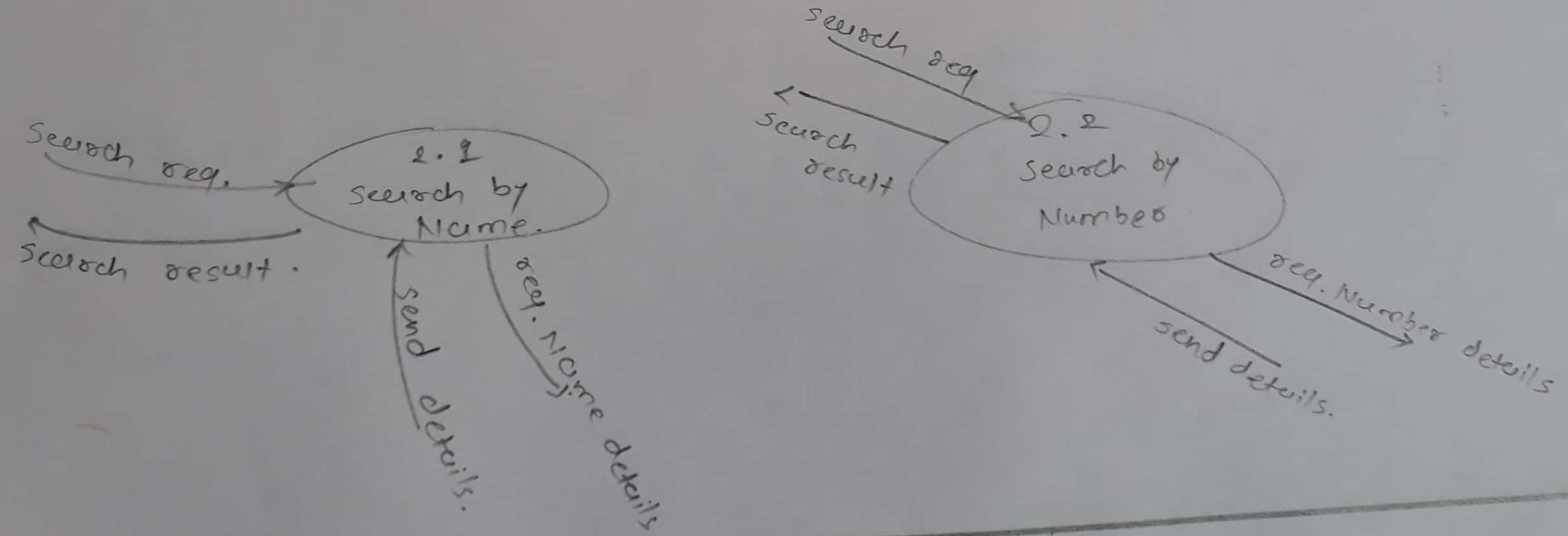
Scanned by CamScanner

Level 2 diagram.

(1) Registration

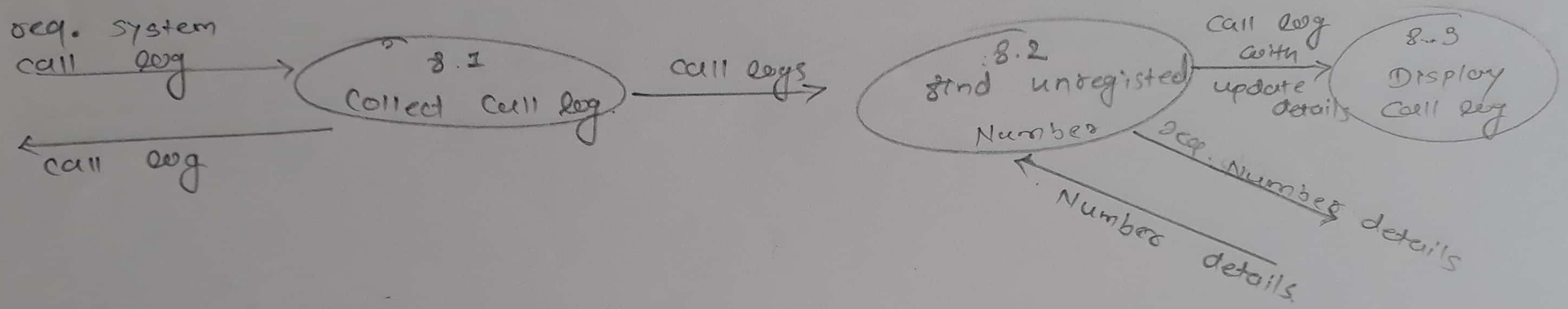


(2) Search.

