



HELLENIC REPUBLIC
**National and Kapodistrian
University of Athens**
—EST. 1837—

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

Ανάπτυξη Βάσης Δεδομένων και Διαδικτυακής Εφαρμογής LA Crime tracker

Code repository:
<https://github.com/panagiotagyft/LA-Crime-Tracker.git>



LA Crimes tracker

Συστήματα Βάσεων Δεδομένων
(M149)

Διδάσκοντας: Αλέξης Δελής
Παναγιώτα Γύφτου, 7115112400025
Παναγιώτης Κεχρινιώτης: 711511240032

Περιεχόμενα

1. Συμβολή Συνεργατών.....	3
2. Σχήμα βάσης δεδομένων.....	4
2.1. Σχεδιαστικές επιλογές οργάνωσης βάσης σε πίνακες.....	4
2.2. Ευρετήρια.....	4
2.3. E/R Schema.....	5
2.4. Παραδοχές.....	5
3. Κώδικας SQL των <i>Queries</i>	6
3.1. Query1.....	6
3.2. Query2.....	6
3.3. Query3.....	7
3.4. Query4.....	7
3.5. Query5.....	8
3.6. Query6.....	8
3.7. Query7.....	9
3.8. Query8.....	10
3.9. Query9.....	11
3.10. Query10.....	12
3.11. Query11.....	13
3.12. Query12.....	14
3.13. Query13.....	15

4. Web Εφαρμογή.....	16
4.1. Σχεδιάγραμμα Εφαρμογής.....	16
4.2. Διαχείριση Αυθεντικοποίησης και Προστασίας Περιοχών με Django και React: Εγγραφή, Σύνδεση και Ανακατεύθυνση.....	17
4.3. Τεχνικές Βελτιστοποίησης.....	18
4.3.1. Insert.....	18
4.3.2. Update.....	19
4.3.3. Search.....	19
5. Στιγμιότυπα της εφαρμογής.....	20
6.1. Queries.....	20
6.2. Εφαρμογή.....	27
6. Χρήσιμες εντολές.....	33

Συμβολή Συνεργατών

Παναγιώτης Κεχρινιώτης

- Queries: 1, 3, 5, 7, 9, 11
- Indices

Παναγιώτα Γύφτου

- Queries: 2, 4, 6, 8, 10, 12, 13
- Updates, Insert, Search
- Υλοποίηση full stack εφαρμογής (React, Django, Postgresql)
- Δημιουργία script για τη φόρτωση των δεδομένων στους πίνακες

Από κοινού

- Σχεδιασμός του σχήματος της βάσης

Σχήμα βάσης δεδομένων

2.1. Σχεδιαστικές επιλογές οργάνωσης βάσης σε πίνακες

Προσπαθήσαμε να μειώσουμε την επανάληψη πληροφορίας και να δημιουργήσουμε tables/relations όπου υπάρχει νόημα και δεν περιέπλεκε άσκοπα την βάση μας.

Γενικά με όσα πεδία είχαν άμεση συσχέτιση με άλλα που τα εξηγούν (πχ περιγραφή) ή τους προσδίδουν πληροφορία τα βγάλαμε από το βασικό relation Crime_report και τα ορίσαμε ως ξεχωριστά relations (π.χ. Victim, Weapon), ώστε να αποφευχθεί η επανάληψη της πληροφορίας περιγραφής του εκάστου κωδικού, καθώς το συγκεκριμένο είδος πεδίου είναι τύπου text όπου δεσμεύει αρκετή μνήμη σε σχέση π.χ. με μια αριθμητική τιμή.

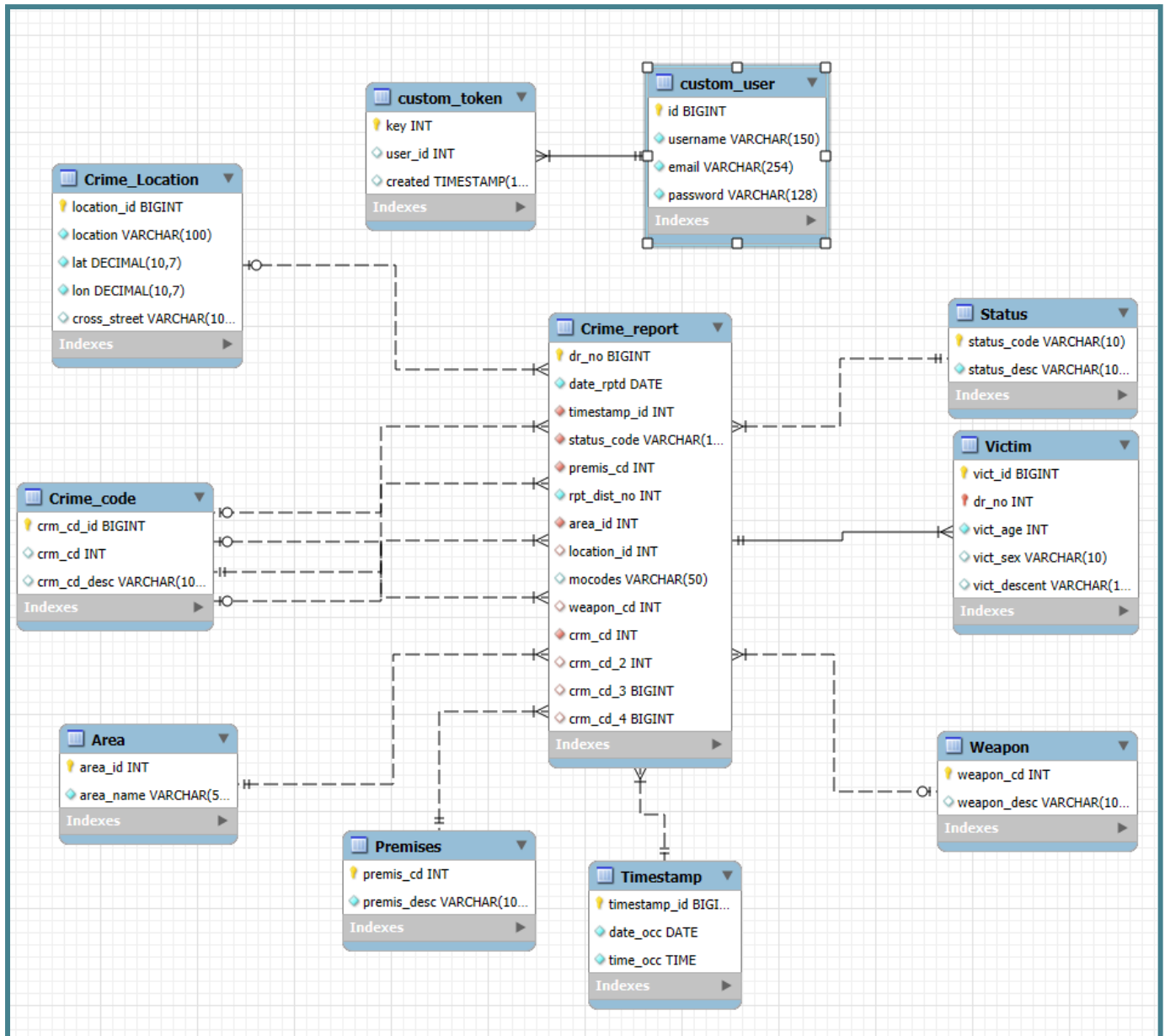
Η επιλογή της τοποθέτησης του πεδίου rpt_dist_no να περιλαμβάνεται στο πίνακα Crime_report, και όχι σε ξεχωριστό πίνακα με το πεδίο area_id, με το οποίο συνδέεται, καθώς το πρώτο είναι υπό-τμήμα (υπό-περιοχή) του δεύτερου πεδίου, για τον λόγο ότι θα επαναλαμβάνεται πληροφορία, συγκεκριμένα το όνομα της περιοχής και το id αντίστοιχα.

