Αναφορά Project 1

Ομαδική

Μέλος Α

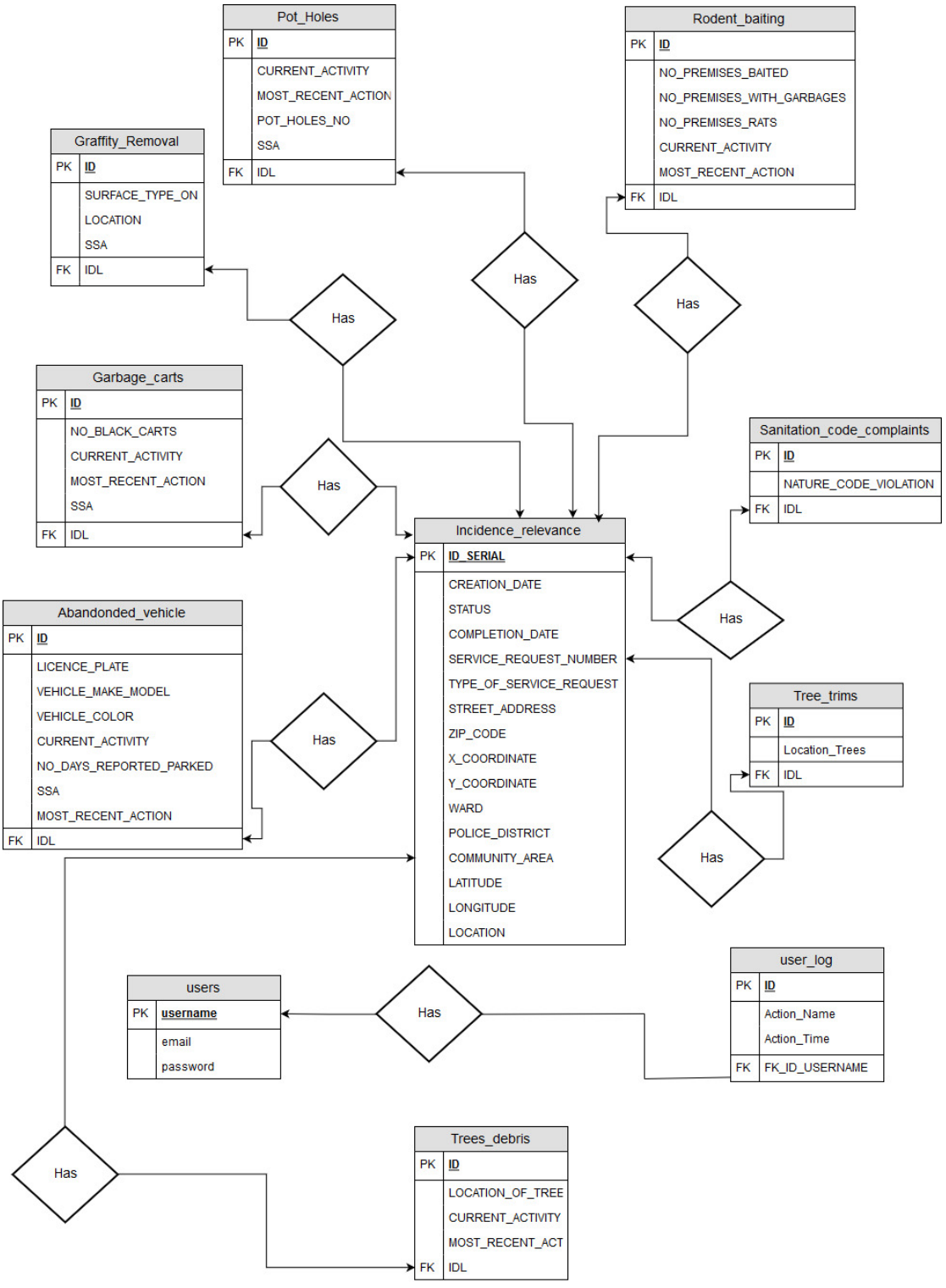
Ονοματεπώνυμο	Νικόλαος Κουτσάκης
ΑΜ	CS1.18.0006
Σχολή	Τμήμα Πληροφορικής & Τηλεπικοινωνιών
Όνομα Μεταπτυχιακού	ΠΜΣ Πληροφορικής
Ειδίκευση	Θεμελιώσεις Πληροφορικής και Εφαρμογές
Ακαδημαϊκό Έτος	2018-2019
Μάθημα	Μ149 Συστήματα Βάσεων Δεδομένων

Μέλος Β

Ονοματεπώνυμο	Αλέξανδρος Φώτιος Ντογραματζής
ΑΜ	CS2.18.0010
Σχολή	Τμήμα Πληροφορικής & Τηλεπικοινωνιών
Όνομα Μεταπτυχιακού	ΠΜΣ Πληροφορικής
Ειδίκευση	Διαχείριση Δεδομένων, Πληροφορίας και Γνώσης
Ακαδημαϊκό Έτος	2018-2019
Μάθημα	Μ149 Συστήματα Βάσεων Δεδομένων

Βάση Δεδομένων

Σχήμα Βάσης Δεδομένων(ER –MODEL)



Ουσιαστικά μαζέψαμε τα κοινά πεδία από όλα τα αρχεία csv σε ένα πίνακα(μια οντότητα) Incidence_relevance και για τα επιπλέον πεδία-στήλες που είχε κάθε αρχείο csv φτιάξαμε ξεχωριστές οντότητες-πίνακες οι οποίοι συνδέονται με την οντότητα Incidence_relevance με σχέση 1-1(το πρωτεύον κλειδί ID του πίνακα αυτού είναι ξένο κλειδί στους άλλους πίνακες). Προφανώς για τα αρχεία που δεν υπήρχε επιπλέον πληροφορία από τα πεδία του Incidence_relevance δεν φτιάχτηκαν ξεχωριστές οντότητες. Υπάρχει ο πίνακας των χρηστών users της εφαρμογής, καθώς και το αρχείο καταγραφής ενεργειών των χρηστών user_log. Κάθε χρήστης κάνει διάφορες ενέργειες καθώς χειρίζεται την εφαρμογή οπότε υπάρχει μια σχέση ανάμεσα στους πίνακες users-user_log ένα προς πολλά(στο user_log FK to PK username της users). Παρακάτω φαίνονται τα script PostgreSQL για την δημιουργία των πινάκων της βάσης δεδομένων μας:

```
CREATE TABLE "Incidence_relevance"
(
  "ID_SERIAL" integer NOT NULL,
  "CREATION_DATE" date,
  "STATUS" character varying(15),
  "COMPLETION_DATE" date,
  "SERVICE_REQUEST_NUMBER" character varying(25),
  "TYPE_OF_SERVICE_REQUEST" character varying,
  "STREET_ADDRESS" character varying(50),
  "ZIP_CODE" character(5),
  "X_COORDINATE" numeric(20,10),
  "Y_COORDINATE" numeric(20,10),
  "WARD" numeric(2,0),
  "POLICE_DISTRICT" numeric(2,0),
  "COMMUNITY_AREA" numeric(2,0),
  "LATITUDE" numeric(19,15),
  "LONGITUDE" numeric(19,15),
  "LOCATION" character varying(120),
  CONSTRAINT "Incidence_relevance_pkey" PRIMARY KEY ("ID_SERIAL"),
  CONSTRAINT "Incidence_relevance_WARD_check" CHECK ("WARD" >= 0::numeric
AND "WARD" <= 50::numeric) NOT VALID,
  CONSTRAINT "Incidence_relevance_POLICE_DISTRICT_check" CHECK
("POLICE_DISTRICT" >= 0::numeric AND "POLICE_DISTRICT" <= 31::numeric) NOT
VALID,
  CONSTRAINT "Incidence_relevance_COMMUNITY_AREA_check" CHECK
("COMMUNITY_AREA" >= 0::numeric AND "COMMUNITY_AREA" <= 77::numeric) NOT
VALID)
```

```
CREATE TABLE "Abandonded_vehicle"
(
   "ID" integer NOT NULL,
   "LICENCE_PLATE" character varying(400),
   "VEHICLE_MAKE_MODEL" character varying(65),
   "VEHICLE_COLOR" character varying(15),
   "CURRENT_ACTIVITY" character varying(25),
   "NO_DAYS_REPORTED_PARKED" double precision,
   "SSA" numeric(2,0),
   "MOST_RECENT_ACTION" character varying(65),
   "IDL" integer NOT NULL DEFAULT
nextval('"Abandonded_vehicle_IDL_seq"'::regclass),
   CONSTRAINT "Abandonded_vehicle_pkey" PRIMARY KEY ("ID"),
   CONSTRAINT "INCIDENCE_RELEVANCE_RELATION" FOREIGN KEY ("IDL")
      REFERENCES public."Incidence_relevance" ("ID_SERIAL") MATCH SIMPLE
      ON UPDATE CASCADE
      ON DELETE CASCADE,
   CONSTRAINT "Abandonded_vehicle_SSA_check" CHECK ("SSA" >= 0::numeric AND
"SSA" <= 69::numeric) NOT VALID)
```

```
CREATE TABLE "Garbage_carts"
(
   "ID" integer NOT NULL,
   "NO_BLACK_CARTS" double precision,
   "CURRENT_ACTIVITY" character varying(25) ,
   "MOST_RECENT_ACTION" character varying(100) ,
   "SSA" numeric(2,0),
   "IDL" integer NOT NULL ,
   CONSTRAINT "Garbage_carts_pkey" PRIMARY KEY ("ID"),
   CONSTRAINT "Garbage_carts_IDL_fkey" FOREIGN KEY ("IDL")
      REFERENCES public."Incidence_relevance" ("ID_SERIAL") MATCH SIMPLE
      ON UPDATE CASCADE
      ON DELETE CASCADE,
```

```
   CONSTRAINT "Garbage_carts_SSA_check" CHECK ("SSA" >= 0::numeric AND "SSA" <=
69::numeric) NOT VALID

)
```

```
CREATE TABLE "Graffity_Removal"

(

  "ID" integer NOT NULL,

   "SURFACE_TYPE_ON" character varying(50),

   "LOCATION" character varying(40) ,

   "SSA" numeric(2,0),

   "IDL" integer NOT NULL,

   CONSTRAINT "Graffity_Removal_pkey" PRIMARY KEY ("ID"),

   CONSTRAINT "Graffity_Removal_IDL_fkey" FOREIGN KEY ("IDL")

      REFERENCES public."Incidence_relevance" ("ID_SERIAL") MATCH SIMPLE

      ON UPDATE CASCADE

      ON DELETE CASCADE,

   CONSTRAINT "Graffity_Removal_SSA_check" CHECK ("SSA" <= 69::numeric AND "SSA"
>= 0::numeric) NOT VALID)
```

```
CREATE TABLE "Pot_Holes"

(

   "ID" integer NOT NULL,

   "CURRENT_ACTIVITY" character varying(25),

   "MOST_RECENT_ACTION" character varying(90),

   "POT_HOLES_NO" double precision,

   "SSA" numeric(2,0),

   "IDL" integer NOT NULL ,

   CONSTRAINT "Pot_Holes_pkey" PRIMARY KEY ("ID"),

   CONSTRAINT "Pot_Holes_IDL_fkey" FOREIGN KEY ("IDL")

      REFERENCES public."Incidence_relevance" ("ID_SERIAL") MATCH SIMPLE

      ON UPDATE CASCADE
```

```
    ON DELETE CASCADE,
  CONSTRAINT "Pot_Holes_SSA_check" CHECK ("SSA" >= 0::numeric AND "SSA" <=
69::numeric) NOT VALID,
  CONSTRAINT "Pot_Holes_POT_HOLES_NO_check" CHECK ("POT_HOLES_NO" >=
0::numeric::double precision) NOT VALID
)
```

```
CREATE TABLE "Rodent_baiting"
(
  "ID" integer NOT NULL ,
  "NO_PREMISES_BAITED" double precision,
  "NO_PREMISES_WITH_GARBAGES" integer,
  "NO_PREMISES_RATS" double precision,
  "CURRENT_ACTIVITY" character varying(50) ,
  "MOST_RECENT_ACTION" character varying(90) ,
  "IDL" integer NOT NULL ,
  CONSTRAINT "Rodent_baiting_pkey" PRIMARY KEY ("ID"),
  CONSTRAINT "Rodent_baiting_IDL_fkey" FOREIGN KEY ("IDL")
    REFERENCES public."Incidence_relevance" ("ID_SERIAL") MATCH SIMPLE
    ON UPDATE CASCADE
    ON DELETE CASCADE,
  CONSTRAINT "Rodent_baiting_NO_PREMISES_BAITED_check" CHECK
("NO_PREMISES_BAITED" >= 0::double precision) NOT VALID,
  CONSTRAINT "Rodent_baiting_NO_PREMISES_WITH_GARBAGES_check" CHECK
("NO_PREMISES_WITH_GARBAGES" >= 0) NOT VALID,
  CONSTRAINT "Rodent_baiting_NO_PREMISES_RATS_check" CHECK
("NO_PREMISES_RATS" >= 0::double precision) NOT VALID
)
```

```
CREATE TABLE "Sanitation_code_complaints"
(
  "ID" integer NOT NULL ,
```

```
    "NATURE_CODE_VIOLATION" character varying(60) ,
    "IDL" integer NOT NULL ,
    CONSTRAINT "Sanitation_code_complaints_pkey" PRIMARY KEY ("ID"),
    CONSTRAINT "INCIDENCE_RELEEVANCE_RELATION" FOREIGN KEY ("IDL")
        REFERENCES public."Incidence_relevance" ("ID_SERIAL") MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE CASCADE)
```

```
CREATE TABLE "Tree_trims"
(
    "ID" integer NOT NULL ,
    "Location_Trees" character varying(25) ,
    "IDL" integer NOT NULL ,
    CONSTRAINT "Tree_trim_pkey" PRIMARY KEY ("ID"),
    CONSTRAINT "Tree_trim_IDL_fkey" FOREIGN KEY ("IDL")
        REFERENCES public."Incidence_relevance" ("ID_SERIAL") MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE CASCADE
)
```

```
CREATE TABLE "Trees_debris"
(
    "ID" integer NOT NULL ,
    "LOCATION_OF_TREES" character varying(25) ,
    "CURRENT_ACTIVITY" character varying(25) ,
    "MOST_RECENT_ACTION" character varying(90) ,
    "IDL" integer NOT NULL ,
    CONSTRAINT "Trees_debris_trim_pkey" PRIMARY KEY ("ID"),
    CONSTRAINT "Trees_debris_trim_IDL_fkey" FOREIGN KEY ("IDL")
        REFERENCES public."Incidence_relevance" ("ID_SERIAL") MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE CASCADE
```

```
)
```

```
CREATE TABLE users
(
   username character varying(50)  NOT NULL,
   email character varying(80) NOT NULL,
   password character varying NOT NULL,
   CONSTRAINT users_pkey PRIMARY KEY (username)
)
```

```
        CREATE TABLE user_log
        (
           "ID" integer NOT NULL ,
           "Action_Name" character varying(200) ,
           "Action_Time" timestamp with time zone,
           "FK_ID_USERNAME" character varying(50) ,
           CONSTRAINT user_log_pkey PRIMARY KEY ("ID"),
           CONSTRAINT "user_log_FK_ID_USERNAE_fkey" FOREIGN KEY ("FK_ID_USERNAME")
              REFERENCES public.users (username) MATCH SIMPLE
              ON UPDATE NO ACTION
              ON DELETE NO ACTION
        )
```

