

DOKUZ EYLÜL ÜNİVERSİTESİ
YÖNETİM BİLİŞİM SİSTEMLERİ BÖLÜMÜ
BİLİŞİM SİSTEMLERİ ANALİZİ VE TASARIMI PROJE RAPORU

RESTORAN YÖNETİM PANELİ

Ata AVCI 2019469117
Mert YÜCEL 2020469069
Berat BAYRAKTAR 2021469007
Berkay AKAN 2021469109

Prof. Dr. YILMAZ GÖKŞEN

İZMİR 2024

ÖZET

TAB GIDA bünyesinde olan FEBS GIDA BURGER KING restoranı ile anlaşarak onların bir sorununa bilişim sistemleriyle yardımcı olmak istedik. İşletme ile yapılan görüşmemizde işletmenin günlük raporlar anlamında bir problemin olmadığını fakat uzun vadede raporlamalarda sıkıntı yaşadıklarını belirttiler. Elleriinde restoranın dört yıllık verileri mevcuttu fakat veriler dağınık ve düzensizdi. Bunların düzenlenmesi ve raporlanması için çalışmalarda bulunduk. İlk hedefimiz restoranın isteklerini karşılamaktı. Bunun için çeşitli grafiklerle ve istekleri doğrultusunda tablolar hazırladık. Elde ettiğimiz verileri ay bazlı, yıl bazlı ve günlük bazlı olarak raporladık. Devam eden süreçte personel yönetimi ve ihtiyacı için personel verilerini çeşitli formüllerle ihtiyacı kadar personel mantığıyla bir sayfa tasarladık. Fazlasıyla işletmenin taleplerini yerine getirdik. Proje sonunda zincir restoranın nasıl işlediğini ve temel ihtiyaçlarını görmüş olduk. İşletmeyle görüştüktan sonra işleyişi pozitif yönde etkileyen projemizle birlikte işletme daha net kararlar alabilir hale geldi. Aynı zamanda enflasyonun yükselmesiyle işletmelerin başarı ölçütü olarak ciroyu değil misafir sayısını baz aldıklarını öğrenmiş olduk.

1)GİRİŞ

Projeye başlarken amacımız zincir restoranların nerelerde problem yaşadıklarını görmek ve bu işletmelere bilişim yardımıyla problemin çözümünü bulmaktı. FEBS GIDA ile görüşmeye gittiğimizde ellerinde yaklaşık olarak dört yıllık fiş ve ciro verileri vardı fakat Excel de tutulmasına rağmen bazı günlerin cirosu eksikti ya da paket ve restoran cirosu olarak ayırlamamıştı. Tahminlemeler ve raporlar oluşturulamıyordu çünkü dağınık ve düzenlenmemiş veriler vardı. İşletme ana merkezinden sadece günlük kasa raporlarını ve fiş sayılarını alabiliyordu. Aynı zamanda TAB GIDA'YA %17 oranda cirodan pay vermek zorundaydı fakat yıllık ve aylık verilerini düzenleyemediği için bunun kıyaslamasını tam olarak yapamıyordu. Bu problemlere bilişim sistemleri ile çare bulabileceğimizi anladık ve bu projeyle işletmenin sorunlarını çeşitli raporlamalarla ve grafiklerle çözdük.

