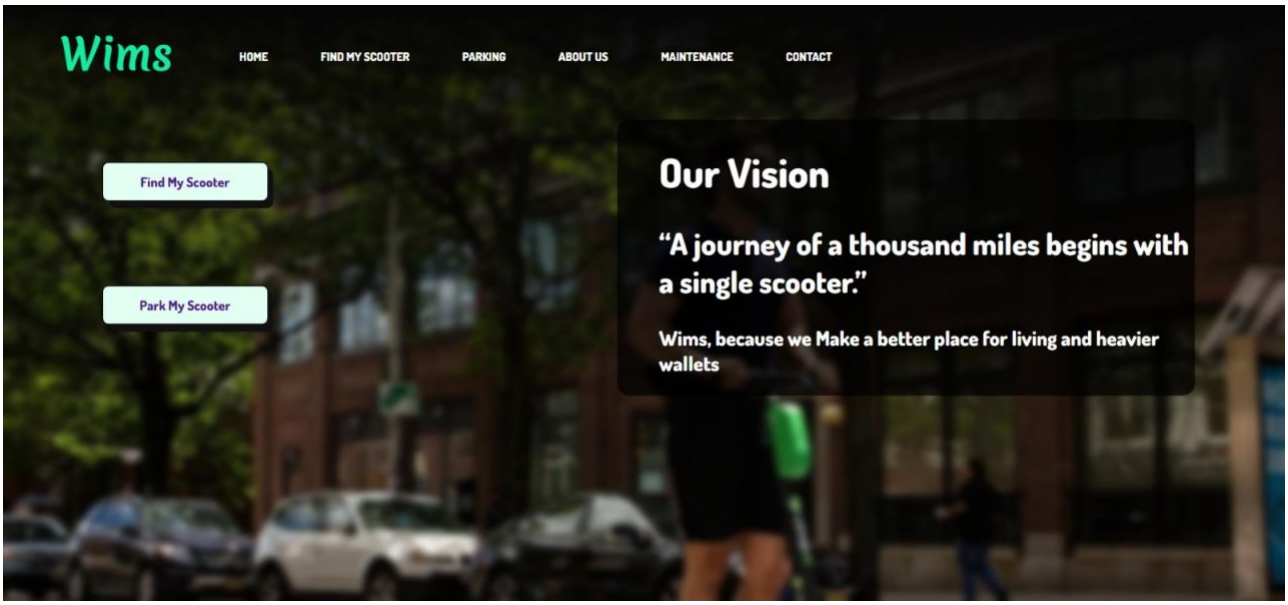
פרויקט קבוצתי חלק ג

Wims



מגישים:

עמית מוזר – 208347286 , רועי בורג – 313322422 , יוסף זנבר – 316495597 , אמיר אפוד - 206251316

רקע
חלק זה סיכם עבורנו את הקורס וכלל שימוש בכלל התכנים הנלמדים- מימוש דפי HTML מורכבים, שימוש בJS, api עם google maps, חיבור לבסיסי נתונים ושימוש במחלקות ושאילתות תוך יצירת סביבת עבודה בflask. כמו כן סביבה זו סודרה בתור בלופרינטס על כל עקרונותיו.
ניסינו ליצור ממשק שייתן ערך עבור המשתמש תוך מתן דגש על עבודה שתכלול תכנים מורכבים המממשים את הנלמד בכיתה.

מטרת האתר הינה להוות פלטפורמה המאפשרת צפייה בכל השירותים הקיימים של הקורקינטים החשמליים בהתאם למיקום גיאוגרפי בכך לחסוך מעבר מיותר בין האפליקציות השונות. האתר מאפשר למשתמש להזין עיר ולצפות בקורקינטים השונים ובחניות שיש בעיר אשר נבחרה.
בנוסף באתר יש ממשק מיוחד עבור טכנאים שבו הם יכולים להזין, לעדכן ולמחוק רשומות וכן לצפות בפעילויות חריגות של הקורקינטים.

יצירת בסיס הנתונים

יצרנו סכמת בסיס נתונים בשם Team8 והגדרנו מספר טבלאות חדשות בבסיס הנתונים.

טבלת Parkings – בטבלה זו כל רשומה מייצגת חניה וכל חניה כוללת עיר, קווי רוחב וגובה , והאם החניה פנויה או לא.