Σχόλια πάνω στην επιλογή των τύπων των δεδομένων και των ελέγχων τους:

- Το εύρος στα πεδία τύπου character varying καθορίστηκε γενικά από το μέγιστο πλήθος των χαρακτήρων στα αντίστοιχα πεδία στα csv αρχεία. Για την ακρίβεια καθορίστηκε κάπως μεγαλύτερο από αυτό το πλήθος.
- Τα περισσότερα πεδία όπως POLICE_DISTRICT,WARD,SSA,COMMUNITY_AREA που δε θα χρειαστεί να κάνουμε πράξεις καθορίστηκαν numeric με length 2 αυτό προέκυψε από τα δεδομένα τους στα csv .Από τα αρχεία αυτά προέκυψε ότι τα πεδία αυτά παίρνουν ακέραιες τιμές σε συγκεκριμένο διάστημα τιμών και για αυτό κάνουμε ελέγχους CHECK για αυτά τα πεδία.
- Τα x, y latitude, longitude καθορίστηκαν numeric καθώς γενικά δε θα χρησιμοποιηθούν σε πράξεις και το εύρος και η ακρίβεια τους καθορίστηκε από

τις τιμές στα αντίστοιχα πεδία στα αρχεία csv. Για την ακρίβεια ορίστηκαν λίγο μεγαλύτερα τα συγκεκριμένα χαρακτηριστικά των πεδίων

- ZIP_CODE:πενταψήφιος αριθμός που δε θα χρησιμοποιηθεί για πράξεις οπότε character(5)
- Τα πεδία CREATION_DATE,COMPLETION_DATE ορίζονται γιατί στα csv αρχεία πρακτικά δεν καταγράφεται πληροφορία για την ώρα(παντού η ώρα είναι 00:00 οπότε δε λαμβάνεται υπόψη)
- Κάποια πεδία στους πίνακες όπως NO_PREMISES_BAITED, NO_PREMISES_RATS, POT_HOLES_NO, NO_BLACK_CARTS, NO_DAYS_REPORTED_PARKED τον οποίο ο τύπος θα ήταν κανονικά INTEGER,παρατηρήθηκε στα csv αρχεία ότι περιείχαν από λάθος κάποιες δεκαδικές τιμές αλλά επειδή έπρεπε να συμπεριληφθούν αυτές τις εγγραφές στους πίνακες ορίστηκε να παίρνουν τα πεδία αυτά δεκαδικές τιμές.
- Στους πίνακες που έχουν σχέση με τον Incidence_relevance ορίστηκε όποτε διαγράφονται εγγραφές στον Incidence_relevance θα πρέπει να γίνονται διαγραφή και οι αντίστοιχες εγγραφές στους άλλους πίνακες(ON DELETE CASCADE)

Εισαγωγή Δεδομένων από csv αρχεία στη Βάση Δεδομένων

Το πέρασμα των δεδομένων από τα csv αρχεία στους κατάλληλους πίνακες της βάση δεδομένων έγινε μέσω ενός προγράμματος σε java. Ουσιαστικά χρησιμοποιήθηκε για την ανάγνωση των αρχείων μια πρόσθετη εξωτερική βιβλιοθήκη της java η opencsv:

http://zetcode.com/articles/opencsv/

https://sourceforge.net/projects/opencsv/

οποία διαθέτει ένα csvreader που μας εξασφαλίζει ότι τα δεδομένα από τα csv αρχεία θα διαβαστούν σωστά. Μετά την ανάγνωση των δεδομένων αυτά περνιούνται σε πίνακες. Στη συνέχεια με τη βοήθεια του JDBC συνδεόμαστε στον server και στην βάση δεδομένων και με κατάλληλα Prepared Statements εκτελέσθηκαν όλα οι εντολές INSERT INTO για να γίνουν όλα τα δεδομένα εισαγωγή στη βάση δεδομένων. Για λόγους απλότητας δεν παρατίθεται ο κώδικας που γράφτηκε για αυτή τη διαδικασία αλλά η παραπάνω συνοπτική περιγραφή του.

Αποθηκευμένες συναρτήσεις(stored functions)

1.

```
CREATE OR REPLACE FUNCTION public.find_total_request_number_per_type_by_date(
        date1 date,
        date2 date)
  RETURNS TABLE(type_of_service_request character varying, plithos bigint)
  LANGUAGE 'plpgsql'
AS $BODY$
```

```
DECLARE
   var_r record;
BEGIN
FOR var_r IN(
SELECT "TYPE_OF_SERVICE_REQUEST",COUNT(*) AS PLITHOS
        FROM public."Incidence_relevance"
        WHERE "CREATION_DATE">=date1 AND "CREATION_DATE"<=date2
        GROUP BY "TYPE_OF_SERVICE_REQUEST"
        ORDER BY COUNT(*) DESC)
        LOOP
        type_of_service_request:=(var_r."TYPE_OF_SERVICE_REQUEST");
   plithos:=var_r.PLITHOS;
        RETURN NEXT;
        END LOOP;
END ;

$BODY$;
```

Παράδειγμα εκτέλεσης

```
SELECT * FROM find_total_request_number_per_type_by_date(
        '2013-08-04',
        '2016-11-22'
)
```

2.

```
CREATE OR REPLACE FUNCTION
public.find_specific_type_count_request_by_date(
        date1 date,
        date2 date,
        type_name character varying)
   RETURNS TABLE(creation_date date, type_of_service_request character
varying, plithos bigint)
   LANGUAGE 'plpgsql'

AS $BODY$
```

```
DECLARE
  var_r record;
BEGIN
FOR var_r IN(
SELECT "CREATION_DATE","TYPE_OF_SERVICE_REQUEST",COUNT(*)AS
PLITHOS
        FROM public."Incidence_relevance"
        WHERE "CREATION_DATE">=date1 AND "CREATION_DATE"<=date2
AND "TYPE_OF_SERVICE_REQUEST"=type_name
        GROUP BY "CREATION_DATE","TYPE_OF_SERVICE_REQUEST"
        ORDER BY "CREATION_DATE"

)
        LOOP
        creation_date:=var_r."CREATION_DATE";
        type_of_service_request:=(var_r."TYPE_OF_SERVICE_REQUEST");
  plithos:=var_r.PLITHOS;
        RETURN NEXT;
        END LOOP;
END ;

$BODY$;
```

Παράδειγμα εκτέλεσης

```
SELECT * FROM public.find_specific_type_count_request_by_date(
        '2011-08-14',
        '2015-11-22',
        'Sanitation Code Violation')
```