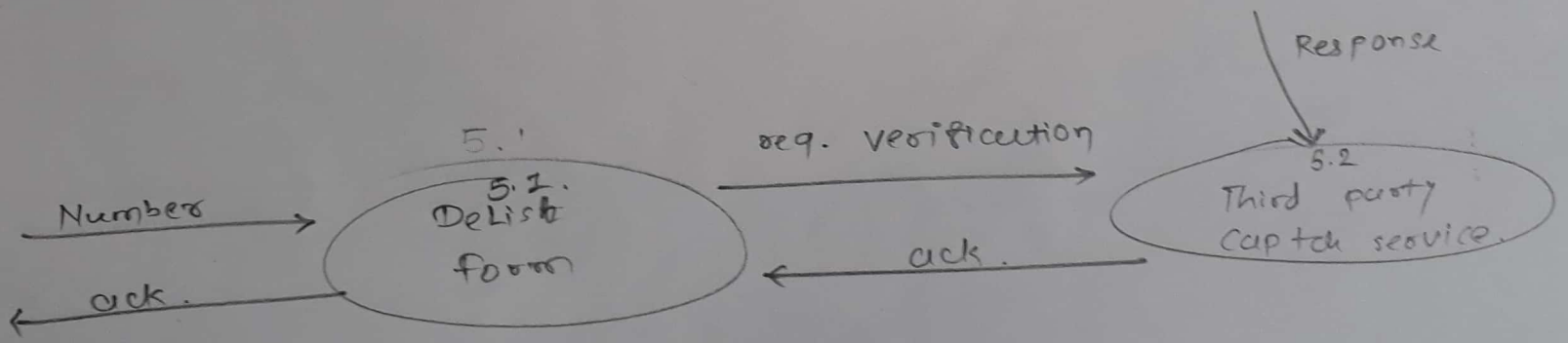


Scanned by CamScanner

(3) Show call log

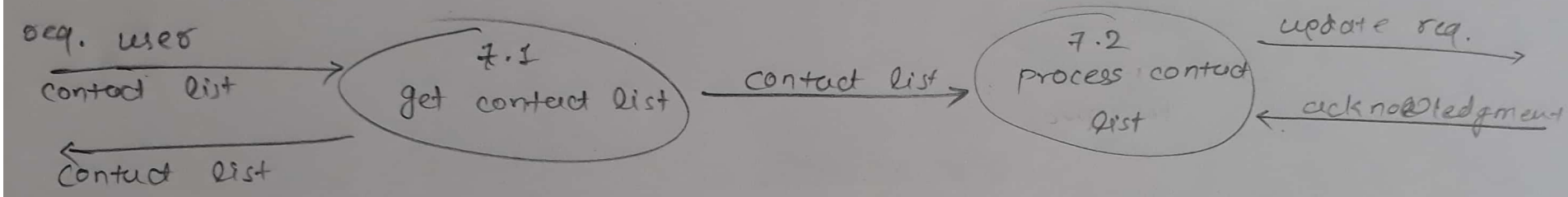


(4) DeList Number

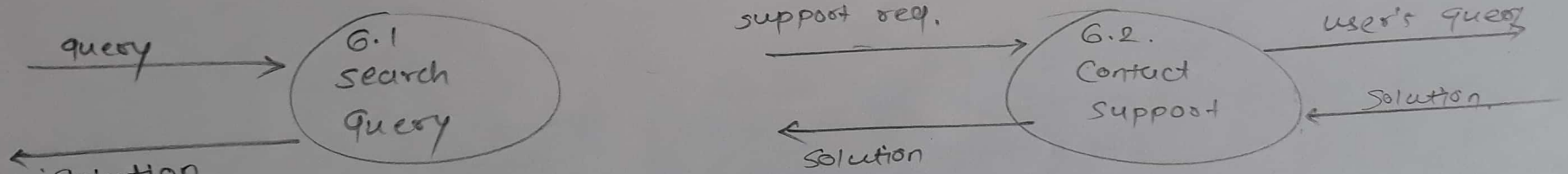


Scanned by CamScanner

(5) update database.