טבלת Scooters בטבלה זו כל רשומה מייצגת קורקינט וכל קורקינט מצוין לגביו מספר סידורי , החברה אליה הוא שייך, עיר , קווי גובה ורוחב, האם יש קסדה ומצב סוללה.

טבלת Technicians – טבלה זו מחזיקה רשומות לגבי הטכנאים של החברה, מיועד עבור דף הmaintenance על מנת שהטכנאים יוכלו להוסיף, לעדכן ולמחוק קורקינטים. טבלה זו מחזיקה את שם המשתמש והסיסמא של הטכנאי בעת התחברות לדף.

טבלת Contacts – טבלה זו צוברת את הפניות המצטברות מהלקוחות וכוללת מספר סידורי של הפניה, שם , מייל ותוכן.

יצירת מחלקות
בחלק זה בנינו מחלקות מקשרות בין האתר לבין בסיס הנתונים על מנת שהשאילתות לא יגיעו באופן ישיר אל מחלקת DB_Manager אלא רק דרך המחלקות המתווכות.
המחלקות שבנינו הינן מחלקת Scooters, Parkings, Technicians, Contact_Details. במחלקות השונות ישנם בנאים ושיטות. שיטות אלו מאפשרות התקשרות לבסיס הנתונים ושיטות נוספות.

שאילתות לדוגמה –

מחלקת **Technicians**

שאילתת עזר לחישוב מרחק עבור זיהוי קורקינט גנוב

```
def getParkingVsScooters(self):
    query = 'select s.scooter_id, s.longitude, s.latitude, s.firm_name, s.city, max(p.longitude) as pLat \
            ,max(p.latitude) as pLon \
            from scooters as s join parkings as p on s.city = p.city \
            where s.battery_level > 0 \
            group by s.scooter_id \
            order by s.scooter_id'
    scooters = self.db.fetch(query)
    return json.dumps(scooters)
```

מחלקת **Scooters**

יצירת קורקינט חדש בבסיס הנתונים -

```
def insertScooter(self):
    query = "INSERT INTO scooters(scooter_id,battery_level, longitude, latitude, firm_name, helmet, city)  \
            VALUES ('%s','%s','%s','%s','%s','%s','%s')" \
            %(self.scooter_id, self.battery_level, self.longitude, self.latitude, self.firm_name,
            self.helmet, self.city)
    self.db.commit(query)
```

שאילתות מחיקה ועדכון של קורקינט בבסיס הנתונים -

```
def deleteScooter(self):
    query = "DELETE FROM scooters WHERE scooter_id='%s';" % self.scooter_id
    self.db.commit(query)

def updateScooter(self):
    query = "UPDATE scooters SET firm_name = '%s', helmet = '%s'WHERE scooter_id='%s';" \
            % (self.firm_name, self.helmet, self.scooter_id)
    self.db.commit(query)
```

שאילתה המחזירה קורקינטים הנמצאים על פי עיר -

```
def get_scooters_city(self,city):
    query = "select longitude, latitude,firm_name from team8.scooters where city='%s';" % city
    scooters = self.db.fetch(query)
    return scooters
```

Contacts – שאילתה המכניסה לבסיס הנתונים רשומה חדשה כאשר התבצעה.

```
def insertMessage(self):
    query = "INSERT INTO contact(name,email, message) VALUES ('%s','%s','%s')" \
            % (self.name, self.email, self.message)
    self.db.commit(query)
```

תוכן דינמי

לכל אורך בניית האתר הקפדנו על תוכן דימני המשתנה בהתאם לבסיס הנתונים בכל רגע ומציג למשתמש את התוכן העדכני ביותר.
בדף החניות והקורקינטים המיקומים אשר מוצגים במפה על פי המיקומים העדכניים שבבסיס הנתונים, בנוסף משתנים על פי הקלט מהמשתמש.
בדף הטכנאים מוצגים הקורקינטים כאשר כל עדכון, הכנסה ומחיקה יגררו שינוי של הרשימה המוצגת.

מימוש הטפסים

עמוד Contact Us – הטופס בעמוד זה נשלח לInsert_Details, פונקציה זו מתקשרת עם המחלקה המתווכת של contact us ומכניסה רשומות לטבלת contact.