3.

```
CREATE OR REPLACE FUNCTION
public.find_most_common_request_type_perzipcode(
        date1 date)
  RETURNS TABLE(zipcode character, type_of_service_request character
varying, plithos bigint)
  LANGUAGE 'plpgsql'


AS $BODY$

DECLARE
  var_r record;
BEGIN
FOR var_r IN
(SELECT  table2."ZIP_CODE" AS "ZIPCODE"
,"table1"."TYPE_OF_SERVICE_REQUEST" AS
"SERVICE_REQUEST_TYPE","table2"."PLITHOS" AS "COUNT"
```

```
FROM (SELECT  "TYPE_OF_SERVICE_REQUEST", "ZIP_CODE",COUNT(*)
        FROM public."Incidence_relevance"
        where "CREATION_DATE"=date1
   GROUP BY "ZIP_CODE","TYPE_OF_SERVICE_REQUEST") as
table1("TYPE_OF_SERVICE_REQUEST", "ZIP_CODE","PLITHOS"),
        (SELECT "ZIP_CODE", MAX("PLITHOS")
FROM (SELECT  "TYPE_OF_SERVICE_REQUEST", "ZIP_CODE",COUNT(*)
        FROM public."Incidence_relevance"
        where "CREATION_DATE"=date1
   GROUP BY "ZIP_CODE","TYPE_OF_SERVICE_REQUEST") as
table1("TYPE_OF_SERVICE_REQUEST", "ZIP_CODE","PLITHOS")
        GROUP BY "ZIP_CODE") AS table2("ZIP_CODE","PLITHOS")
        WHERE "table1"."ZIP_CODE"="table2"."ZIP_CODE" AND
"table1"."PLITHOS"="table2"."PLITHOS"
ORDER BY "table2"."ZIP_CODE")
        LOOP
        zipcode:=var_r."ZIPCODE";
        type_of_service_request:=var_r."SERVICE_REQUEST_TYPE";
   plithos:=var_r."COUNT";
        RETURN NEXT;
        END LOOP;
END ;

$BODY$;
```

Παράδειγμα εκτέλεσης

```
SELECT * FROM public.find_most_common_request_type_perzipcode(
        '2014-05-23'
)
```

4.

```
CREATE OR REPLACE FUNCTION
public.find_avg_completion_time_per_type_by_date(
        date1 date,
        date2 date)
   RETURNS TABLE(type_of_service_request character varying, avg_time
interval)
   LANGUAGE 'plpgsql'


AS $BODY$

DECLARE
   var_r record;
BEGIN
FOR var_r IN(
```

```
SELECT
"TYPE_OF_SERVICE_REQUEST",avg(age("COMPLETION_DATE","CREATION_DAT
E")) AS AVG_TIME
        FROM public."Incidence_relevance"
        WHERE "COMPLETION_DATE" IS NOT  NULL AND
"CREATION_DATE">=date1 AND "CREATION_DATE"<=date2
  GROUP BY "TYPE_OF_SERVICE_REQUEST"
        ORDER BY AVG_TIME
                                                                                )
        LOOP
        type_of_service_request:=var_r."TYPE_OF_SERVICE_REQUEST";
  avg_time:=var_r.AVG_TIME;
        RETURN NEXT;
        END LOOP;
END ;

$BODY$;
```

Παράδειγμα εκτέλεσης

```
SELECT * FROM public.find_avg_completion_time_per_type_by_date(
        '2011-09-17',
        '2012-08-27'
)
```

5.

```
CREATE OR REPLACE FUNCTION
public.find_most_common_request_type_in_bounding_box(
        date1 date,
        xl double precision,
        xr double precision,
        yb double precision,
        yu double precision)
  RETURNS TABLE(type_of_service_request character varying)
  LANGUAGE 'plpgsql'

AS $BODY$

DECLARE
  var_r record;
BEGIN
FOR var_r IN(
SELECT table1."TYPE_OF_SERVICE_REQUEST" AS
"TYPE_OF_SERVICE_REQUEST"  FROM
(SELECT  "TYPE_OF_SERVICE_REQUEST",COUNT(*)
        FROM public."Incidence_relevance"
        WHERE "CREATION_DATE"=date1 AND
 ("LONGITUDE">=xl AND "LONGITUDE"<=xr) AND
 ("LATITUDE">=yb AND "LATITUDE"<=yu)
        GROUP BY "TYPE_OF_SERVICE_REQUEST"
```

```
        ORDER BY COUNT(*) DESC
        LIMIT 1) AS table1("TYPE_OF_SERVICE_REQUEST","PLITHOS")        )

        LOOP
        type_of_service_request:=var_r."TYPE_OF_SERVICE_REQUEST";

        RETURN NEXT;
        END LOOP;
END ;

$BODY$;
```

Παράδειγμα εκτέλεσης

```
SELECT * FROM public.find_most_common_request_type_in_bounding_box(
        '2015-03-14', -88.00, -87.00, 41.00, 42.00)
```

6.

```
CREATE OR REPLACE FUNCTION public.find_topssa_by_date(
        date1 date,
        date2 date)
  RETURNS TABLE(ssa character varying, plithos bigint)
  LANGUAGE 'plpgsql'


AS $BODY$

DECLARE
  var_r record;
BEGIN
FOR var_r IN(
SELECT "SSA",COUNT(*) AS PLITHOS FROM
((SELECT "SSA","IDL" FROM
"Abandonded_vehicle") UNION ALL (SELECT "SSA","IDL" FROM
"Garbage_carts")  UNION ALL (SELECT "SSA","IDL" FROM
"Graffity_Removal") UNION ALL (SELECT "SSA","IDL" FROM
"Pot_Holes")) AS A1("SSA","IDL"),"Incidence_relevance"
WHERE "SSA" IS NOT NULL  AND "Incidence_relevance"."ID_SERIAL"=A1."IDL"
AND

"Incidence_relevance"."CREATION_DATE">=date1 AND

"Incidence_relevance"."CREATION_DATE"<=date2
GROUP BY "SSA"
ORDER BY PLITHOS DESC
LIMIT 5)
        LOOP
        ssa:=var_r."SSA";
  plithos:=var_r.PLITHOS;
```

```
        RETURN NEXT;
        END LOOP;
END ;

$BODY$;
```

Παράδειγμα εκτέλεσης

```
SELECT * FROM public.find_topssa_by_date(
        '2013-10-02',
        '2015-05-17'
)
```

7.

```
CREATE OR REPLACE FUNCTION public.find_duplicate_licence_plate(
        )
  RETURNS TABLE(licence_plate character varying, plithos bigint)
  LANGUAGE 'plpgsql'


AS $BODY$

DECLARE
  var_r record;
BEGIN
FOR var_r IN(
SELECT "LICENCE_PLATE",COUNT("LICENCE_PLATE") AS PLITHOS
FROM "Abandonded_vehicle"
GROUP BY "LICENCE_PLATE"
HAVING(COUNT("LICENCE_PLATE")>1)
ORDER BY PLITHOS DESC)

        LOOP
        licence_plate:=var_r."LICENCE_PLATE";
  plithos:=var_r.PLITHOS;
        RETURN NEXT;
        END LOOP;
END ;

$BODY$;
```

Παράδειγμα εκτέλεσης

```
SELECT * FROM public.find_duplicate_licence_plate()
```

8.

```
CREATE OR REPLACE FUNCTION
public.find_second_most_common_color_vehicle(
        )
  RETURNS TABLE(vehicle_color character varying, plithos bigint)
  LANGUAGE 'plpgsql'

  COST 100
  VOLATILE
  ROWS 1000
AS $BODY$

DECLARE
  var_r record;
BEGIN
FOR var_r IN(
SELECT  "VEHICLE_COLOR",COUNT(*) AS PLITHOS
        FROM public."Abandonded_vehicle"
        GROUP BY "VEHICLE_COLOR"
        ORDER BY COUNT(*) DESC
        LIMIT 1 OFFSET 1)

        LOOP
        vehicle_color:=var_r."VEHICLE_COLOR";
  plithos:=var_r.PLITHOS;
        RETURN NEXT;
        END LOOP;
END ;

$BODY$;
```

Παράδειγμα εκτέλεσης

```
SELECT * FROM public.find_second_most_common_color_vehicle()
```

9.

```
CREATE OR REPLACE FUNCTION public.find_less_premises_baited(
        k1 bigint)
  RETURNS TABLE(id_serial integer, idl integer, no_premises_baited double
precision, no_premises_with_garbages integer, no_premises_rats double
precision, current_activity character varying, most_recent_action character
varying, creation_date date, status character varying, completion_date date,
service_request_number character varying, type_of_service_request character
varying, street_address character varying, zip_code character, x_coordinate
numeric, y_coordinate numeric, ward numeric, police_district numeric,
community_area numeric, latitude numeric, longitude numeric, locationl
character varying)
  LANGUAGE 'plpgsql'


AS $BODY$
```

```
BEGIN
 RETURN QUERY SELECT "ID_SERIAL","ID", "NO_PREMISES_BAITED",
"NO_PREMISES_WITH_GARBAGES", "NO_PREMISES_RATS",
"CURRENT_ACTIVITY", "MOST_RECENT_ACTION", "CREATION_DATE",
"STATUS", "COMPLETION_DATE", "SERVICE_REQUEST_NUMBER",
"TYPE_OF_SERVICE_REQUEST", "STREET_ADDRESS", "ZIP_CODE",
"X_COORDINATE", "Y_COORDINATE", "WARD", "POLICE_DISTRICT",
"COMMUNITY_AREA", "LATITUDE", "LONGITUDE", "LOCATION"
        FROM "Rodent_baiting"  JOIN "Incidence_relevance" ON
"Rodent_baiting"."IDL"="Incidence_relevance"."ID_SERIAL"
        WHERE "NO_PREMISES_BAITED"<k1;
END;
$BODY$;
```

Παράδειγμα εκτέλεσης

```
SELECT * FROM public.find_less_premises_baited(4)
```

10.

```
CREATE OR REPLACE FUNCTION public.find_less_premises_garbage(
        k1 bigint)
   RETURNS TABLE(id_serial integer, idl integer, no_premises_baited double
precision, no_premises_with_garbages integer, no_premises_rats double
precision, current_activity character varying, most_recent_action character
varying, creation_date date, status character varying, completion_date date,
service_request_number character varying, type_of_service_request character
varying, street_address character varying, zip_code character, x_coordinate
numeric, y_coordinate numeric, ward numeric, police_district numeric,
community_area numeric, latitude numeric, longitude numeric, locationl
character varying)
   LANGUAGE 'plpgsql'

AS $BODY$

BEGIN
 RETURN QUERY SELECT "ID_SERIAL","ID", "NO_PREMISES_BAITED",
"NO_PREMISES_WITH_GARBAGES", "NO_PREMISES_RATS",
"CURRENT_ACTIVITY", "MOST_RECENT_ACTION", "CREATION_DATE",
"STATUS", "COMPLETION_DATE", "SERVICE_REQUEST_NUMBER",
"TYPE_OF_SERVICE_REQUEST", "STREET_ADDRESS", "ZIP_CODE",
"X_COORDINATE", "Y_COORDINATE", "WARD", "POLICE_DISTRICT",
"COMMUNITY_AREA", "LATITUDE", "LONGITUDE", "LOCATION"
        FROM "Rodent_baiting"  JOIN "Incidence_relevance" ON
"Rodent_baiting"."IDL"="Incidence_relevance"."ID_SERIAL"
        WHERE "NO_PREMISES_WITH_GARBAGES"<k1;
END;
$BODY$;
```