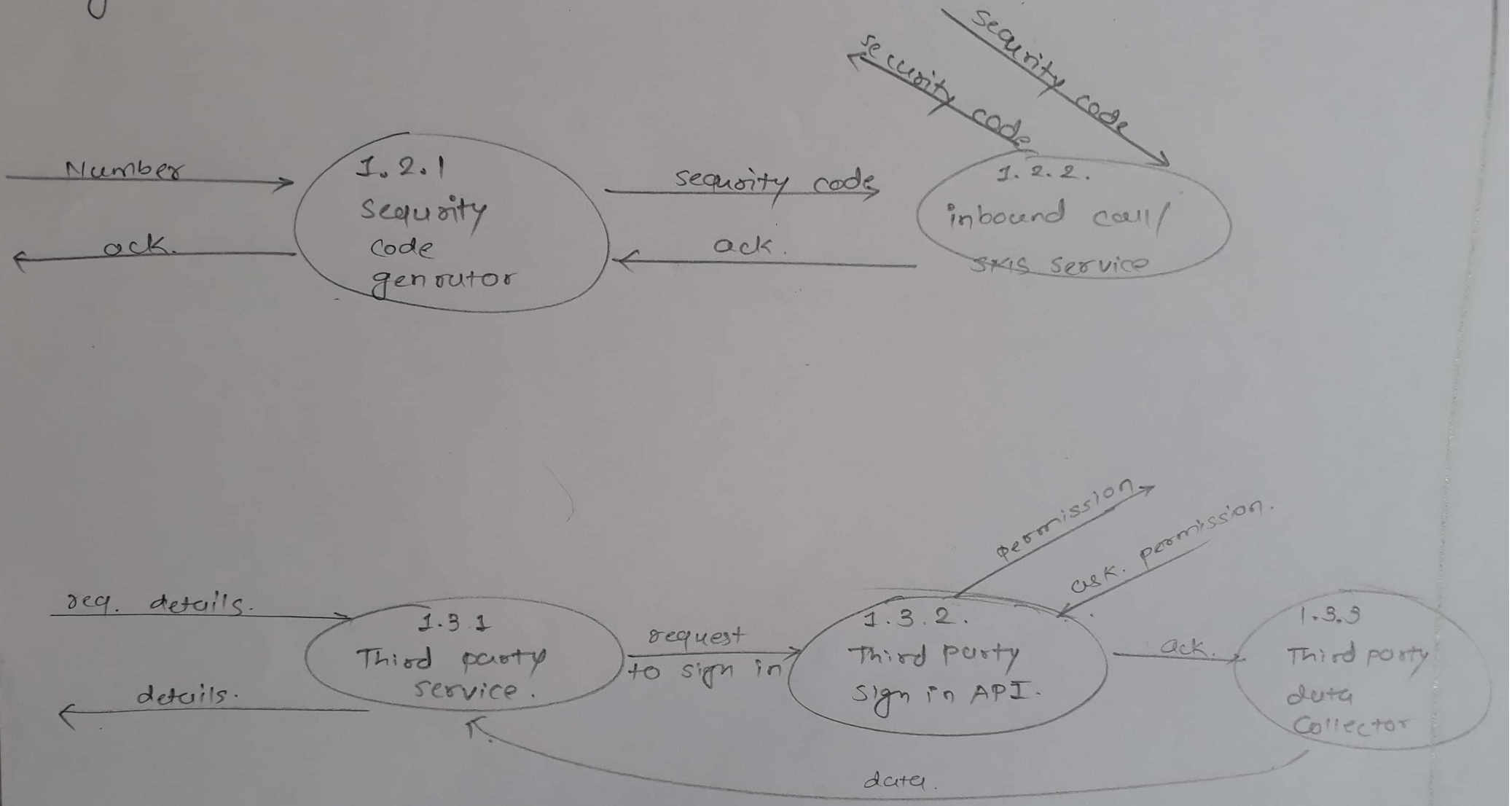


(6) support



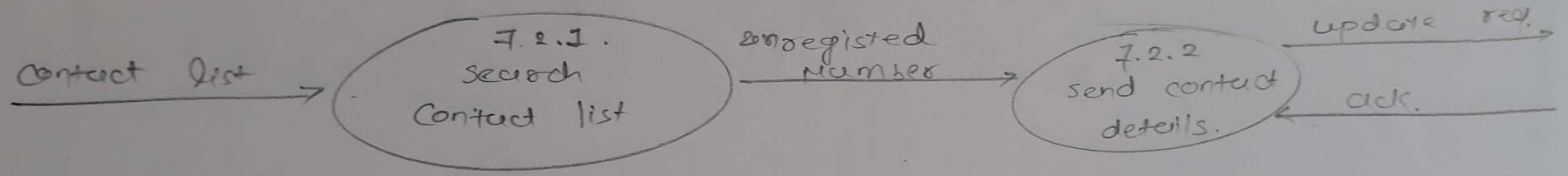
Scanned by CamScanner

(1) Registration.



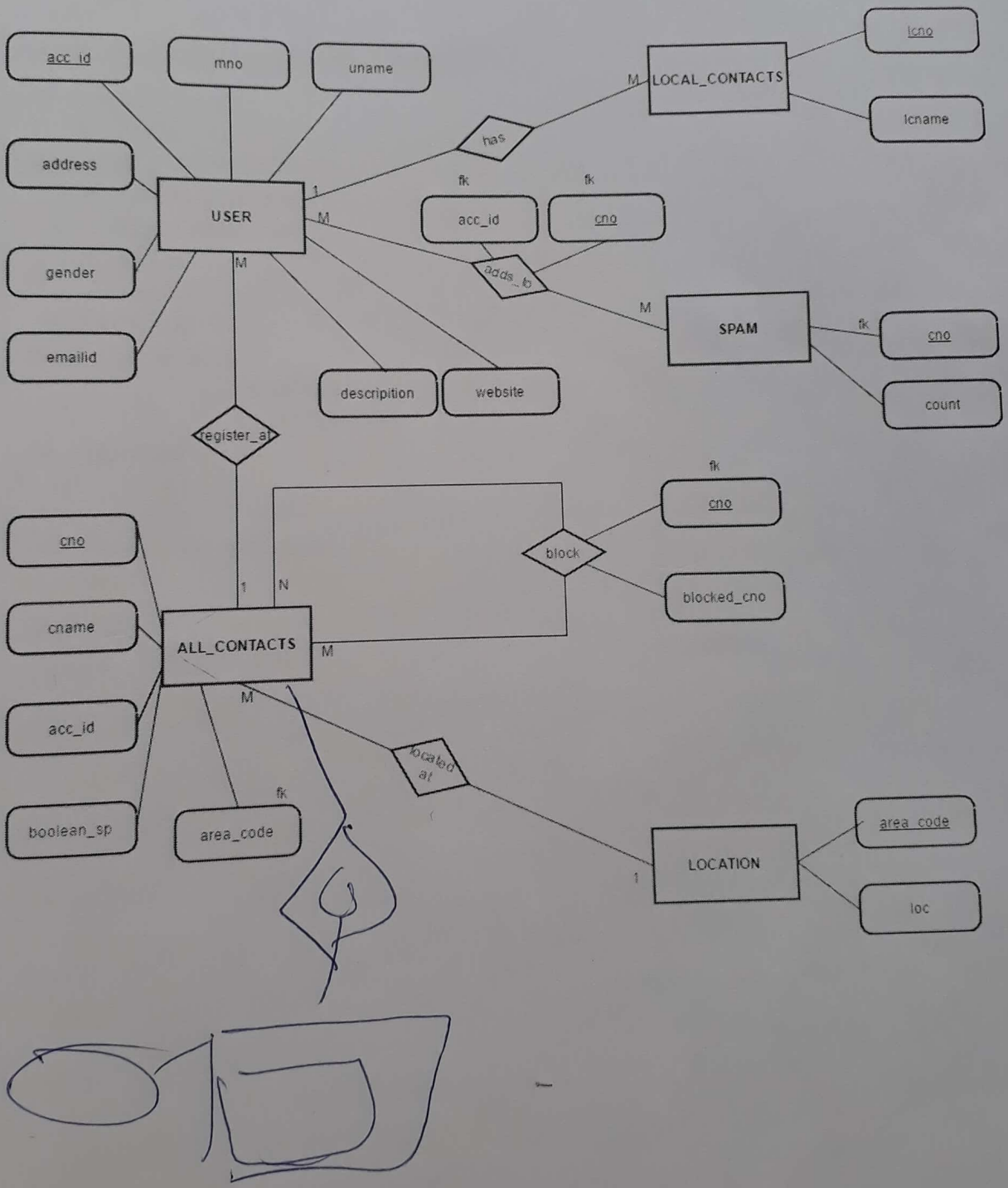
Scanned by CamScanner

update database



Scanned by CamScanner

ER DIAGRAM



Scanned by CamScanner

DATA DICTIONARY

Definition: A data dictionary contains description and defines concerning the data structure, data elements, their relationships and other characteristics of a system.