Χωρίσαμε επιπλέον την ώρα και την ημερομηνία από το Crime_report relation αφού δεν ήταν μοναδικά αυτά τα πεδία, και για να αυξήσουμε την απόδοση της βάσης, τους δημιουργήσαμε ρητά ένα πρωτεύων κλειδί. Παρόμοιο σκεπτικό ακολουθήσαμε στο Crime_code, καθώς και στο Crime_Location.

2.2. Ευρετήρια

Τα ευρετήρια που δημιουργήθηκαν είναι απαραίτητα στα QUERIES, διότι βελτιστοποιούν την απόδοση των πιο κρίσιμων λειτουργιών, όπως οι συνδέσεις (JOIN), το φιλτράρισμα δεδομένων (WHERE), η ομαδοποίηση (GROUP BY) και η ταξινόμηση (ORDER BY). Συγκεκριμένα, για να βρεθεί ο συνολικός αριθμός αναφορών ανά "Crm Cd" ή περιοχή, να υπολογιστεί ο μέσος αριθμός εγκλημάτων ανά ώρα, ή να εντοπιστούν τα πιο κοινά εγκλήματα ή όπλα ή συγκεκριμένα γεωγραφικά ή χρονικά πλαίσια, τα ευρετήρια στις στήλες crm_cd, date_occ, area_id, weapon_cd και lat/lon μειώνουν σημαντικά τον χρόνο αναζήτησης. Επιπλέον, για πιο πολύπλοκα ερωτήματα, όπως ο εντοπισμός ζευγών εγκλημάτων ή η εύρεση της μεγαλύτερης περιόδου χωρίς συγκεκριμένο έγκλημα, τα ευρετήρια στις στήλες rpt_dist_no, timestamp_id και date_rptd διευκολύνουν τη γρήγορη πρόσβαση στα δεδομένα και βελτιώνουν τη συνολική απόδοση της βάσης, ιδιαίτερα όταν διαχειρίζεται μεγάλους όγκους δεδομένων.

2.3. E/R Schema



2.4. Παραδοχές

Έχουμε κάνει την παραδοχή ότι οι NULL τιμές σε αριθμητικές τιμές αναγνωρίζεται με το **-1**.

Κώδικας SQL των *Queries*

Query 1

Find the total number of reports per “Crm Cd” that occurred within a specified time range and sort them in a descending order.

```
sql = ""  
    SELECT cd.crm_cd, COUNT(*) AS report_count  
    FROM Crime_report AS rpt  
    JOIN Crime_code AS cd ON rpt.crm_cd = cd.crm_cd_id  
    JOIN Timestamp AS ts ON rpt.timestamp_id = ts.timestamp_id  
    WHERE ts.time_occ BETWEEN %s AND %s  
    GROUP BY cd.crm_cd  
    ORDER BY report_count DESC;  
""
```

Query 2

Find the total number of reports per day for a specific “Crm Cd” and time range.

```
sql=""  
    SELECT report.date_rptd, COUNT(report.dr_no) as total_reports  
    FROM Crime_Report AS report  
    JOIN Timestamp AS time ON report.timestamp_id = time.timestamp_id  
    JOIN Crime_code AS code ON report.crm_cd = code.crm_cd_id  
    WHERE code.crm_cd = %s AND time.time_occ BETWEEN %s AND %s  
    GROUP BY report.date_rptd  
    ORDER BY total_reports DESC;  
""
```

Query 3

Find the most common crime committed regardless of code 1, 2, 3, and 4, per area for a specific day.

```
sql="""
    WITH FilteredCrimes AS (
        SELECT Area.area_id, Crime_report.crm_cd, crm_cd_2, crm_cd_3, crm_cd_4
        FROM Crime_report
        JOIN Area ON Crime_report.area_id = Area.area_id
        WHERE date_rptd = %s
    ),
    FlattenedCrimes AS (
        SELECT area_id, FilteredCrimes.crm_cd AS crime_code
        FROM FilteredCrimes
        UNION ALL
        SELECT area_id, FilteredCrimes.crm_cd_2 AS crime_code
        FROM FilteredCrimes
        UNION ALL
        SELECT area_id, FilteredCrimes.crm_cd_3 AS crime_code
        FROM FilteredCrimes
        UNION ALL
        SELECT area_id, crm_cd_4 AS crime_code
        FROM FilteredCrimes
    ),
    CrimeCounts AS (
        SELECT area_id, crime_code, COUNT(*) AS crime_count
        FROM FlattenedCrimes
        GROUP BY area_id, crime_code
    )
    SELECT DISTINCT ON (CrimeCounts.area_id)
        CrimeCounts.area_id,
        Crime_code.crm_cd,
        CrimeCounts.crime_count
    FROM CrimeCounts
    JOIN Crime_code ON CrimeCounts.crime_code = Crime_code.crm_cd_id
    WHERE Crime_code.crm_cd != -1
    ORDER BY CrimeCounts.area_id, crime_count DESC;
    """
```

Query 4

Find the average number of crimes occurred per hour (24 hours) for a specific date range.

```
sql = """
    SELECT COUNT(report.dr_no) / (COUNT(DISTINCT time.date_occ) * 24) AS avg_number_of_crimes_per_hour
    FROM Crime_report AS report
    JOIN Timestamp AS time ON report.timestamp_id = time.timestamp_id
    WHERE time.date_occ BETWEEN %s AND %s;
    """
```