Παράδειγμα εκτέλεσης

```
SELECT * FROM public.find_less_premises_garbage(3)
```

11.

```
CREATE OR REPLACE FUNCTION public.find_less_premises_rats(
        k1 bigint)
   RETURNS TABLE(id_serial integer, idl integer, no_premises_baited double
precision, no_premises_with_garbages integer, no_premises_rats double
precision, current_activity character varying, most_recent_action character
varying, creation_date date, status character varying, completion_date date,
service_request_number character varying, type_of_service_request character
varying, street_address character varying, zip_code character, x_coordinate
numeric, y_coordinate numeric, ward numeric, police_district numeric,
community_area numeric, latitude numeric, longitude numeric, locationl
character varying)
   LANGUAGE 'plpgsql'

AS $BODY$

BEGIN
 RETURN QUERY SELECT "ID_SERIAL","ID", "NO_PREMISES_BAITED",
"NO_PREMISES_WITH_GARBAGES", "NO_PREMISES_RATS",
"CURRENT_ACTIVITY", "MOST_RECENT_ACTION", "CREATION_DATE",
"STATUS", "COMPLETION_DATE", "SERVICE_REQUEST_NUMBER",
"TYPE_OF_SERVICE_REQUEST", "STREET_ADDRESS", "ZIP_CODE",
"X_COORDINATE", "Y_COORDINATE", "WARD", "POLICE_DISTRICT",
"COMMUNITY_AREA", "LATITUDE", "LONGITUDE", "LOCATION"
        FROM "Rodent_baiting"  JOIN "Incidence_relevance" ON
"Rodent_baiting"."IDL"="Incidence_relevance"."ID_SERIAL"
        WHERE "NO_PREMISES_RATS"<k1;
END;
$BODY$;
```

Παράδειγμα εκτέλεσης

```
SELECT * FROM public.find_less_premises_rats(2)
```

12.

```
CREATE OR REPLACE FUNCTION public.find_pothole_with_rodentbaiting_by_date(
        date1 date)
   RETURNS TABLE(police_district numeric)
   LANGUAGE 'plpgsql'
```

```
AS $BODY$


DECLARE
   var_r record;
BEGIN
FOR var_r IN(
SELECT DISTINCT  "POLICE_DISTRICT"
FROM "Incidence_relevance" JOIN "Pot_Holes" ON
"Incidence_relevance"."ID_SERIAL"="Pot_Holes"."IDL"
WHERE "Pot_Holes"."POT_HOLES_NO">1  AND "CREATION_DATE"=date1
        AND "POLICE_DISTRICT" IN
(SELECT "POLICE_DISTRICT" FROM "Incidence_relevance" JOIN "Rodent_baiting" ON
"Incidence_relevance"."ID_SERIAL"="Rodent_baiting"."IDL"
 WHERE "Rodent_baiting"."NO_PREMISES_BAITED">1 AND "CREATION_DATE"=date1))
        LOOP
        police_district:=var_r. "POLICE_DISTRICT";
        RETURN NEXT;
        END LOOP;
END ;


$BODY$;
```

Παράδειγμα εκτέλεσης

```
SELECT * FROM public.find_pothole_with_rodentbaiting_by_date(
        '2014-07-28'
)
```

Δημιουργία και Χρήση Ευρετηρίων

Δημιουργήσαμε τα παρακάτω ευρετήρια(index) στα παρακάτω πεδία των πινάκων της βάσης δεδομένων με στόχο την αποδοτικότερη εκτέλεση των ερωτημάτων 1-12

αλλά και των 2 επιπλέον ερωτημάτων της εφαρμογής(αναζήτηση με βάση ταχυδρομικό κώδικα και διεύθυνση):

- CREATE INDEX "LICENCE_IDX" ON "Abandonded_vehicle" ("LICENCE_PLATE")
- CREATE INDEX abandonded_vehicle_ssa_idx ON "Abandonded_vehicle"("SSA")
- CREATE INDEX garbage_carts_ssa_idx ON "Garbage_carts"("SSA")
- CREATE INDEX graffity_removal_ssa_idx ON "Graffity_Removal"("SSA")
- CREATE INDEX pot_holes_ssa_idx ON "Pot_Holes" ("SSA")
- CREATE INDEX pot_holes_no_idx ON "Pot_Holes" ("POT_HOLES_NO")
- CREATE INDEX "Rodent_Baiting_idx1" ON "Rodent_baiting" ("NO_PREMISES_BAITED")
- CREATE INDEX "INCIDENCE_RELEVANCE_IDX1"ON "Incidence_relevance" ("CREATION_DATE")
- CREATE INDEX "INCIDENCE_RELEVANCE_IDX2" ON "Incidence_relevance" ("COMPLETION_DATE")
- CREATE INDEX idx_id ON "Incidence_relevance" ("ID_SERIAL")
- CREATE INDEX incindence_relevance_street_idx ON "Incidence_relevance" ("STREET_ADDRESS")
- CREATE INDEX incindence_relevance_tos_idx ON "Incidence_relevance" ("TYPE_OF_SERVICE_REQUEST")
- CREATE INDEX incindence_relevance_zip_idx ON "Incidence_relevance" ("ZIP_CODE")

Εφαρμογή

Ρυθμίσεις-Χρησιμοποιούμενες τεχνολογίες

Για να στήσουμε τη web εφαρμογή μας στο localhost,χρησιμοποιήθηκε ο apache web server του xampp. Ρυθμίστηκε για ασφάλεια κάθε ιστοσελίδα της εφαρμογής να τρέχει με πρωτόκολλο https. Για αυτό έγιναν τα παρακάτω:

- Δημιουργία SSL certificate(περιέχει το public key) για password encryption. Δεν χρησιμοποιείται το default key του xampp γιατί είναι ίδιο για όλους τους χρήστες που το κατεβάζουν. Στις παρακάτω εικόνες φαίνονται τα βήματα:

1.

```
C:\\xampp\\apache>newcert
Loading 'screen' into random state - done
Generating a 1024 bit RSA private key
.............................++++++
.....................................++++++
writing new private key to 'privkey.pem'
Enter PEM pass phrase:                          //αφου μπεί το pass phrase

Verifying - Enter PEM pass phrase:              //επιβεβαίωση pass phrase


-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:  //συμπλήρωση κωδικού χώρας

State or Province Name (full name) [Some-State]:NY                          //Συμπλήρωση κάποιων άλλων πληροφοριών-εδώ
Locality Name (eg, city) []:New York                                        //κάποιες είναι συμπληρωμένες
Organization Name (eg, company) [Internet Widgits Pty Ltd]:Rob's Great Company
Organizational Unit Name (eg, section) []:
Common Name (eg, YOUR name) []:

Email Address []:

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
Enter pass phrase for privkey.pem:

writing RSA key
Loading 'screen' into random state - done
Signature ok
subject=/C=xx/ST=xx/L=xxxx/O=xxx/CN=commonname
Getting Private key

-----
Das Zertifikat wurde erstellt.
The certificate was provided.

Press any key to continue . . .

C:\\xampp\\apache>
```

2. Εισαγωγή certificate στο browser. Η διαδικασία είναι παρόμοια για κάθε browser. Στο Mozilla Options->Privacy & Security->ViewCertificates και κάνουμε import το αρχείο: C:\xampp\apache\conf\ssl.crt\server.crt .Στο τέλος επιλέγουμε την επιλογή Trust this CA to identify web sites.
3. Ρύθμιση Apache Server ώστε οι φάκελοι που/αρχεία που θα χρησιμοποιούνται από αυτόν θα γίνονται πάντα encryption. Αυτό γίνεται, προσθέτωντας στο αρχείο με διαδρομή την παρακάτω C:\xampp\apache\conf\extra\httpd-xampp.conf τη φράση SSLRequireSSL στο κομμάτι ανάμεσα στα tag Directory των παρακλατω path:
   - C:\xampp\phpmyadmin
   - C:\xampp\htdocs\xampp
   - C:\xampp\webalizer
4. Τέλος ακολουθεί ένα προαιρετικό βήμα το οποίο ουσιαστικά κάνει redirect τα http σε https {πιο φιλικό στο χρήστη}, προσθέτοντας κάτω από την εντολή :

```
LoadModule rewrite_module modules/mod_rewrite.so
```

στο αρχείο: C:\xampp\apache\conf\extra\httpd-xampp.conf το παρακάτω :

```
<IfModule mod_rewrite.c>
    RewriteEngine On

    # Redirect /xampp folder to https
    RewriteCond %{HTTPS} !=on
    RewriteCond %{REQUEST_URI} xampp
    RewriteRule ^(.*) https://%{SERVER_NAME}$1 [R,L]

    # Redirect /phpMyAdmin folder to https
    RewriteCond %{HTTPS} !=on
    RewriteCond %{REQUEST_URI} phpmyadmin
    RewriteRule ^(.*) https://%{SERVER_NAME}$1 [R,L]

    # Redirect /security folder to https
    RewriteCond %{HTTPS} !=on
    RewriteCond %{REQUEST_URI} security
    RewriteRule ^(.*) https://%{SERVER_NAME}$1 [R,L]

    # Redirect /webalizer folder to https
    RewriteCond %{HTTPS} !=on
    RewriteCond %{REQUEST_URI} webalizer
    RewriteRule ^(.*) https://%{SERVER_NAME}$1 [R,L]
</IfModule>
```

Για το front-end κομμάτι της εφαρμογής χρησιμοποιήθηκαν οι παρακάτω βασικές τεχνολογίες:

- HTML5
- Javascript
- CSS

Για το back-end κομμάτι της εφαρμογής χρησιμοποιήθηκε PHP. Για το authentication χρησιμοποιήθηκε η βιβλιοθήκη PHP-JWT με την οποία ορίζονται αντικείμενα τύπου JSON Web Token τα οποία παρέχουν ένα ασφαλή τρόπο μετάδοσης πληροφορίων ως μέρη ενός JSON object.Ουσιαστικά το αντικείμενο αυτό παίρνει τις πληροφορίες σε μορφή αναπαράστασης JSON της κωδικοποιεί με κάποιο secret key το οποίο το ξέρει μόνο ο server. Αν οι πληροφορίες που δίνει ο χρήστης είναι σωστές φτιάχνεται ένα κωδικοποιημένο jwt το οποίο ουσιαστικά είναι αυτό που κάνει login το χρήστη στην εφαρμογή. Για να είναι εφικτή η επικοινωνία της php με το server της postgresql όπου είναι αποθηκευμένη η βάση μας, ξεσχολιάζουμε τις παρακάτω γραμμές στο αρχείο php.ini:

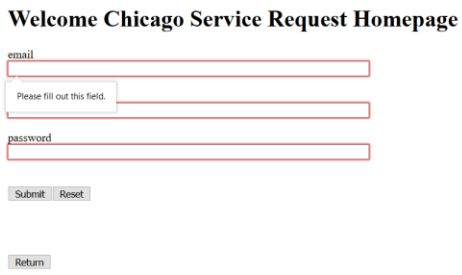
```
extension=php_pdo_pgsql.dll
extension=php_pgsql.dll
```