2)YÖNTEM

Yaptığımız projede şelale yöntemini kullandık çünkü yapılması gerekenleri sırayla aşama aşama hallettik. İşletmenin problemlerine çözüm bulmak için web tabanlı bir program yazmaya karar verdik. Projeye başlarken ilk önce işletmenin verilerini aldık Excel üzerinde düzenlemeler yapıldıktan bu verilerin analiziz yapıldı ve bu verilerle ilişkisel bir veri tabanı kurmaya karar verdik. Özellikle ilişkisel veri tabanını tercih etmemizin sebebi her yılın, ayın, günün tarih üzerinde birbiriyle bağlı olmasıdır. Veri tabanı için MySQL kullandık. Veri tabanından istenilen problemlerin çözümü için talepler üzerine sorguları yazdık. Daha sonra bilgisayarımızı sunucu haline getirmek için Node.js kullandık. Node.js ile server kurulumunu yaptık ve veri tabanımızdan elde ettiğimiz sonuçları grafiklere ve tablolara aktardık. Kullanıcının rahat etkileşimi için basit ve dinamik bir tasarım kullandık. Web tasarımı için html ve css kullandık. Kullanıcı istediği iki tarih arasında ciro ve fiş bilgilerini görebilir, günlük ciro ve fiş bilgilerini girebilir, aylık ve günlük ciro, fiş ortalamalarını görebilir, personel bilgilerini görebilir şekilde bir uygulama yazıldı. Gün sonunda paket ve restoran cirosunu ayrı ayrı görebilir ve bunun yanında fiş sayısının paket ve restoran ayrımı yapılmıştır. Bunun sayesinde başarı ölçütü olarak görülen fiş sayısının hangi tarafta fazla olduğunu görebilir. Bu işlem aylık, yıllık ve günlük olmak üzere tasarlanmıştır. Uygulama sadece yatırımcıya özel olduğu için tek bir user_password etkileşimi yaptık. Bunların tamamını SQL üzerinde yaptığımız sorguları çeşitli chartlara entegre ederek oluşturduk. Kullandığımız teknolojiler MySQL, HTML, CSS, Node.js ve JavaScript olarak sıralayabiliriz. SQL ile veri analizi yapıldı, HTML ve CSS ile sayfa tasarımı yapıldı ve Node.js ile bilgisayarımızı sunucu haline getirdik.

3)UYGULAMA

Şekil 1



Bu uygulamamızın ana sayfasıdır. Burada restoran tarihçesi ve restoranın mesai saatlerini görebiliriz. Sol tarafta ise oluşturulan raporların sekmeleri mevcuttur.

Şekil 2

gg.aa.yyyy

Restoran ID:

Restoran Ciro:

Paket Ciro:

Fiş Paket:

Fiş Restoran:

Veri Ekle

Şekil 3

```
app.post('/veri_ekle_iki_tablo', (req, res) => {
  const { tarih, restoran_id, restoran_ciro, paket_ciro, fis_paket, fis_restoran } = req.body;

  // Tüm gerekli alanların dolu olup olmadığını kontrol et
  if (!tarih || !restoran_id || !restoran_ciro || !paket_ciro || !fis_paket || !fis_restoran) {
    return res.status(400).send('Tüm alanları doldurun.');
```

```
  } else {
    // İlk olarak gunluk_ciro tablosuna veri ekleme
    const ciroSql = "INSERT INTO `gunluk_ciro` (`tarih`, `restoran_id`, `restoran_ciro`, `paket_ciro`) VALUES (?, ?, ?, ?)";
    const ciroValues = [tarih, restoran_id, restoran_ciro, paket_ciro];

    db.query(ciroSql, ciroValues, (ciroErr, ciroResult) => {
      if (ciroErr) {
        return res.status(500).send('Ciro tablosuna veri eklenirken bir hata oluştu');
      }

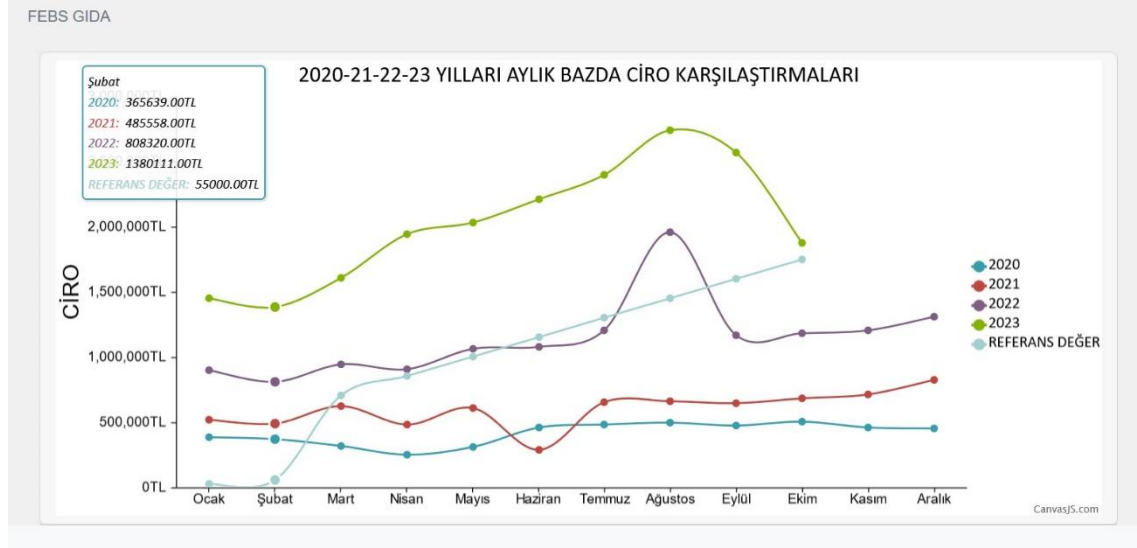
      // Ardından gunluk_fis tablosuna veri ekleme
      const fisSql = "INSERT INTO `gunluk_fis` (`tarih`, `restoran_id`, `fis_paket`, `fis_restoran`) VALUES (?, ?, ?, ?)";
      const fisValues = [tarih, restoran_id, fis_paket, fis_restoran];

      db.query(fisSql, fisValues, (fisErr, fisResult) => {
        if (fisErr) {
          return res.status(500).send('Fis tablosuna veri eklenirken bir hata oluştu');
        }

        res.status(200).send('Her iki tabloya da veri başarıyla eklendi');
      });
    });
  }
});
```