עמוד Maintenance – בכניסה לעמוד זה אנו מבקשים מהטכנאי להתחבר על פי שם המשתמש והסיסמא שלו, הטופס בודק את תקינות שם המשתמש והסיסמא מהטבלה הרלוונטית ומעביר את הטכנאי לעמוד המיועד.
תחילה, כשיכנס הטכנאי לדף נציג לו טבלה של כל הקורקינטים והשדות הרלוונטיים על מנת לבצע בהם פעולות. בעמוד זה הטכנאי יכול להוסיף רשומה של קורקינט, לעדכן רשומה ולמחוק אותה. בכל פעולה שלו יטען הדף מחדש ויציג לו את הטבלה המעודכנת לאחר השינויים שערך.
לטכנאי יש אפשרות לבצע מעקב על קורקינטים ולבדוק היתכנות לכך שקורקינט נגנב. בעת לחיצה על כפתור look for scooters out of range נשלחת קריאה לשיטה במחלקת Technicians, ושיטה זו מבצעת שאילתה ומחברת בין scooters לparkings על בסיס העיר. השאילתה מחזירה עבור כל קורקינט את השדות הרלוונטים של הקורקינט ואת הקואורדינטות של החניה הרחוקה ביותר ממנו. לאחר מכן מתבצע חישוב שמחזיר בקילומטרים את המרחק בין שתי קואורדינטות אלו. קורקינטים שנמצאים מחוץ לטווח המוגדר לעיר (בלומר, ככל הנראה הקורקינט נגנב ונלקח אל מחוץ לאזור הרצוי) יוצגו על המסך. לדוגמה, יתווסף קורקינט שנמצא בנתניה אך העיר אליו משויך היא תל אביב, וזאת כמובן בתוספת המרחק שנמצא מאזור זה.
בנוסף, בדף זה ממומש עיקרון נוסף אשר למדנו בביתה והוא Session, הטכנאי יישאר מחובר עד אשר ילחץ על Logout, לחיצה על כפתור זה תנתק את הטכנאי ותנקה את הסשן.

תכנים נוספים רלוונטים

עמוד findMy –
בעמוד זה היוזר מתבקש לבחור עיר מתוך רשימת גלילה, לאחר בחירת העיר אנו מציגים לו את הקורקינטים מטבלת Scooters בבסיס הנתונים על המפה בעזרת API עם גוגל מפות. היוזר יכול להזין מרחק בקילומטרים שהוא מתכנן לנסוע ולקבל טבלת השוואה בין החברות השונות על פי תעריף פתיחה תעריף לדקת נסיעה.
הצגת הקורקינטים במפה מתבצעת על ידי קובץ ה-js, בתוך הקובץ מתבצעת קריאה לפונקציית fetch אשר שולחת כפרמטר את העיר אותה המשתמש בחר ומחזירה את תוצאת השאילתה מהטבלה. בשלב הבא הפונקציה יוצרת את המפה וממקמת את הקורקינטים על המפה בהתאם לקווי אורך ורוחב עם הicon המתאים לפי סוג קורקינט.

עמוד Parking –
בעמוד זה היוזר מתבקש לבחור עיר מתוך רשימת גלילה, לאחר בחירת העיר אנו מציגים לו את החניות מטבלת parkings בבסיס הנתונים על המפה בעזרת API עם גוגל מפות.

הצגת החניות במפה מתבצעת על ידי קובץ ה-js, בתוך הקובץ מתבצעת קריאה לפונקציית fetch אשר שולחת כפרמטר את העיר אותה המשתמש בחר ומחזירה את תוצאת השאילתה מהטבלה. בשלב הבא הפונקציה יוצרת את המפה וממקמת את החניות על המפה בהתאם לקווי אורך ורוחב עם icon מתאים.

<u>הנחות</u>

- כל הקורקינטים בבסיס הנתונים נמצאים בבאר שבע או בתל אביב לצורך ההדגמה אך פעילות החברה גם בירושלים וב-LA.
- טכנאים יכולים להוסיף רק קורקינטים חדשים ולא חניות חדשות, חניות מעודכנות ישירות בבסיס הנתונים של החברה לאחר עדכון מפת החניות של העירייה.