Query 5

Find the most common “Crm Cd” in a specified bounding box(as designated by GPS-coordinates) for a specific day.

```
sql = """
    SELECT cc.crm_cd, COUNT(*) AS frequency
    FROM crime_report cr
    JOIN crime_location cl ON cr.location_id = cl.location_id
    JOIN Crime_code cc ON cr.crm_cd = cc.crm_cd_id
    JOIN Timestamp ts ON cr.timestamp_id = ts.timestamp_id
    WHERE ts.date_occ = %s
    AND cl.lat >= %s AND cl.lat <= %s
    AND cl.lon >= %s AND cl.lon <= %s
    GROUP BY cc.crm_cd
    ORDER BY frequency DESC
    LIMIT 5;
    """
```

Query 6

Find the top-5 Area names with regards to total number of crimes reported per day for a specific date range. The same for Rpt Dist No.

```
if input_type == 'area_name':
    sql = """
        SELECT area.area_name AS name, COUNT(report.dr_no) AS total_crimes
        FROM Crime_Report AS report
        JOIN Area ON area.area_id = report.area_id
        WHERE report.date_rptd BETWEEN %s AND %s
        GROUP BY area.area_name
        ORDER BY total_crimes DESC
        LIMIT 5
        """
else:
    sql = """
        SELECT rpt_dist_no, COUNT(dr_no) AS total_crimes
        FROM Crime_Report
        WHERE date_rptd BETWEEN %s AND %s
        GROUP BY rpt_dist_no
        ORDER BY total_crimes DESC
        LIMIT 5
        """
```

Query 7

Find the pair of crimes that has co-occurred in the area with the most reported incidents for a specific date range.

```
sql = """
    WITH AreaIncidentCounts AS (
        SELECT area_id
        FROM crime_report
        JOIN Timestamp ON crime_report.timestamp_id = Timestamp.timestamp_id
        WHERE date_occ BETWEEN %s AND %s
        GROUP BY area_id
        ORDER BY COUNT(*) DESC
        LIMIT 1
    )
    SELECT
        cc1.crm_cd AS crime1,
        cc2.crm_cd AS crime2,
        COUNT(*) AS co_occurrence_count
    FROM crime_report cr1
    JOIN crime_report cr2
        ON cr1.area_id = cr2.area_id
        AND cr1.timestamp_id = cr2.timestamp_id
        AND cr1.dr_no < cr2.dr_no
    JOIN Crime_code cc1 ON cc1.crm_cd_id = cr1.crm_cd
    JOIN Crime_code cc2 ON cc2.crm_cd_id = cr2.crm_cd AND cc1.crm_cd_id < cc2.crm_cd_id
    JOIN Timestamp ts1 ON cr1.timestamp_id = ts1.timestamp_id
    WHERE cr1.area_id = (SELECT area_id FROM AreaIncidentCounts)
    AND ts1.date_occ >= %s AND ts1.date_occ <= %s
    GROUP BY crime1, crime2
    ORDER BY co_occurrence_count DESC;
    """
```

Query 8

Find the second most common crime that has co-occurred with a particular crime for a specific date range.

```
sql=""
WITH AllPairs AS (
  -- Step 1: Generate all possible pairs of crimes using a self-join
  SELECT
    c1.crm_cd AS crime1, c1.crm_cd_id AS crime1_id, -- The first crime in the pair
    c2.crm_cd AS crime2, c2.crm_cd_id AS crime2_id -- The second crime in the pair
  FROM
    Crime_code c1
  INNER JOIN
    Crime_code c2 ON c1.crm_cd_id < c2.crm_cd_id -- Ensure unique pairs by comparing IDs
  WHERE c1.crm_cd <> c2.crm_cd AND c1.crm_cd <> -1 AND c2.crm_cd <> -1
),
FilteredPairs AS (
  -- Step 2: Filter pairs involving the specific crime
  SELECT
    crime1, crime1_id,
    crime2, crime2_id
  FROM
    AllPairs
  WHERE
    %s IN (crime1, crime2)
),
PairOccurrences AS (
  -- Step 3: Find matching records for each pair and group them
  SELECT
    fp.crime1, -- The first crime in the pair
    fp.crime2, -- The second crime in the pair
    cr.dr_no -- Unique incident identifier
  FROM FilteredPairs AS fp
  JOIN Crime_report cr ON (
    (fp.crime1_id = cr.crm_cd OR fp.crime1_id = cr.crm_cd_2 OR fp.crime1_id = cr.crm_cd_3 OR fp.crime1_id = cr.crm_cd_4)
    AND
    (fp.crime2_id = cr.crm_cd OR fp.crime2_id = cr.crm_cd_2 OR fp.crime2_id = cr.crm_cd_3 OR fp.crime2_id = cr.crm_cd_4)
  )
  WHERE cr.date_rptd BETWEEN %s AND %s
),
GroupedPairCounts AS (
  -- Step 4: Count the number of occurrences for each pair
  SELECT crime1, crime2, COUNT(*) AS pair_count -- Count how many times this pair occurred
  FROM PairOccurrences
  GROUP BY crime1, crime2 -- Group by each unique pair
)
-- Step 5: Sort the results by the number of occurrences
SELECT crime1, crime2, pair_count
FROM GroupedPairCounts
ORDER BY pair_count DESC
LIMIT 1 OFFSET 1;
""
```

Query 9

Find the most common type of weapon used against victims depending on their group of age.

The age groups are formed by bucketing ages every 5 years, e.g., $0 \leq x < 5$, $5 \leq x < 10$, etc..

```
sql=""
WITH WeaponFrequency AS (
    SELECT
        FLOOR(v.vict_age / 5) * 5 AS age_group,
        w.weapon_cd,
        w.weapon_desc,
        COUNT(*) AS frequency
    FROM victim v
    JOIN Crime_report cr ON v.dr_no = cr.dr_no
    JOIN Weapon w ON cr.weapon_cd = w.weapon_cd
    WHERE v.vict_age IS NOT NULL AND v.vict_age > 0
        AND w.weapon_cd > 0
    GROUP BY age_group, w.weapon_cd, w.weapon_desc
)
SELECT DISTINCT ON (age_group)
    age_group,
    weapon_cd,
    weapon_desc,
    frequency
FROM WeaponFrequency
ORDER BY age_group, frequency DESC;
""
```