Δυνατότητες

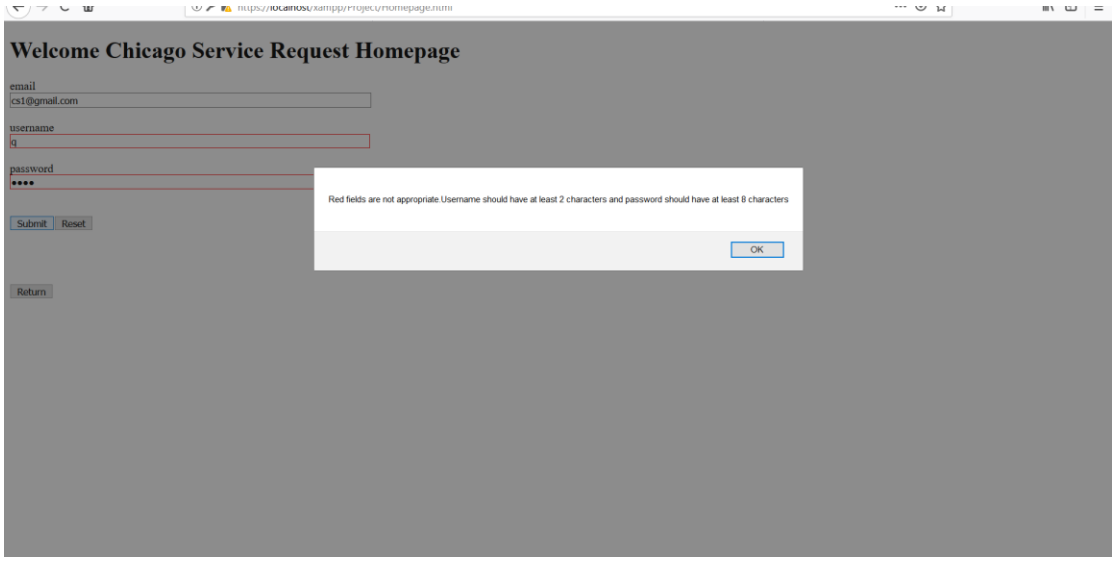
Εμφανίζεται μια σελίδα με δύο κουμπιά το ένα σε πάει στη register φόρμα όπου ο χρήστης δίνει username,email και ένα password για να εγγραφή αν ολοκληρωθεί επιτυχώς η εγγραφή εμφανίζεται ένα μήνυμα που ενημερώνει το χρήστη για αυτό και ένα link επιστροφής την αρχική σελίδα. Το άλλο κουμπί στην αρχική σελίδα εμφανίζει μια login φόρμα όπου αν ο χρήστης δώσει σωστά στοιχεία εμφανίζεται ένα menu δύο επιλογών. Μια επιλογή δίνει στο χρήστη δυνατότητα εισαγωγής κάποιο συμβάντος που συνέβη στην πόλη του Chicago μέσω μιας φόρμας . Ο χρήστης εισάγει τα στοιχεία στη φόρμα και πατάει submit και εμφανίζεται ένα μήνυμα ότι το συμβάν καταγράφηκε και δίνει τη δυνατότητα στο χρήστη είτε να επιστρέψει στην σελίδα με το μενού επιλογών του χρήστη είτε να κάνει logout. Η άλλη επιλογή στη σελίδα του χρήστη εμφανίζει ένα υπομενού στο οποίο ο χρήστης μπορεί να κάνει αναζήτηση συμβάντων με βάση ZIPCODE ή STREET_ADDRESS(εμφάνιση σχετικών φορμών) ή να πατήσει στην τρίτη επιλογή για αναζήτηση που του δίνει τη δυνατότητα να

επιλέξει μια από τις λειτουργίες που υλοποιούν συναρτήσεις 1-3 που είδαμε παραπάνω. συμπληρώνοντας τη σχετική φόρμα να εκτελέσει μια από αυτές με παραμέτρους τα στοιχεία που συμπλήρωσε στην αντίστοιχη φόρμα. Όλες αυτές οι αναζητήσεις οδηγούν τον χρήστη μετά την υποβολή της αντίστοιχη φόρμα σε μια σελίδα με τα αποτελέσματα της αναζήτησης και τη δυνατότητα στη συνέχεια επιστροφή στη βασική σελίδα χρήστη ή αποσύνδεσης. Όλες οι ενέργειες του χρήστη καταγράφονται από την εφαρμογή σε ένα πίνακα της βάσης δεδομένων μας. Παρακάτω παρατίθενται κάποιες εικόνες που αποδεικνύουν την καλή λειτουργία της εφαρμογής:

ΕΙΚΟΝΑ 1

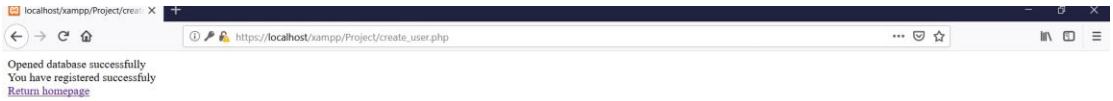


ΕΙΚΟΝΑ 2

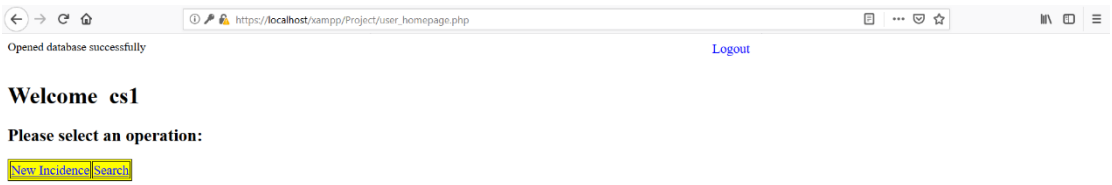


EIKONA 3

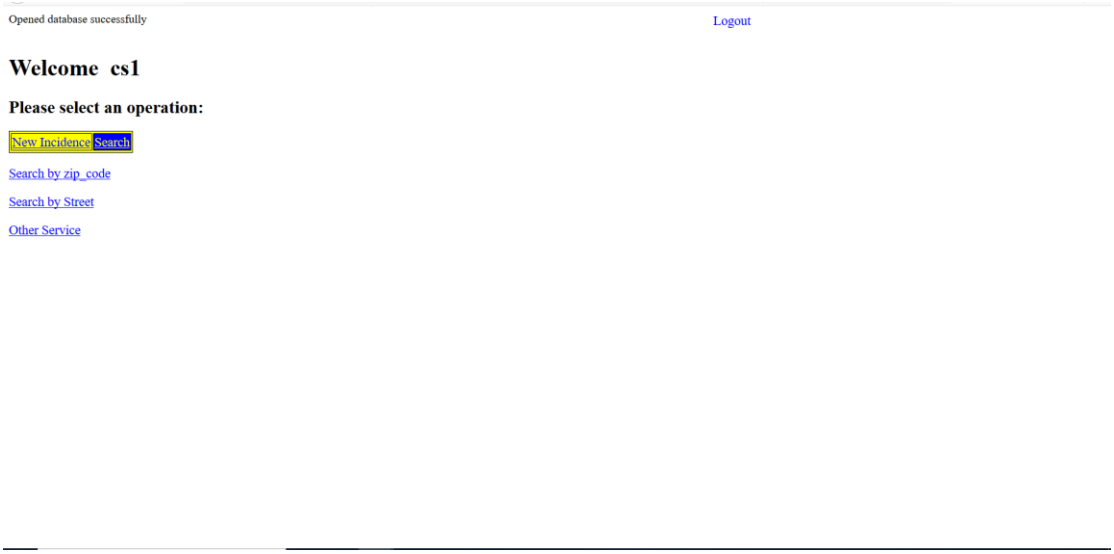
EIKONA 4



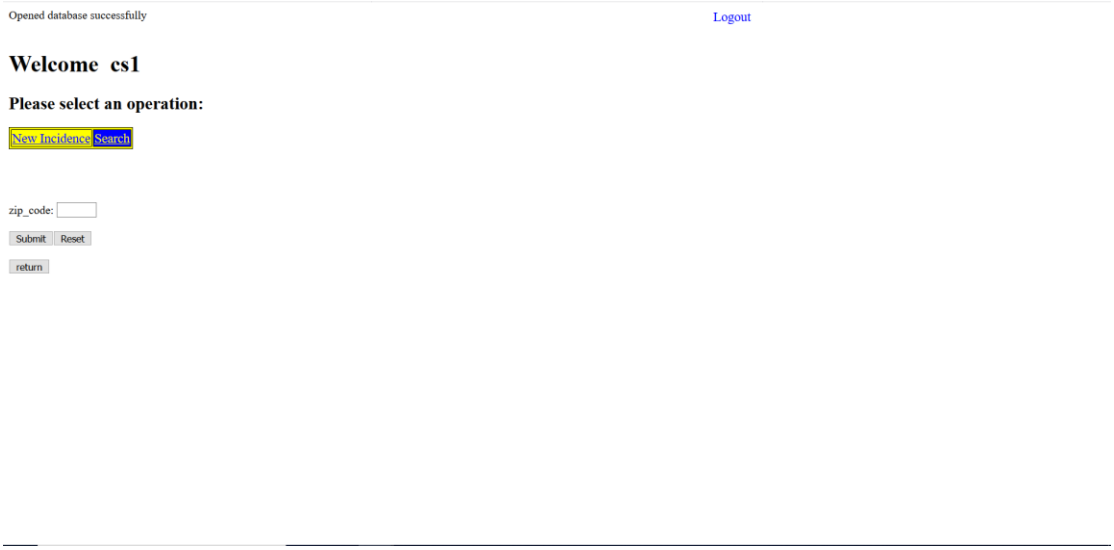
EIKONA 5



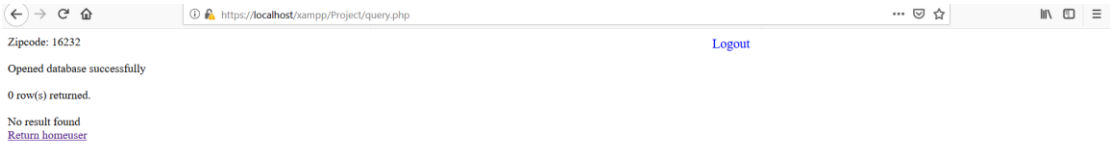
EIKONA 6



EIKONA 7

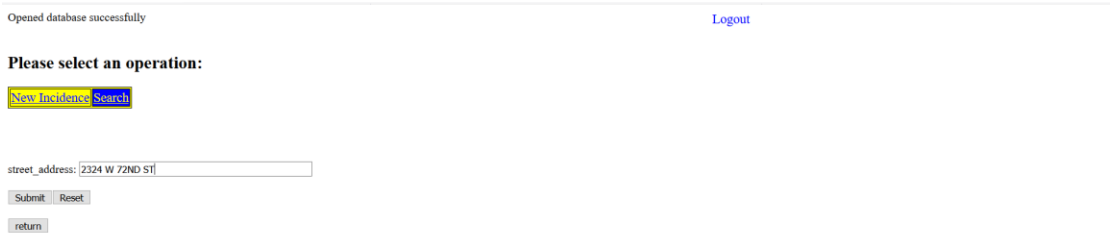


EIKONA 8

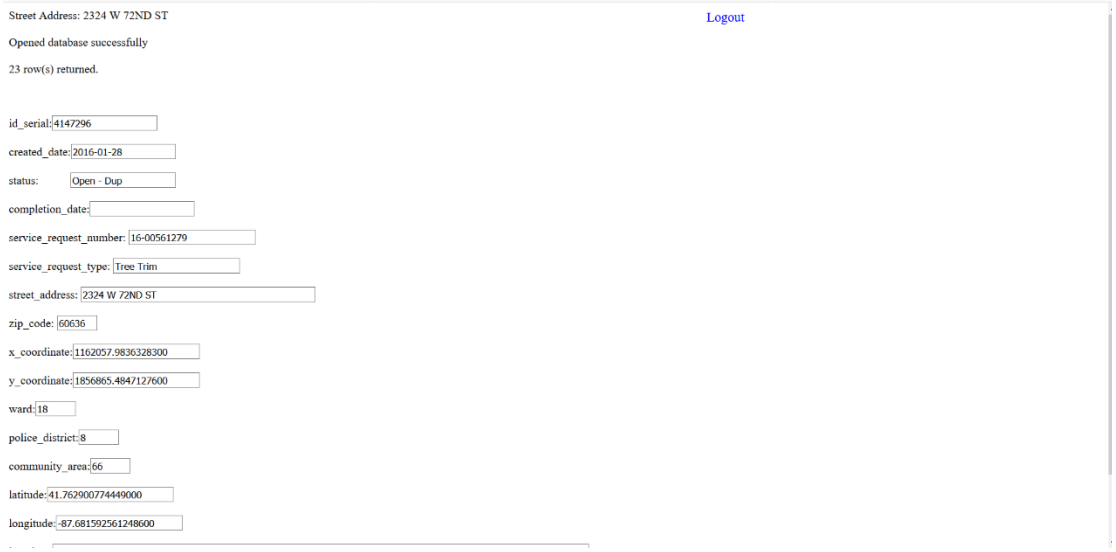


EIKONA 9

EIKONA 10



EIKONA 11



EIKONA 12

id_serial: 4147296

created_date: 2016-01-28

status: Open - Dup

completion_date:

service_request_number: 16-00561279

service_request_type: Tree Trim

street_address: 2324 W 72ND ST

zip_code: 60636

x_coordinate: 1162057.9836328300

y_coordinate: 1856865.4847127600

ward: 18

police_district: 8

community_area: 66

latitude: 41.762900774449000

longitude: -87.681592561248600

location: {'latitude': '41.762900774449015', 'longitude': '-87.681592561248651', 'needs_recoding': False}

previous <- 23 next ->

Return homeuser

EIKONA 13

Opened database successfully Logout

Please select an operation:

New Incidence Search

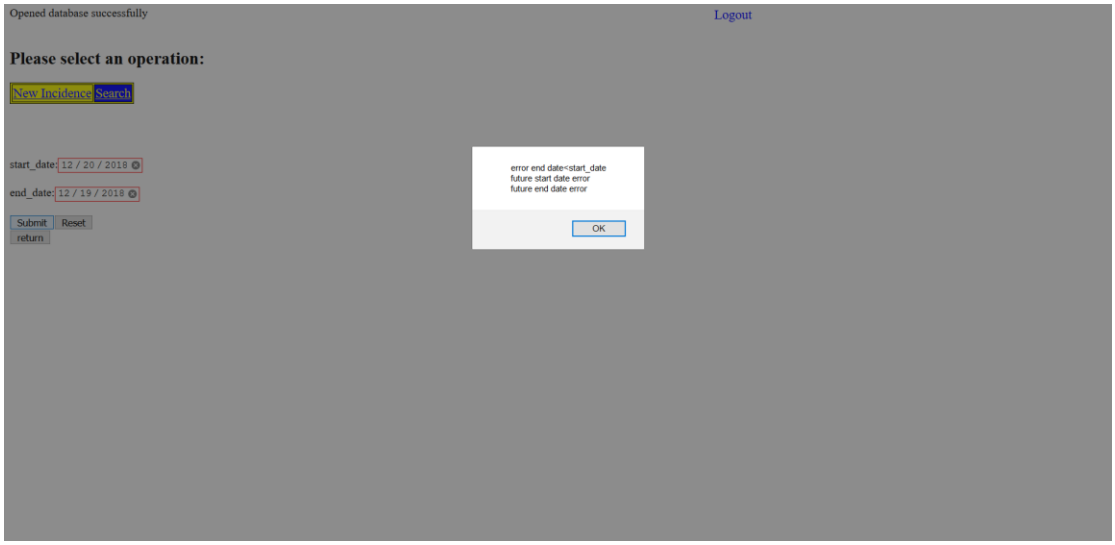
Find total request per type by date

Find total requests per day for a specific request type

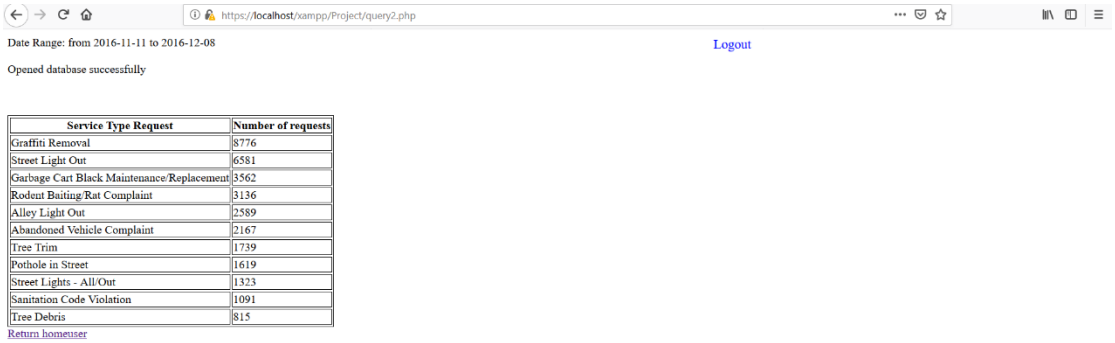
Find the most common service request per zipcode for a day