Programın bu kısmında restoranın gün sonunda paket, restoran cirosunu ve paket, restoran fiş sayısını girerek uygulamanın güncel olması sağlanacaktır. Girilen verilere göre istatistikler değişecektir. Veri tabanı aynı zamanda güncellenecektir.

Şekil 4



Bu şekilde restoran son dört yıllık toplam ve aylık bazlı ciro olmak üzere kıyaslama yapabilir ve istediği yılı kaldırıp ekleyebilir. Bunun sayesinde geçmişte yapılan işi görebilir.

Şekil 5

Personel Fis Bilgileri

Başlangıç Tarihi

01.01.2023

Bitiş Tarihi

01.02.2023

Verileri Getir

Tarih: 2023-01-31T21:00:00.000Z

Fis Paket: 13

Fis Restoran: 8

Tarih: 2023-01-30T21:00:00.000Z

Fis Paket: 13

Fis Restoran: 7

Şekil 6

```
app.get('/personel', (req, res) => {
  const baslangicTarihi = req.query.baslangic_tarihi;
  const bitisTarihi = req.query.bitis_tarihi;

  // SQL sorgusu
  const sqlQuery = `
    SELECT
      ROUND((gunluk_fis.fis_paket) / personel.personel_sayi) AS fis_paket,
      ROUND((gunluk_fis.fis_restoran) / personel.personel_sayi) AS fis_restoran,
      gunluk_fis.tarih
    FROM
      gunluk_fis, restoran, personel
    WHERE
      gunluk_fis.restoran_id = restoran.restoran_id
      AND personel.restoran_id = restoran.restoran_id
      AND gunluk_fis.tarih BETWEEN ? AND ?
  `;

  // Veritabanı sorgusunu çalıştırma
  db.query(sqlQuery, [baslangicTarihi, bitisTarihi], (err, results) => {
    if (err) {
      res.status(500).send(err);
    } else {
      res.json(results);
    }
  });
});
```

Burada personel başına düzen fiş sayısı bulunmaktadır. Seçilen iki tarih arasında paket fiş sayısı ve restoran fiş sayısının bir personele düşen sayısını görebiliyoruz. İşletmenin talebine göre personel başı fiş oranını sabitlersek eksik veya fazla personelle çalıştığını görebilir. Buna göre personel alıp veya çıkartabilir.