Query 10

Find the area with the longest time range without an occurrence of a specific crime. Include the time range in the results. The same for Rpt Dist No.

```

if input_type == 'area_name':
    sql = """
    -- Create a table with unique crime dates for each area
    WITH area_dates AS (
        SELECT area_name, date.date_occ AS date
        FROM Crime_report AS cr
        INNER JOIN Area AS a ON a.area_id = cr.area_id
        JOIN Crime_code AS code ON code.crm_cd_id = cr.crm_cd
        JOIN Timestamp AS date ON date.timestamp_id = cr.timestamp_id
        WHERE code.crm_cd = %s
        GROUP BY a.area_name, date.date_occ
    )

    -- Final query to find the longest time gap without the specific crime
    SELECT a1.area_name,
        a1.date AS start_date, -- Start of the gap period
        MIN(a2.date) AS end_date, -- Earliest next date (end of the gap period)
        MIN(a2.date) - a1.date AS gap -- Calculate the time gap
    FROM area_dates AS a1
    INNER JOIN area_dates AS a2 ON a1.area_name = a2.area_name AND a1.date < a2.date -- Ensure that a2.date is after a1.date
    GROUP BY a1.area_name, a1.date
    ORDER BY gap DESC
    LIMIT 1;
    """
else:
    sql = """
    WITH rpt_dist_dates AS (
        SELECT cr.rpt_dist_no AS rpt_dist_no, date.date_occ AS date
        FROM Crime_report AS cr
        JOIN Crime_code AS code ON code.crm_cd_id = cr.crm_cd
        JOIN Timestamp AS date ON date.timestamp_id = cr.timestamp_id
        WHERE code.crm_cd = %s
        GROUP BY cr.rpt_dist_no, date
    )

    SELECT a1.rpt_dist_no, a1.date AS start_date, MIN(a2.date) AS end_date, MIN(a2.date) - a1.date AS gap
    FROM rpt_dist_dates AS a1
    INNER JOIN rpt_dist_dates AS a2 ON a1.rpt_dist_no = a2.rpt_dist_no AND a1.date < a2.date
    GROUP BY a1.rpt_dist_no, a1.date
    ORDER BY gap DESC
    LIMIT 1;
    """

```

Query 11

For 2 crimes of your choice, find all areas that have received more than 1 report on each of these 2 crimes on the same day. The 2 crimes could be for example: “CHILD ANNOYING (17YRS & UNDER)” or “THEFT OF IDENTITY”. Do not restrict yourself to just these 2 specific types of crimes of course!

```
sql = """
    WITH FilteredCrimes AS (
        SELECT
            a.area_name,
            ts.date_occ AS report_date,
            cc.crm_cd_desc,
            COUNT(*) AS report_count
        FROM crime_report cr
        JOIN area a ON cr.area_id = a.area_id
        JOIN Crime_code cc ON cr.crm_cd = cc.crm_cd_id
        JOIN Timestamp ts ON cr.timestamp_id = ts.timestamp_id
        WHERE cc.crm_cd_desc IN (%s, %s)
        GROUP BY a.area_name, ts.date_occ, cc.crm_cd_desc
    ),
    CrimeCounts AS (
        SELECT
            area_name,
            report_date,
            COUNT(DISTINCT crm_cd_desc) AS distinct_crimes,
            report_count
        FROM FilteredCrimes
        GROUP BY area_name, report_date, report_count
    )
    SELECT
        area_name
    FROM CrimeCounts
    WHERE distinct_crimes = 2
    GROUP BY area_name
    """
```

Query 12

Find the number of division of records for crimes reported on the same day in different areas using the same weapon for a specific time range.

```
sql = """
    WITH dr_date_weapon AS(
        SELECT cr.dr_no, cr.date_rptd, cr.weapon_cd, cr.area_id
        FROM Crime_report AS cr
        JOIN Timestamp AS time ON time.timestamp_id = cr.timestamp_id
        WHERE time.time_occ BETWEEN %s AND %s AND cr.weapon_cd <> -1
        GROUP BY cr.dr_no, cr.date_rptd, cr.weapon_cd, cr.area_id
        ORDER BY cr.date_rptd, cr.weapon_cd, cr.area_id
    ),
    RemoveDuplicates AS(
        SELECT date_rptd, weapon_cd, area_id
        FROM dr_date_weapon
        GROUP BY date_rptd, weapon_cd, area_id
        HAVING COUNT(area_id) = 1
        ORDER BY date_rptd, weapon_cd, area_id
    ),
    AllRes AS (
        SELECT date_rptd, weapon_cd, COUNT(*) as num_drno
        FROM RemoveDuplicates
        GROUP BY date_rptd, weapon_cd
    )
    SELECT date_rptd, weapon_cd, num_drno
    FROM AllRes
    WHERE num_drno > 1;
    """
```

Query 13

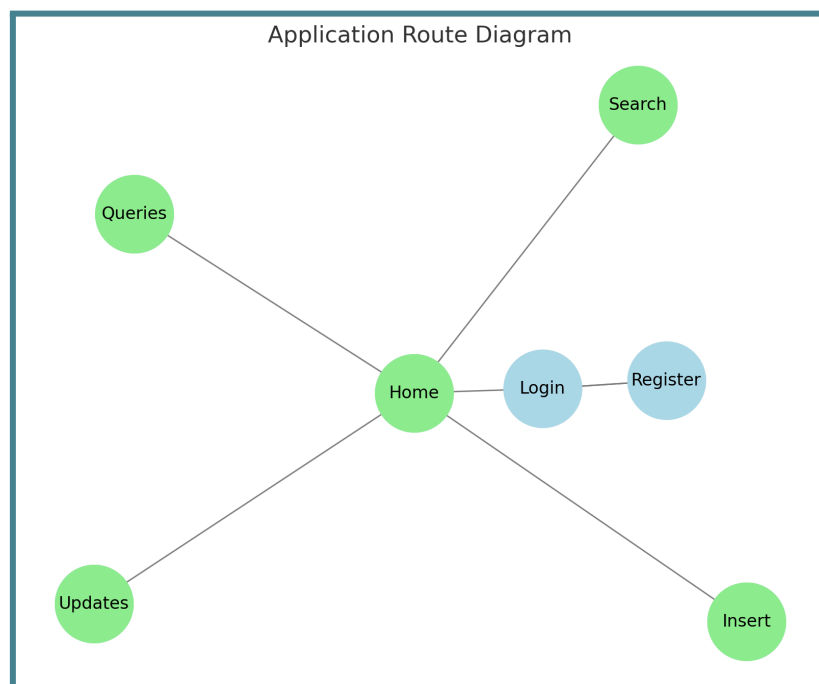
Find the crimes that occurred for a given number of times N on the same day, in the same area, using the same weapon, for a specific time range. Return the list of division of records numbers, the area name, the crime code description and the weapon description.

```
sql = """
    WITH Filtered_Crimes AS (
        SELECT
            cr.area_id AS area_id,
            cr.crm_cd AS crime_code_id,
            cr.weapon_cd AS weapon_cd,
            ts.date_occ AS date,
            COUNT(cr.dr_no) AS counter
        FROM Crime_report cr
        JOIN Timestamp ts ON cr.timestamp_id = ts.timestamp_id
        WHERE ts.time_occ BETWEEN %s AND %s
        GROUP BY cr.area_id, cr.crm_cd, cr.weapon_cd, ts.date_occ
        HAVING COUNT(cr.dr_no) = %s
    ),
    DRCODES AS (
        SELECT
            cr.dr_no,
            cr.area_id AS area_id,
            cr.crm_cd AS crime_code_id,
            cr.weapon_cd AS weapon_cd,
            ts.date_occ AS date
        FROM Filtered_Crimes fc
        JOIN Timestamp ts ON ts.date_occ = fc.date
        JOIN Crime_report cr ON
            fc.area_id = cr.area_id AND
            fc.crime_code_id = cr.crm_cd AND
            fc.weapon_cd = cr.weapon_cd AND
            ts.timestamp_id = cr.timestamp_id
        ORDER BY cr.area_id, cr.crm_cd, cr.weapon_cd, ts.date_occ
    )
    SELECT d.dr_no, a.area_name, code.crm_cd_desc, wp.weapon_desc
    FROM DRCODES d
    JOIN Area a ON a.area_id = d.area_id
    JOIN Crime_code code ON code.crm_cd_id = d.crime_code_id
    JOIN Weapon wp ON wp.weapon_cd = d.weapon_cd
    WHERE code.crm_cd <> -1 AND wp.weapon_cd <> -1
    ORDER BY a.area_name, code.crm_cd_desc, wp.weapon_desc;
    """
```