TRUECALLER DATA DICTIONARY:

- TABLES
  - ADDS_TO
  - ALL_CONTACTS
  - LOCAL_CONTACTS
  - LOCATION
  - SPAM
  - USER1
- PACKAGE
  - PCK
    - PROCEDURE
      - FINDAC
- TRIGGER
  - T1
  - T2
  - REG_TRIGGER

Scanned by CamScanner

Object Type **TABLE** Object **ADDS_TO**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
ADDS_TO	ACC_ID	NUMBER	-	5	0	1	-	-	-
	CNO	VARCHAR2	12	-	-	2	-	-	-

Object Type **TABLE** Object **ALL_CONTACTS**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
ALL_CONTACTS	CNO	VARCHAR2	12	-	-	1	-	-	-
	CNAME	VARCHAR2	20	-	-	-	✓	-	-
	AREA_CODE	VARCHAR2	2	-	-	-	✓	-	-
	ACC_ID	NUMBER	-	5	0	-	✓	-	-
	BOOLEAN_SP	NUMBER	-	1	0	-	✓	-	-

Object Type **TABLE** Object **LOCAL_CONTACTS**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
LOCAL_CONTACTS	LCNO	VARCHAR2	12	-	-	1	-	-	-
	LNAME	VARCHAR2	20	-	-	-	✓	-	-

Object Type **TABLE** Object **LOCATION**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
LOCATION	AREA_CODE	VARCHAR2	2	-	-	1	-	-	-
	LOC	VARCHAR2	20	-	-	-	✓	-	-

1 - 2

Object Type **TABLE** Object **SPAM**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
SPAM	CNO	VARCHAR2	12	-	-	1	-	-	-
	COUNT	NUMBER	-	5	0	-	✓	-	-

Scanned by CamScanner

Object Type **TABLE** Object **USER1**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
USER1	ACC_ID	NUMBER	-	5	0	1	-	-	-
	MNO	VARCHAR2	12	-	-	-	✓	-	-
	UNAME	VARCHAR2	20	-	-	-	✓	-	-
	EMAILID	VARCHAR2	20	-	-	-	✓	-	-
	DESCRIPTION	VARCHAR2	50	-	-	-	✓	-	-
	WEBSITE	VARCHAR2	30	-	-	-	✓	-	-
	ADDRESS	VARCHAR2	30	-	-	-	✓	-	-
	GENDER	VARCHAR2	10	-	-	-	✓	-	-

QUERY FOR ALL TABLES

- CREATE TABLE ADDS_TO
  (
  ACC_ID NUMBER(5) CONSTRAINT FK_ACCID REFERENCES USER1,
  CNO VARCHAR2(12) CONSTRAINT FK_CNO REFERENCES ALL_CONTACTS,
  PRIMARY KEY(ACC_ID,CNO)
)

- CREATE TABLE ALL_CONTACTS
  (
  CNO VARCHAR2(12) CONSTRAINT PK_CNO PRIMARY KEY,
  CNAME VARCHAR2(20),
  AREA_CODE VARCHAR(2) CONSTRAINT FK_AREA_CODE REFERENCES LOCATION,
  ACC_ID NUMBER(5) CONSTRAINT FK_ACC_ID REFERENCES USER1,
  BOOLEAN_SP NUMBER(1)
)
- CREATE TABLE LOCAL_CONTACTS
  (
  LCNO VARCHAR2(12) CONSTRAINT LCNO_PRIMARYKEY PRIMARY KEY,
  LNAME VARCHAR2(20)
)
- CREATE TABLE LOCATION
  (
  AREA_CODE VARCHAR2(2) CONSTRAINT PK_AREA_CODE PRIMARY KEY,
  LOC VARCHAR2(20)
)
- CREATE TABLE SPAM
  (
  CNO VARCHAR2(12) CONSTRAINT FK_SPAM_CNO REFERENCES ALL_CONTACTS,
  COUNT NUMBER(5),
  PRIMARY KEY(CNO))