Şekil 7

Tarih Aralığına Göre Günlük Ciro		
<div>Başlangıç Tarihi: 01.01.2023 Bitiş Tarihi: 01.02.2023 <button>Sorgula</button></div>		
Tarih	Paket Ciro	Restoran Ciro
2022-12-31T21:00:00.000Z	41716.5	17223.45
2023-01-01T21:00:00.000Z	19582.5	10894.38
2023-01-02T21:00:00.000Z	25856	13803.31
2023-01-03T21:00:00.000Z	25481.5	13648.38
2023-01-04T21:00:00.000Z	28731	13455.27
2023-01-05T21:00:00.000Z	27819	14561.66
2023-01-06T21:00:00.000Z	28243	21005.04
2023-01-07T21:00:00.000Z	33776	11716.11
2023-01-08T21:00:00.000Z	25482	10500.8
2023-01-09T21:00:00.000Z	24049	10923.17
2023-01-10T21:00:00.000Z	24936.49	13204.37
2023-01-11T21:00:00.000Z	27400	17066.23
2023-01-12T21:00:00.000Z	38306.99	18064.03
2023-01-13T21:00:00.000Z	36951.5	26314.03

Şekil 8

```
app.get('/iki-tarih-ciro', (req, res) => {
  const tarih1 = req.query.tarih1; // İstenen tarih aralığı parametreleri
  const tarih2 = req.query.tarih2;

  const query = `SELECT gunluk_ciro.paket_ciro as paket, gunluk_ciro.restoran_ciro as restoran, gunluk_ciro.tarih
  FROM gunluk_ciro
  WHERE gunluk_ciro.tarih BETWEEN ? AND ?`;

  // Veritabanı sorgusunu çalıştırma
  db.query(query, [tarih1, tarih2], (error, results) => {
    if (error) {
      res.status(500).json({ error: 'Veritabanı hatası: ' + error.message });
      throw error;
    }

    res.json(results); // Sorgu sonuçlarını JSON olarak döndürme
  });
});
```

İşletme bu kısımda tercih ettiği iki tarih aralığında günlük paket ve restoran cirolarını görebilir. İşletmenin talebi üzerine paket ve restoran cirolarını ayrı olarak sınıflandırılmıştır.

Şekil9

Tarih Aralığına Göre Günlük Fiş

Başlangıç Tarihi:

Bitiş Tarihi:

[Sorgula](#)

Tarih	Paket Fiş	Restoran Fiş
2022-12-31T21:00:00.000Z	140	293
2023-01-01T21:00:00.000Z	168	128
2023-01-02T21:00:00.000Z	196	149
2023-01-03T21:00:00.000Z	183	139
2023-01-04T21:00:00.000Z	192	158
2023-01-05T21:00:00.000Z	205	167
2023-01-06T21:00:00.000Z	188	203
2023-01-07T21:00:00.000Z	240	177
2023-01-08T21:00:00.000Z	177	119
2023-01-09T21:00:00.000Z	171	104