Web Εφαρμογή

4.1. Σχεδιάγραμμα Εφαρμογής

1. **Αρχική Διαδρομή (/)**
 - Εμφανίζει τη σελίδα **Login**.
2. **Διαδρομή /login**
 - Εμφανίζει τη σελίδα **Login**.
3. **Διαδρομή /register**
 - Εμφανίζει τη σελίδα **Register**.
4. **Προστατευμένες Διαδρομές (απαιτείται σύνδεση χρήστη μέσω του *ProtectedRoute*):**
 - **/home**: Εμφανίζει τη σελίδα **Home**.
 - **/queries**: Εμφανίζει τη σελίδα **Queries**.
 - **/insert**: Εμφανίζει τη σελίδα **Insert**.
 - **/updates**: Εμφανίζει τη σελίδα **Updates**.
 - **/search**: Εμφανίζει τη σελίδα **Search**.
5. **Διαχείριση Διαδρομών**
 - Το `createBrowserRouter` συνδέει κάθε διαδρομή με το αντίστοιχο στοιχείο.
 - Το `RouterProvider` προσφέρει τη λειτουργικότητα πλοήγησης στην εφαρμογή.



4.2. Διαχείριση Αυθεντικοποίησης και Προστασίας Περιοχών με Django και React: Εγγραφή, Σύνδεση και Ανακατεύθυνση

Η εγγραφή του χρήστη επιτυγχάνεται μέσω του `CustomUserManager` στο backend, το οποίο επαληθεύει τα δεδομένα (π.χ., email), κρυπτογραφεί τον κωδικό πρόσβασης και αποθηκεύει τα στοιχεία στον πίνακα `custom_user` στη βάση δεδομένων. Κατά τη σύνδεση, τα στοιχεία που παρέχει ο χρήστης (όνομα χρήστη και κωδικός πρόσβασης) ελέγχονται έναντι των αποθηκευμένων στο backend, και αν είναι σωστά, δημιουργείται ή επιστρέφεται ένα token από τον πίνακα `custom_token`. Το token αυτό αποστέλλεται στο frontend, όπου αποθηκεύεται σε cookies και χρησιμοποιείται στα αιτήματα προς προστατευμένα endpoints. Στο React, οι διαδρομές προστατεύονται με το component `ProtectedRoute`, το οποίο ελέγχει την ύπαρξη έγκυρου token πριν επιτρέψει την πρόσβαση σε σελίδες όπως `/home` ή `/queries`. Αν το token λείπει ή δεν είναι έγκυρο, ο χρήστης ανακατευθύνεται στη σελίδα σύνδεσης (`/login`).

Πιο συγκεκριμένα:

- Εγγραφή Χρήστη (Registration)

Η εγγραφή ελέγχει τη μοναδικότητα του ονόματος χρήστη, τη μορφή του email και τις τιμές των κωδικών.

Βήματα:

1. Επαλήθευση ότι όλα τα πεδία (username, email, password, confirm_password) είναι συμπληρωμένα.
2. Έλεγχος μορφής email.
3. Διασφάλιση ότι οι κωδικοί πρόσβασης ταιριάζουν (password, confirm_password).
4. Έλεγχος αν το username είναι ήδη καταχωρημένο.
5. Κρυπτογράφηση του κωδικού με `make_password`.
6. Δημιουργία νέου χρήστη στον πίνακα `custom_user`.
7. Δημιουργία token και αποθήκευσή του στον πίνακα `custom_token`.
8. Επιστροφή του token ως απάντηση.

- Σύνδεση Χρήστη (Login)

Η σύνδεση ελέγχει τα credentials του χρήστη και είτε επιστρέφει υπάρχον token είτε δημιουργεί νέο.

Βήματα:

1. Επαλήθευση ότι τα πεδία username και password είναι συμπληρωμένα.
2. Αναζήτηση χρήστη στον πίνακα custom_user και ανάκτηση του κρυπτογραφημένου κωδικού.
3. Επαλήθευση του κωδικού πρόσβασης με check_password.
4. Έλεγχος για υπάρχον token στον πίνακα custom_token.
5. Δημιουργία νέου token αν δεν υπάρχει.
6. Επιστροφή του token ως απάντηση.

4.3. Τεχνικές Βελτιστοποίησης

4.3.1 Insert

Η δημιουργία του **DR_NO** γίνεται αυτόματα, όταν ο χρήστης επιλέγει τις απαραίτητες παραμέτρους για την δημιουργία του και υλοποιείται από το backend στη μέθοδο GenerateDRNOView του functions_views.py. Πιο αναλυτικά η παραγωγή του **DR_NO** βασίζεται σε δύο παραμέτρους: το area_id (κωδικός περιοχής) και την date_rptd (ημερομηνία αναφοράς). Αφού εξάγεται το έτος από την ημερομηνία, εκτελείται ένα ερώτημα στη βάση δεδομένων για να υπολογιστεί ο επόμενος διαθέσιμος αριθμός αναφοράς για την περιοχή και το έτος. Το DR_NO σχηματίζεται με τη μορφή YY + AreaID + NextRecordNumber, όπου YY είναι τα δύο τελευταία ψηφία του έτους, το AreaID είναι ο κωδικός περιοχής, και το NextRecordNumber ένας πενταψήφιος αριθμός.