return

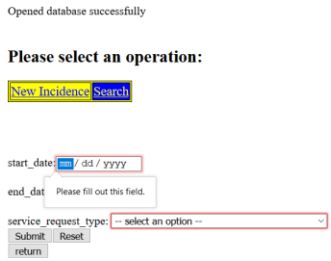
EIKONA 14



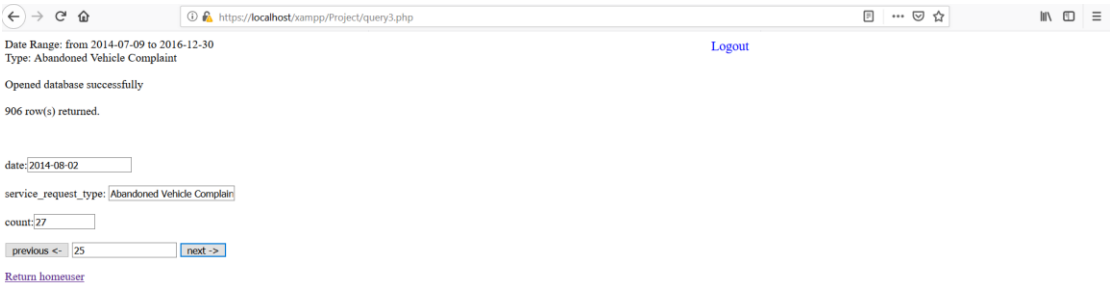
EIKONA 15



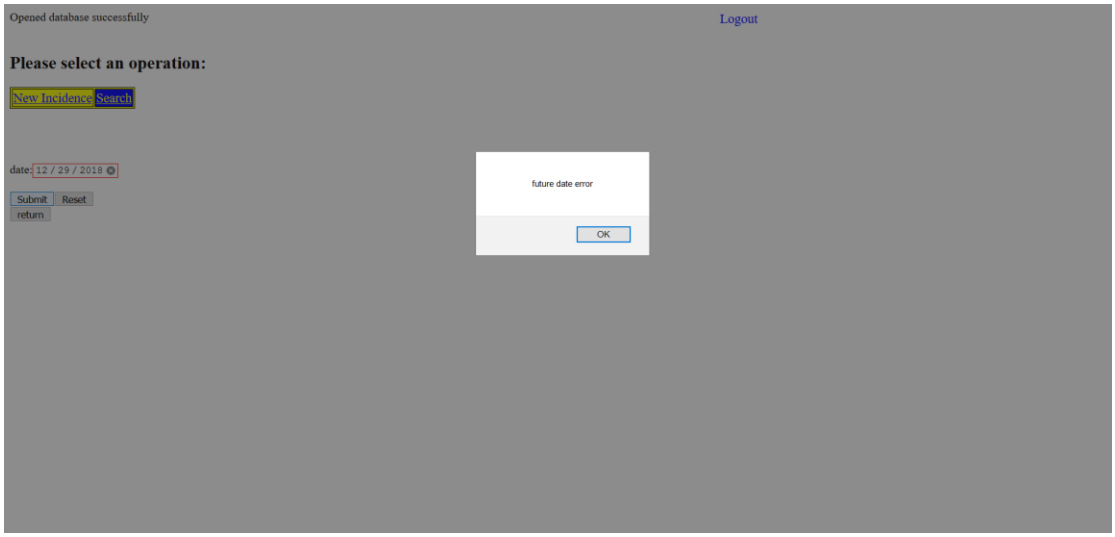
EIKONA 16



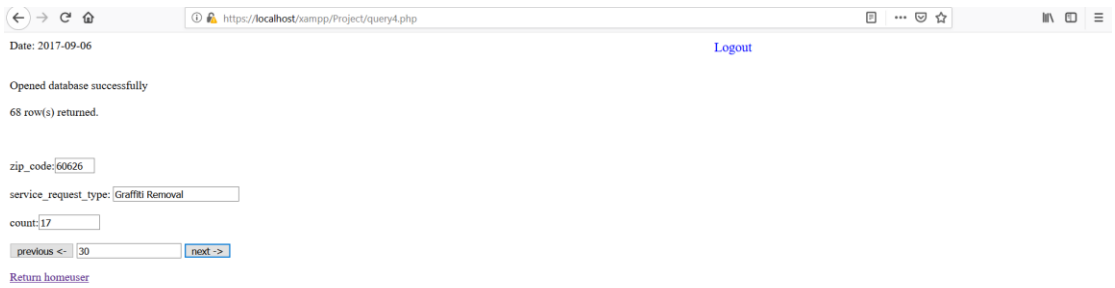
EIKONA 17



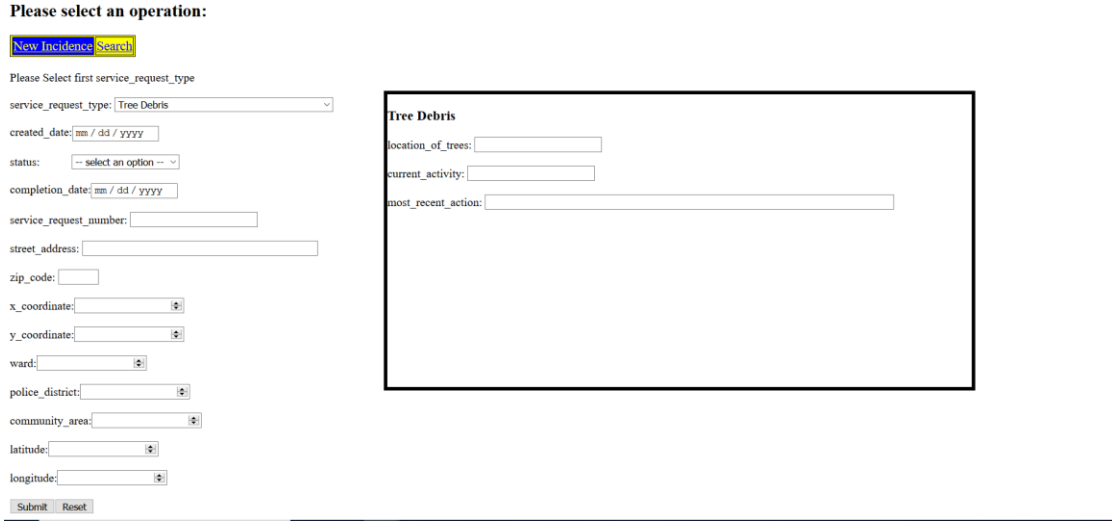
EIKONA 18



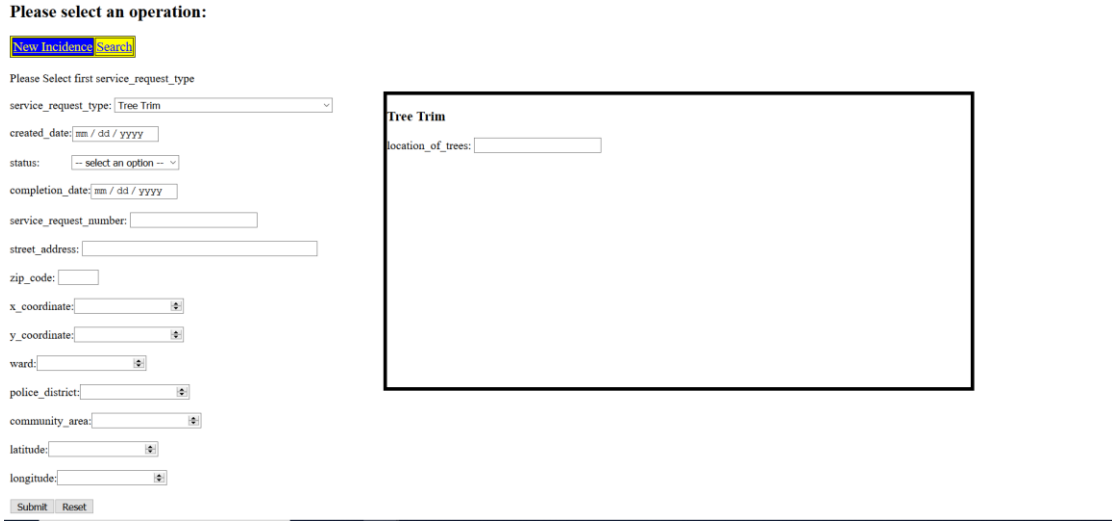
EIKONA 19



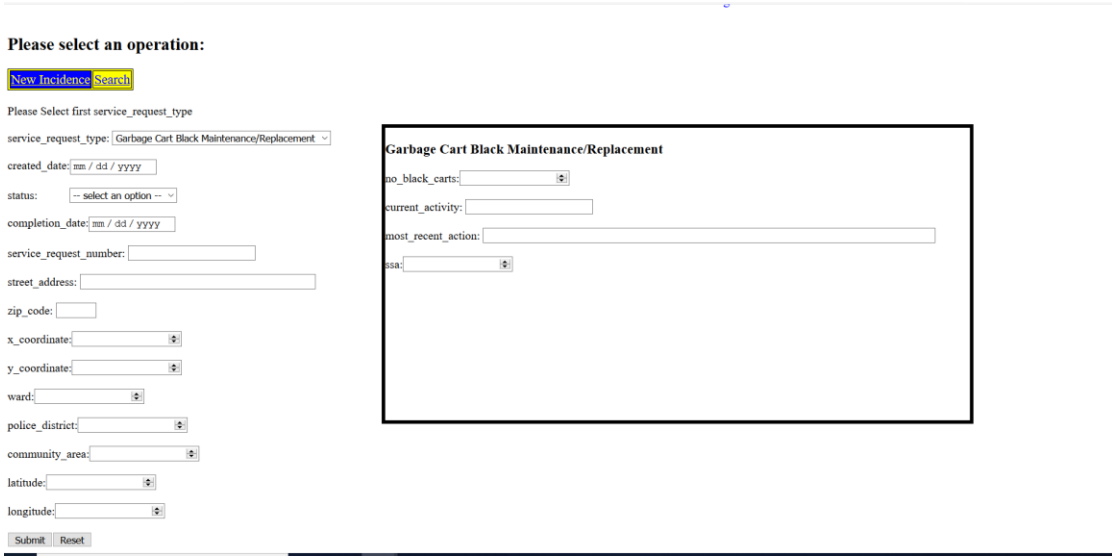
EIKONA 20



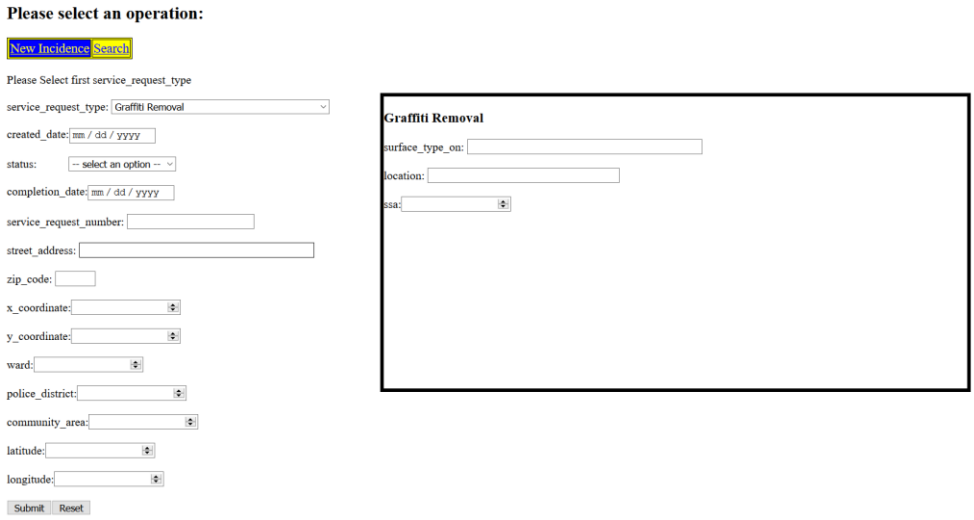
EIKONA 21



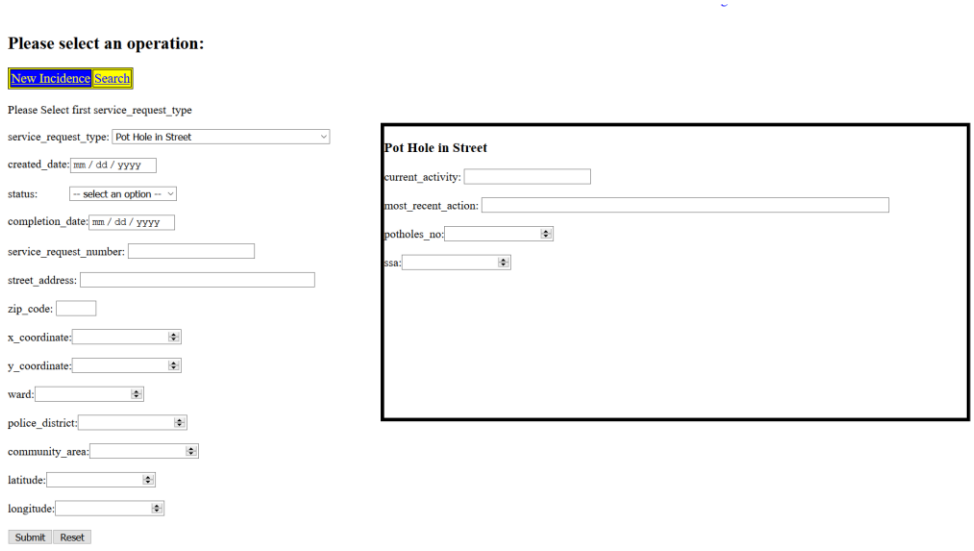
EIKONA 22



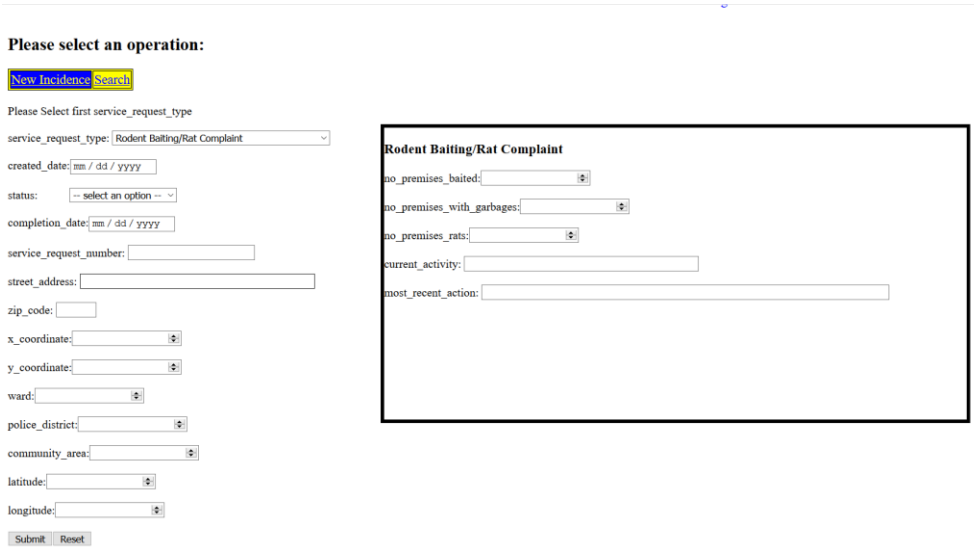
EIKONA 23



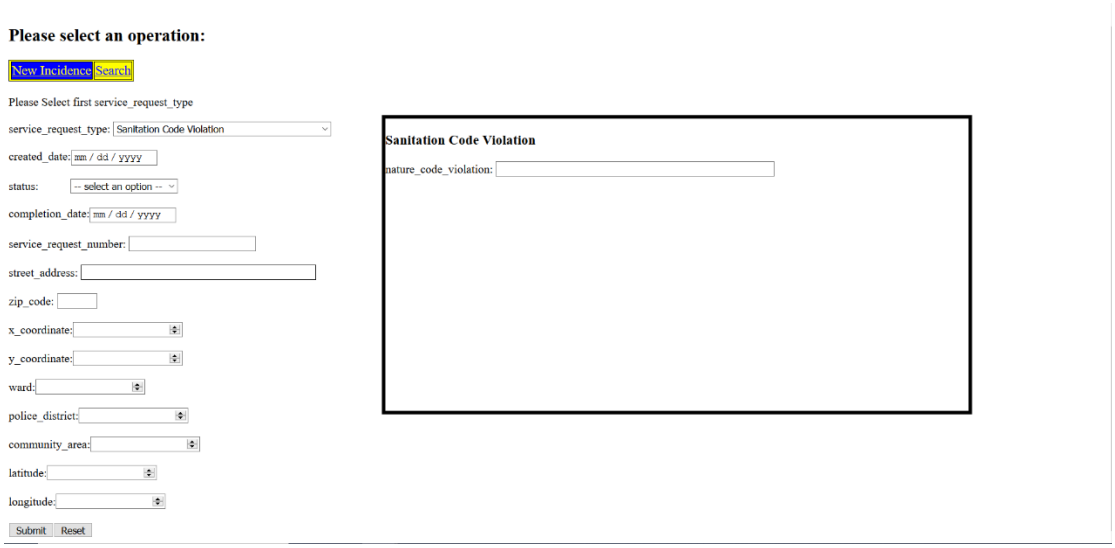
EIKONA 24



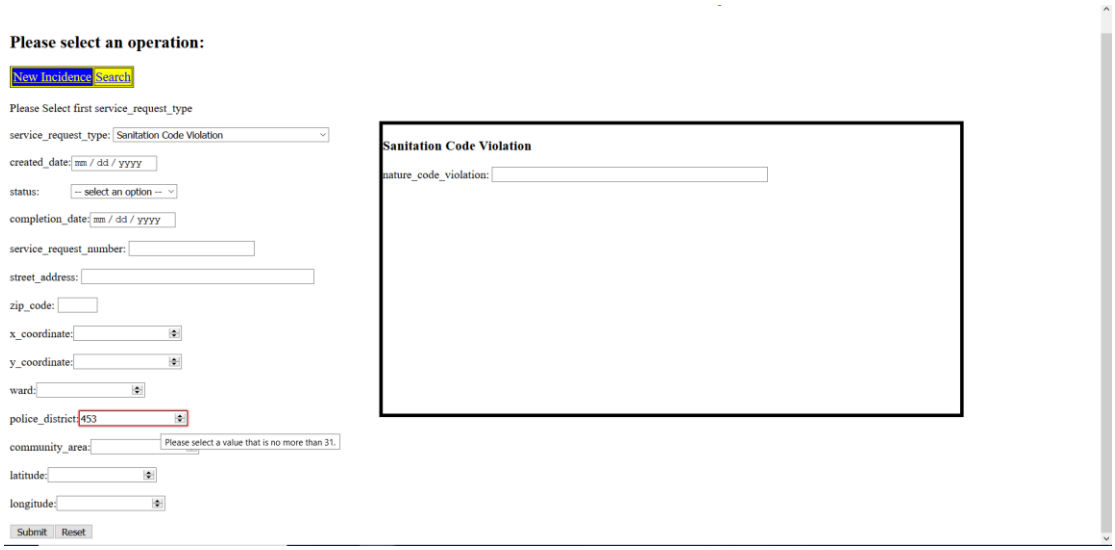
EIKONA 25