Şekil 10

```
app.get('/iki-tarih-fis', (req, res) => {
  const tarih1 = req.query.tarih1; // İstenen tarih aralığı parametreleri
  const tarih2 = req.query.tarih2;

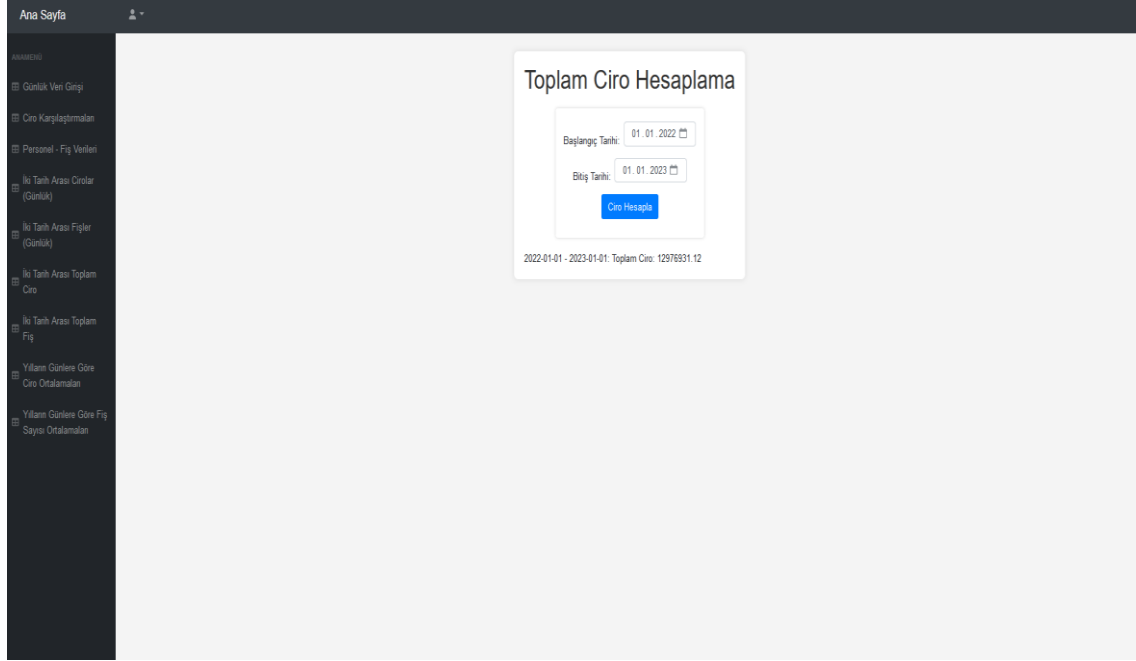
  const query = `SELECT gunluk_fis.fis_paket as paket,gunluk_fis.fis_restoran as restoran,gunluk_fis.tarih
  FROM gunluk_fis
  WHERE gunluk_fis.tarih BETWEEN ? AND ?`;

  // Veritabanı sorgusunu çalıştırma
  db.query(query, [tarih1, tarih2], (error, results) => {
    if (error) {
      res.status(500).json({ error: 'Veritabanı hatası: ' + error.message });
      throw error;
    }

    res.json(results); // Sorgu sonuçlarını JSON olarak döndürme
  });
});
```

İşletme bu kısımda tercih ettiği iki tarih aralığında günlük paket ve restoran fiş sayılarını görebiliriz. İşletmenin talebi üzerine paket ve restoran fiş sayıları ayrı olarak sınıflandırılmıştır.

Şekil 11



Şekil 12

```
app.get('/ciro', (req, res) => {
  const tarih1 = req.query.tarih1; // İstenen tarih aralığı parametreleri
  const tarih2 = req.query.tarih2;

  const query = `SELECT SUM(gunluk_ciro.paket_ciro + gunluk_ciro.restoran_ciro) as toplam_ciro
  FROM gunluk_ciro
  WHERE gunluk_ciro.tarih BETWEEN ? AND ?`;

  // Veritabanı sorgusunu çalıştırma
  db.query(query, [tarih1, tarih2], (error, results) => {
    if (error) {
      res.status(500).json({ error: 'Veritabanı hatası: ' + error.message });
      throw error;
    }

    res.json({ toplam_ciro: results[0].toplam_ciro });
  });
});
```


Uygulamanın bu kısmında yatırımcı seçtiği iki tarih arasında yapmış olduğu toplam cirosunu görebilir. TAB GIDA'YA ödenen para için tasarlanmıştır.