4.3.2. Update

Λογω του μεγάλου όγκου δεδομένων, για την βέλτιστη χρήση της μνήμης όταν ο χρήστης θέλει να κάνει ένα update μιας αναφοράς, για να την βρει και να εξάγει τις τρέχοντες πληροφορίες της, το σύστημα αναζητά μερικώς την εγγραφή. Δηλαδή, ο χρήστης πληκτρολογώντας μέρος του αριθμού αναφοράς `dr_no` ζητά από την βάση όχι όλες τις αναφορές αλλά ένα μέρος αυτών, επιτυγχάνοντας την άμεση λήψη της. Η λειτουργία **search** υλοποιείται μέσω της κλάσης `SearchDRNumbersView`, επιτρέποντας την αναζήτηση αριθμών DR (`dr_no`) που ταιριάζουν με ένα ερώτημα (query) στο πεδίο `Crime_report.dr_no`. Το API λαμβάνει το ερώτημα μέσω GET request και, χρησιμοποιώντας SQL εντολή με τη λέξη-κλειδί `LIKE %query%`, επιστρέφει αποτελέσματα που ταιριάζουν μερικώς στο query.

4.3.3. Search

Η λειτουργία **paging** στην εφαρμογή επιτρέπει την αναζήτηση δεδομένων με δυνατότητα σελιδοποίησης. Η σελιδοποίηση εξασφαλίζει ότι τα αποτελέσματα αναζήτησης χωρίζονται σε μικρότερες ομάδες (σελίδες), βελτιώνοντας την απόδοση και την εμπειρία του χρήστη.

Κατά την αναζήτηση, η εφαρμογή στέλνει αιτήματα GET στο backend με την παράμετρο `page`, η οποία καθορίζει την τρέχουσα σελίδα. Το backend επιστρέφει τα δεδομένα της συγκεκριμένης σελίδας, μαζί με τον συνολικό αριθμό σελίδων. Στη διεπαφή χρήστη, υπάρχει ένα στοιχείο `Pagination`, που εμφανίζει τα κουμπιά πλοήγησης (π.χ., "Previous", "Next") και τους αριθμούς σελίδων. Όταν ο χρήστης επιλέγει διαφορετική σελίδα, η παράμετρος `page` ενημερώνεται, και τα νέα δεδομένα φορτώνονται δυναμικά μέσω του `handlePageChange`.

Στιγμιότυπα της εφαρμογής

5.1. Queries

Query 1

1. Find the total number of reports per "Crm Cd" that occurred within a specified time range and sort them in a descending order.

Start Time: 02:18 μμ End Time: 02:19 μμ

Results:

CRIME CODE	REPORT COUNT
440	39
442	34
354	17
343	12
624	10
740	9
310	8
220	7
330	7
341	7
510	6
956	5
761	4
888	4

Reset

Query 2

2. Find the total number of reports per day for a specific "Crm Cd" and time range.

Start Time: 02:46 μμ End Time: 03:00 μμ Crime Code: 250

Results:

DATE	REPORT COUNT
2024-01-12	2
2020-04-04	1
2020-06-02	1
2021-10-11	1
2021-10-24	1
2022-07-01	1

Reset

Query 3

3. Find the most common crime committed regardless of code 1, 2, 3, and 4, per area for a specific day.

Date
01/01/2024

Results:

AREA CODE	THE MOST FREQUENT CRIME
1	440
2	210
3	230
4	624
5	998
6	624
7	310
8	740
9	310
10	626
11	330
12	761
13	230
14	998

Reset

Query 4

4. Find the average number of crimes occurred per hour (24 hours) for a specific date range.

Start Date
01/01/2023

End Date
05/01/2023

Average Number of Crimes Per Hour:
33

Reset

Query 5

5. Find the most common "Crm Cd" in a specified bounding box (as designated by GPS-coordinates) for a specific day.

Date
26/09/2020

Min Latitude: 33,000 Max Latitude: 34,500 Min Longitude: -119 Max Longitude: -118

Results:

THE MOST FREQUENT CRIME	CRIME COUNT
510	62
624	50
230	45
740	43
626	31

Reset

Query 6

6. Find the top-5 Area names with regards to total number of crimes reported per day for a specific date range. The same for Rpt Dist No.

Select Input Type: Start Date End Date

Area Name 09/01/2024 11/01/2024

Results:

AREA NAME	TOTAL CRIMES
Central	129
Pacific	109
N Hollywood	108
Wilshire	107
77th Street	105

Reset

6. Find the top-5 Area names with regards to total number of crimes reported per day for a specific date range. The same for Rpt Dist No.

Select Input Type: Start Date End Date

Rpt Dist No 09/01/2024 11/01/2024

Results:

RPT DIST NO	TOTAL CRIMES
1764	13
162	12
2156	12
1239	12
1494	11

Reset

Query 7

7. Find the pair of crimes that has co-occurred in the area with the most reported incidents for a specific date range.

Start Date End Date

01/01/2020 01/01/2020




Results:

CRIME 1	CRIME 2	PAIR COUNT
420	815	6
420	860	3
420	627	3
354	420	2
354	121	2
354	815	2
354	921	2
354	860	2
627	815	1
812	860	1
121	921	1
331	921	1
331	121	1
625	921	1

Reset

Query 8

8. Find the second most common crime that has co-occurred with a particular crime for a specific date range.

Start Date: 01/01/2024  End Date: 18/01/2024  Crime Code: 330 

Results:

CRIME 1	CRIME 2	PAIR COUNT
330	310	2

[Reset](#)

Query 9

9. Find the most common type of weapon used against victims depending on their group of age. The age groups are formed by bucketing ages every 5 years.

Results:

AGE GROUP	MOST COMMON WEAPON	OCCURRENCE COUNT
0	400	STRONG-ARM (HANDS, FIST, FEET OR BODILY FORCE)
5	400	STRONG-ARM (HANDS, FIST, FEET OR BODILY FORCE)
10	400	STRONG-ARM (HANDS, FIST, FEET OR BODILY FORCE)
15	400	STRONG-ARM (HANDS, FIST, FEET OR BODILY FORCE)
20	400	STRONG-ARM (HANDS, FIST, FEET OR BODILY FORCE)
25	400	STRONG-ARM (HANDS, FIST, FEET OR BODILY FORCE)
30	400	STRONG-ARM (HANDS, FIST, FEET OR BODILY FORCE)
35	400	STRONG-ARM (HANDS, FIST, FEET OR BODILY FORCE)

[Reset](#)

Query 10

10. Find the area with the longest time range without an occurrence of a specific crime. Include the time range in the results. The same for Rpt Dist No.