Scanned by CamScanner

```
CREATE TABLE USER1
(ACC_ID NUMBER(5) CONSTRAINT ACC_ID_PRIMARYKEY PRIMARY KEY,
MNO VARCHAR2(12),
UNAME VARCHAR2(20),
EMAILID VARCHAR2(20),
DESCRIPTION VARCHAR2(50),
WEBSITE VARCHAR2(30),
ADDRESS VARCHAR2(30),
GENDER VARCHAR2(10))
```

PACKAGE SPECIFICAION

```
CREATE OR REPLACE PACKAGE PCK
AS
        PROCEDURE FINDAC(R ALL_CONTACTS.CNO%TYPE,AC OUT
ALL_CONTACTS.AREA_CODE%TYPE);
END PCK;
/
```

PACKAGE BODY

```
CREATE OR REPLACE PACKAGE BODY PCK
AS
        PROCEDURE FINDAC(R ALL_CONTACTS.CNO%TYPE,AC OUT
ALL_CONTACTS.AREA_CODE%TYPE)
    AS
        BEGIN

        AC:=SUBSTR(R,1,2);

        END FINDAC;

END PCK;
/
```

Scanned by CamScanner

TRIGGERS

1. TRIGGER : T1 (INSERT INTO SPAM)

```
CREATE OR REPLACE TRIGGER T1
AFTER INSERT ON ADDS_TO
FOR EACH ROW

DECLARE
N NUMBER(1);

BEGIN
SELECT COUNT(*) INTO N FROM SPAM WHERE CNO=:NEW.CNO;
IF N>0 THEN
        UPDATE SPAM
        SET COUNT=COUNT+1
        WHERE CNO=:NEW.CNO;
ELSE
        INSERT INTO SPAM VALUES(:NEW.CNO,1);
END IF;
END T1;
/
```

2. TRIGGER : T2 (CHECK SPAM BOOL)

```
CREATE OR REPLACE TRIGGER T2
AFTER UPDATE
ON SPAM
FOR EACH ROW

BEGIN
IF :NEW.COUNT>30 THEN
        UPDATE ALL_CONTACTS
        SET BOOLEAN_SP=1
        WHERE CNO=:NEW.CNO;
END IF;

END T2;
/
```

Scanned by CamScanner

3. TRIGGER : REG_TRIGGER

```
CREATE OR REPLACE TRIGGER REG_TRIGGER
AFTER INSERT ON USER1
FOR EACH ROW

DECLARE

CURSOR C1 IS SELECT * FROM LOCAL_CONTACTS;
X NUMBER(1);
X1 NUMBER(2);
AC ALL_CONTACTS.AREA_CODE%TYPE;

A_ID USER1.ACC_ID%TYPE;

BEGIN

SELECT COUNT(*) INTO X
FROM ALL_CONTACTS
WHERE :NEW.MNO=CNO;

IF X!=0 THEN
  DELETE FROM
  ALL_CONTACTS
  WHERE :NEW.MNO=CNO;
END IF;

PCK.FINDAC(:NEW.MNO,AC);

INSERT INTO ALL_CONTACTS
VALUES(:NEW.MNO,:NEW.UNAME,AC,:NEW.ACC_ID,0);


FOR R1 IN C1
LOOP
```

Scanned by CamScanner

```
SELECT COUNT(*) INTO X
FROM ALL_CONTACTS
WHERE CNO=R1.LCNO;

SELECT ACC_ID INTO A_ID
FROM ALL_CONTACTS
WHERE CNO=:NEW.MNO;

IF(X!=0) THEN
   SELECT COUNT(*) INTO X1
   FROM USER1
   WHERE ACC_ID=A_ID;


      IF (X1=0) THEN
         UPDATE ALL_CONTACTS
         SET CNAME=R1.LNAME,ACC_ID=:NEW.ACC_ID
         WHERE CNO=R1.LCNO;

      ELSE
         CONTINUE;
      END IF;

  ELSE
   PCK.FINDAC(R1.LCNO,AC);
   INSERT INTO ALL_CONTACTS
   VALUES(R1.LCNO,R1.LNAME,AC,:NEW.ACC_ID,0);
 END IF;
END LOOP;
END REG_TRIGGER;
/
```

Scanned by CamScanner