Şekil 13

Tarih Aralığı Seç

Başlangıç Tarihi:
01.01.2022

Bitiş Tarihi:
01.01.2023

Toplamı Göster

Başlangıç Tarihi	Bitiş Tarihi	Toplam Fiş
2022-01-01	2023-01-01	152846

Copyright © Your Website 2023 Privacy Policy Terms & Conditions

Şekil 14

```
app.get('/iki-tarih-toplam-fis', (req, res) => {
  const tarih1 = req.query.tarih1;
  const tarih2 = req.query.tarih2;

  if (!tarih1 || !tarih2) {
    return res.status(400).json({ error: 'Tarih değerleri eksik veya geçersiz.' });
  }

  const query = `SELECT SUM(gunluk_fis.fis_paket+gunluk_fis.fis_restoran)
FROM gunluk_fis
WHERE gunluk_fis.tarih BETWEEN ? AND ?`;

  db.query(query, [tarih1, tarih2], (error, results) => {
    if (error) {
      res.status(500).json({ error: 'Veritabanı hatası: ' + error.message });
      throw error;
    }

    if (results && results.length > 0 && results[0]['SUM(gunluk_fis.fis_paket+gunluk_fis.fis_restoran)']) {
      res.json({ toplam_fis: results[0]['SUM(gunluk_fis.fis_paket+gunluk_fis.fis_restoran)'] });
    } else {
      res.json({ toplam_fis: 0 });
    }
  });
});
```

Uygulamanın bu kısmında yatırımcı seçtiği iki tarih arasında yapmış olduğu toplam fiş sayılarını görebilir. Fiş sayısında belli tarihleri kıyaslamak için tasarlanmıştır.

Şekil 15



Şekil 16

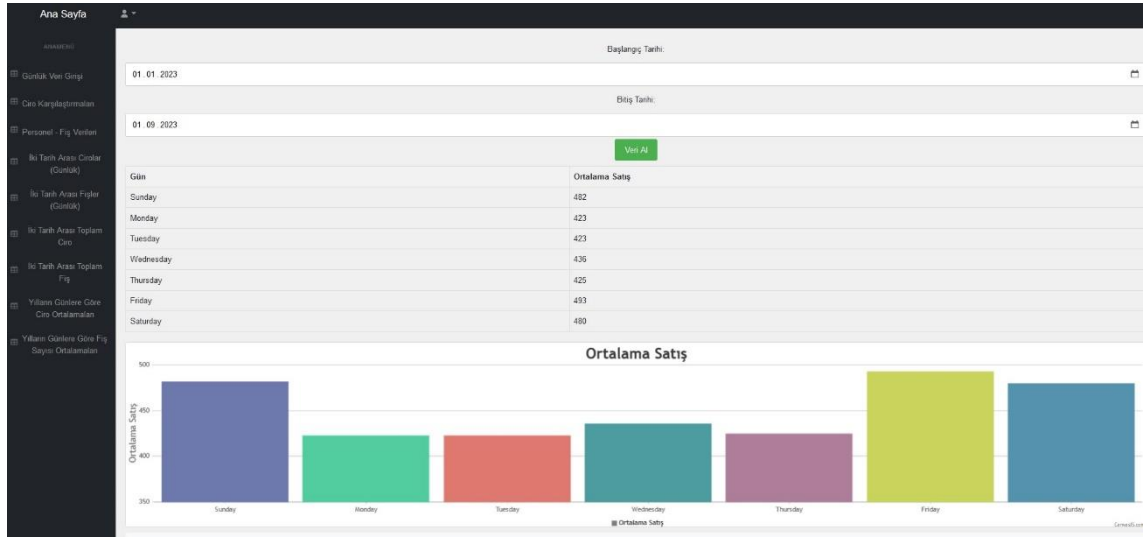
```
app.get('/gunlukciro/:year', (req, res) => {
  const year = req.params.year;

  const query = `
    SELECT
      YEAR(tarih) AS yil,
      MONTHNAME(tarih) AS ay,
      ROUND(AVG(paket_ciro + restoran_ciro), 2) AS ortalama_ciro
    FROM
      gunluk_ciro
    WHERE
      YEAR(tarih) = ?
    GROUP BY
      yil, MONTH(tarih)
    ORDER BY
      yil, MONTH(tarih);
  `;

  db.query(query, [year], (error, results) => {
    if (error) {
      console.error('Error executing query:', error);
      res.status(500).json({ error: 'Error fetching data' });
      return;
    }
    res.json(results);
  });
});
```

Uygulamanın bu kısmını yıllık ciro raporu olarak görebiliriz. Ekranın üst kısmında yılı seçiyoruz daha sonra bize yıllık toplam ciro yanında bir de seçilen yılın aylık ciro ortalamasını hem tablo olarak hem de grafik olarak vermektedir. Burada aylık bazlı kıyaslamalar yapılabilir hangi ayların daha fazla ciro yaptığını görebiliriz.