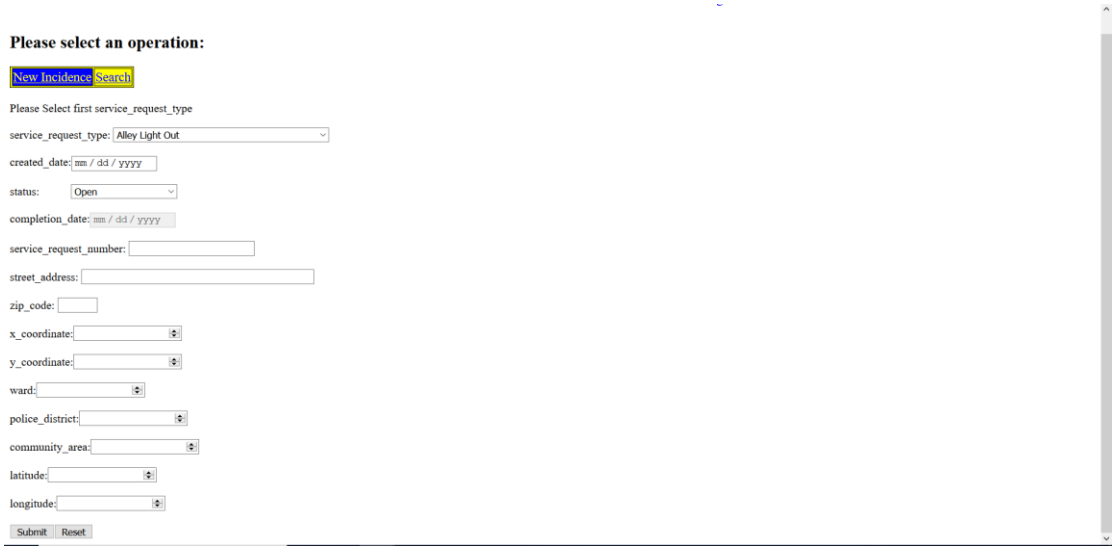
EIKONA 26



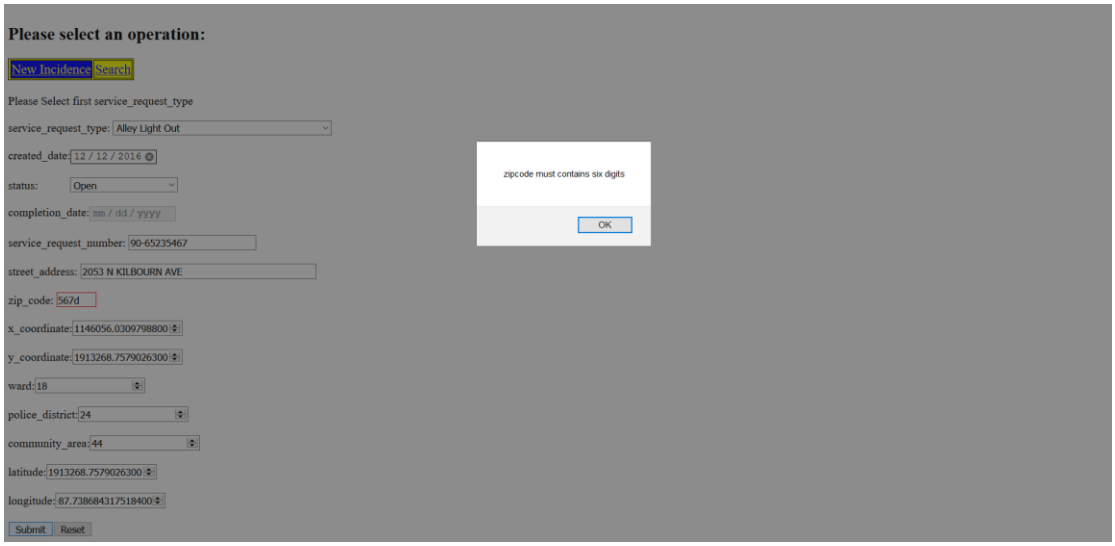
EIKONA 27



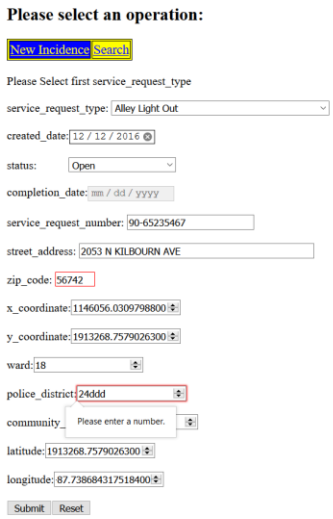
EIKONA 28



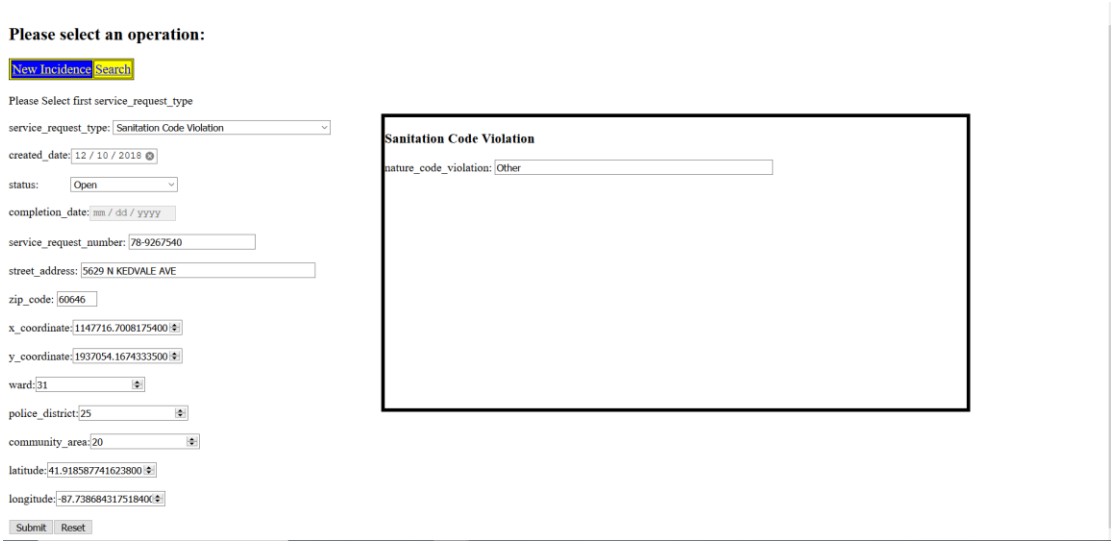
EIKONA 29



EIKONA 30



EIKONA 31



EIKONA 32

ΕΙΚΟΝΑ 33

Στο παραδοτέο εκτός από αυτό το pdf συμπεριλαμβάνονται:

- Εικόνα ER Model
- ένα rar που περιλαμβάνει όλα τα αρχεία κώδικά(.php,html,.js και βιβλιοθήκη PHP-JWT) εκτός από αυτό της βάσης δεδομένων που ήταν πολύ μεγάλο. Για αυτό το λόγο συμπεριλαμβάνεται και link για το rar που περιλαμβάνει και το αρχείο της βάσης δεδομένων όπως προέκυψε από export στο pgadmin https://drive.google.com/open?id=1N-dqZbrwsP3Be8pKRAzfmTTicjAMPNNk