Select Category Specific Crime

Area 113

Results:

AREA NAME	START DATE	END DATE	GAP
West Valley	2020-10-10	2022-02-03	481

Reset

10. Find the area with the longest time range without an occurrence of a specific crime. Include the time range in the results. The same for Rpt Dist No.

Select Category Specific Crime

Rpt Dist No 954

Results:

RPT DIST NO	START DATE	END DATE	GAP
1025	2022-11-08	2023-06-03	207

Reset

Query 11

11. For 2 crimes of your choice, find all areas that have received more than 1 report on each of these 2 crimes on the same day. The 2 crimes could be for example: "CHILD ANNOYING (17YRS & UNDER)" or "THEFT OF IDENTITY". Do not restrict yourself to just these 2 specific types of crimes of course!

Specific Crime 1: CHILD ANNOYING (17YRS & UNDER) Specific Crime 2: THEFT OF IDENTITY

Results:

AREA
77th Street
Central
Devonshire
Foothill
Harbor
Hollenbeck
Hollywood
Mission
N Hollywood
Newton
Northeast
Olympic
Pacific

Reset

Query 12

12. Find the number of division of records for crimes reported on the same day in different areas using the same weapon for a specific time range.

Start Time: 03:52 μμ End Time: 03:55 μμ

Results:

REPORTED DAY	WEAPON CODE	NUMBER OF DIVISION OF RECORDS
2020-03-19	400	2
2020-10-22	400	2
2022-03-18	400	2
2022-06-02	400	3
2023-03-20	400	2
2023-06-08	400	2
2023-07-23	102	2
2023-12-09	400	2
2024-02-01	400	2

Reset

Query 13

13. Find the total number of reports per day for a specific "N" and time range.

Start Time: 03:52 μμ End Time: 03:55 μμ N: 3

Results:

DR_NO	AREA NAME	CRIME CODE DESC	WEAPON DESC
200416184	Hollenbeck	BATTERY - SIMPLE ASSAULT	STRONG-ARM (HANDS, FIST, FEET OR BODILY FORCE)
200416183	Hollenbeck	BATTERY - SIMPLE ASSAULT	STRONG-ARM (HANDS, FIST, FEET OR BODILY FORCE)
200416182	Hollenbeck	BATTERY - SIMPLE ASSAULT	STRONG-ARM (HANDS, FIST, FEET OR BODILY FORCE)
211906169	Mission	BRANDISH WEAPON	HAND GUN
211906171	Mission	BRANDISH WEAPON	HAND GUN
211906170	Mission	BRANDISH WEAPON	HAND GUN
221116454	Northeast	ASSAULT WITH DEADLY WEAPON, AGGRAVATED ASSAULT	HAND GUN
221116453	Northeast	ASSAULT WITH DEADLY WEAPON, AGGRAVATED ASSAULT	HAND GUN
221100975	Northeast	ASSAULT WITH DEADLY WEAPON, AGGRAVATED ASSAULT	HAND GUN
211407701	Pacific	BATTERY - SIMPLE ASSAULT	STRONG-ARM (HANDS, FIST, FEET OR BODILY FORCE)

Reset

5.2. Εφαρμογή

Home Page

Welcome to the homepage!

Please select one of the options below:

Queries

Insert

Updates

Search

giota Logout

Queries


glota
Logout

- Find the total number of reports per "Crm Cd" that occurred within a specified time range and sort them in a descending order.

- Find the total number of reports per day for a specific "Crm Cd" and time range.

- Find the most common crime committed regardless of code 1, 2, 3, and 4, per area for a specific day.

- Find the average number of crimes occurred per hour (24 hours) for a specific date range.

- Find the most common "Crm Cd" in a specified bounding box (as designated by GPS-coordinates) for a specific day.

- Find the top-5 Area names with regards to total number of crimes reported per day for a specific date range. The same for Rpt Dist No.

- Find the pair of crimes that has co-occurred in the area with the most reported incidents for a specific date range.

- Find the second most common crime that has co-occurred with a particular crime for a specific date range.

- Find the most common type of weapon used against victims depending on their group of age. The age groups are formed by bucketing ages every 5 years.


- Find the area with the longest time range without an occurrence of a specific crime. Include the time range in the results. The same for Rpt Dist No.

- For 2 crimes of your choice, find all areas that have received more than 1 report on each of these 2 crimes on the same day. The 2 crimes could be for example: "CHILD ANNOYING (17YRS & UNDER)" or "THEFT OF IDENTITY". Do not restrict yourself to just these 2 specific types of crimes of course!

- Find the number of division of records for crimes reported on the same day in different areas using the same weapon for a specific time range.

- Find the total number of reports per day for a specific "N" and time range.

Insert

giota [Logout](#)

Insert

DR_NO

Date Rptd

mm/dd/yyyy

DATE OCC

mm/dd/yyyy

TIME OCC

--:--

Area Code

Select Area Code

Area Name

Crime Code

Select Crime Code

Crime Description

Crm Cd 2

Select Crime Code

Crm Cd 3

Select Crime Code

Crm Cd 4

Select Crime Code

Premis Cd

Select Premis Cd

Premis Desc

Weapon Used Code

Select Weapon Used Code

Weapon Description

Location

Latitude

Longitude

Cross Street

Status

Select Status

Status Description

Rpt Dist No

Select Rpt Dist No

Mocodes

Enter new Mocodes

Victim Age

Enter new Victim Age

Victim Sex


Select Victim Sex

Victim Descent

Select Victim Descent

Submit

Updates

giota [Logout](#)

Updates

DR_NO

Date Rptd

DATE OCC

TIME OCC

Area Code

Area Name

Crime Code

Crime Description

Crn Cd 2

Crn Cd 3

Crn Cd 4

Premis Cd

Premis Desc

Weapon Used Code

Weapon Description

Location

Latitude

Longitude

Cross Street

Status

Status Description

Rpt Dist No

Mocodes


Victim Age

Victim Sex

Victim Descent

Submit

Search

giota [Logout](#)

Area

Mission

Results:

DR_NO	DATE RPTD	DATE	TIME	STATUS	STATUS DESC	PREMIS CODE	PREMIS CODE DESC	RPT_DIST_NO	AREA ID	AREA NAME	LOACATION ID	LOCATION
221911806	2022-07-10	2020-03-25	00:01:00	IC	Invest Cont	501	SINGLE FAMILY DWELLING	1977	19	Mission	442	13800 RAYEN ST
221915337	2022-10-05	2020-07-08	06:00:00	AO	Adult Other	505	MOTEL	1921	19	Mission	663	12700 ENCINITAS A
221917036	2022-11-03	2020-11-03	14:15:00	AO	Adult Other	507	CONDOMINIUM/TOWNHOUSE	1974	19	Mission	205	9000 WILLIS AV
231905889	2023-02-13	2020-09-18	00:01:00	IC	Invest Cont	501	SINGLE FAMILY DWELLING	1964	19	Mission	1115	14700 GLEDHILL ST
221904163	2022-01-05	2020-11-17	11:50:00	IC	Invest Cont	701	HOSPITAL	1931	19	Mission	1227	15000 RINALC ST
201914024	2020-09-01	2020-09-01	04:30:00	IC	Invest Cont	121	YARD (RESIDENTIAL/BUSINESS)	1984	19	Mission	9300	8800 KESTER AV
201908598	2020-04-09	2020-04-05	09:00:00	AO	Adult Other	501	SINGLE FAMILY DWELLING	1902	19	Mission	9384	14300 FOOTHILL BL
201916813	2020-11-12	2020-11-12	12:31:00	IC	Invest Cont	122	VEHICLE, PASSENGER/TRUCK	1983	19	Mission	9412	14900 ROSCO BL

Previous

1

2

3

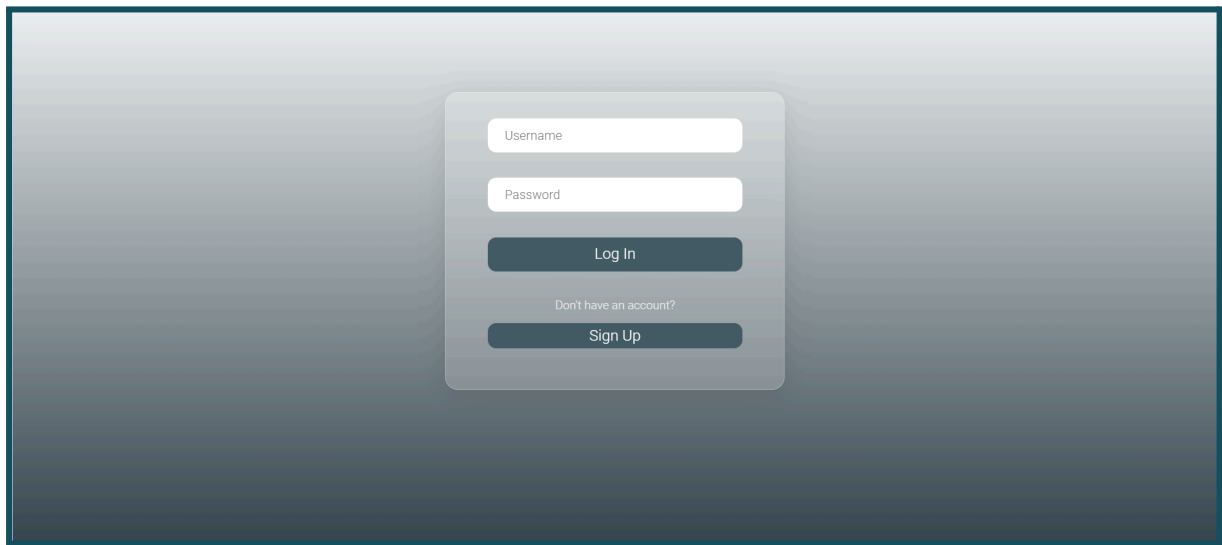
4

5

Next

Reset

Login Page



A login page with a dark blue gradient background. In the center is a light gray rounded rectangle containing the login form. The form has two white input fields labeled 'Username' and 'Password'. Below the 'Password' field is a dark blue button with the text 'Log In'. Underneath the button is the text 'Don't have an account?' followed by another dark blue button with the text 'Sign Up'.

Username

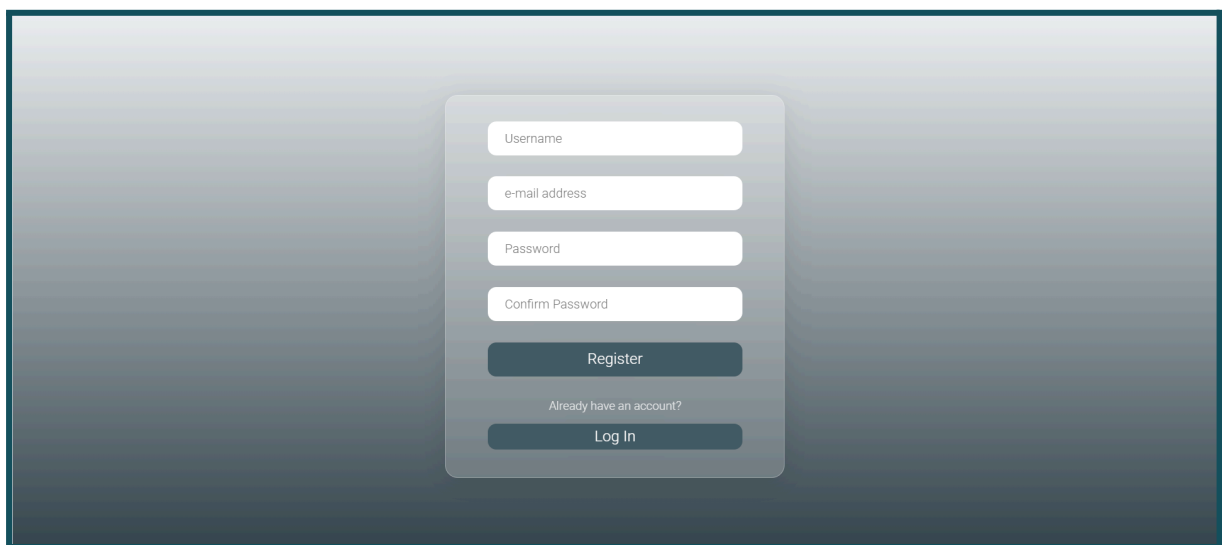
Password

Log In

Don't have an account?

Sign Up

Register Page



A register page with a dark blue gradient background. In the center is a light gray rounded rectangle containing the registration form. The form has four white input fields labeled 'Username', 'e-mail address', 'Password', and 'Confirm Password'. Below the 'Confirm Password' field is a dark blue button with the text 'Register'. Underneath the button is the text 'Already have an account?' followed by a dark blue button with the text 'Log In'.

Username

e-mail address

Password

Confirm Password

Register

Already have an account?

Log In

Χρήσιμες εντολές

Βάση Postgresql

```
sudo service postgresql restart  
psql -U postgres -h localhost  
psql -U crime_user -d crime_tracker -h localhost
```

React

```
cd my-app  
npm install  
npm start
```

Django

```
cd crime_backend  
python3 manage.py makemigrations  
python3 manage.py migrate  
python3 manage.py runserver
```