Şekil 17



Şekil 18

```
app.get('/gunluk_ortalama_satis', (req, res) => {
  const { startDate, endDate } = req.query;

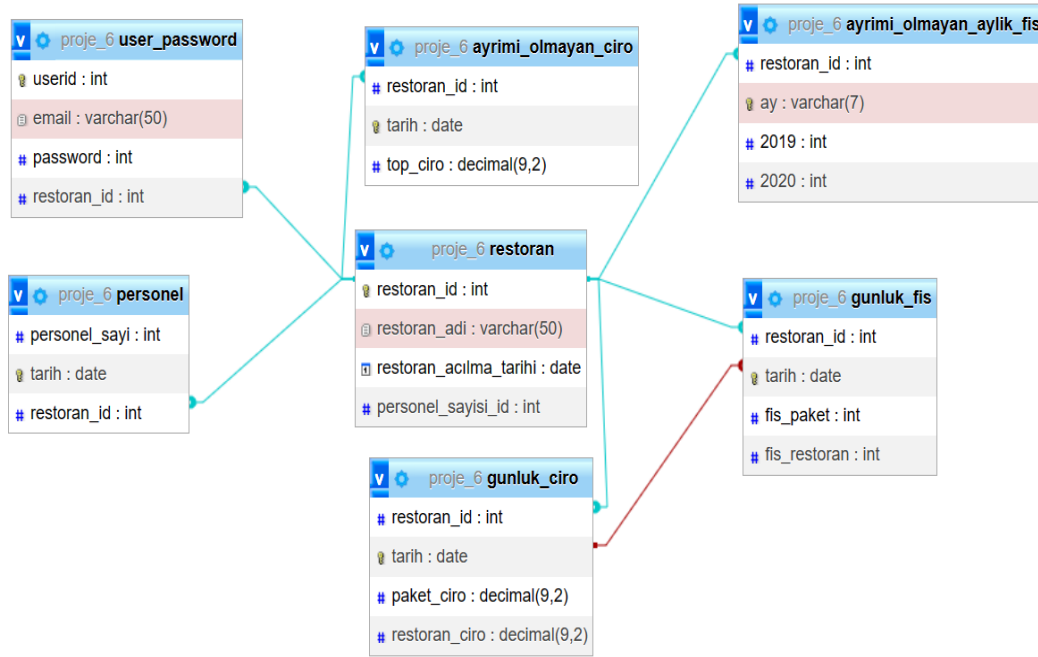
  if (!startDate || !endDate) {
    return res.status(400).json({ error: 'Başlangıç ve bitiş tarihleri gereklidir.' });
  }

  db.query(
    `SELECT
      DAYNAME(tarih) AS gun_adi,
      ROUND(AVG(fis_paket + fis_restoran)) AS ortalama_satis
    FROM
      gunluk_fis
    WHERE
      gunluk_fis.tarih BETWEEN ? AND ?
    GROUP BY
      DAYNAME(tarih)
    ORDER BY
      FIELD(DAYNAME(tarih), 'Pazartesi', 'Salı', 'Çarşamba', 'Perşembe', 'Cuma', 'Cumartesi', 'Pazar');`,
    [startDate, endDate],
    (error, results, fields) => {
      if (error) {
        res.status(500).json({ error: 'Veritabanı hatası' });
        throw error;
      }

      res.json({ results });
    }
  );
});
```

Uygulamanın son bölümünde ile yıllık olarak fişlerin gün bazlı ortalamalarını görebiliriz. Örnek verecek olursak yılın haftalık günlerinde oluşan ortalama yoğunluğu ve fiş sayılarını görebiliriz. Bu bilgileri hem tablo hem de grafik olarak verdik. Bu yapmamızdaki en büyük amaç personel çalıştırırken ihtiyacı kadar çalıştırabilmektir. Yoğun günlerde daha fazla çalıştırırken yoğunluğun düşük olduğu günlerde az personel çalıştırarak kar kısmını arttırmasıdır. Yoğunluklar günlük ortalama olarak belirtilmiştir.

Şekil 19



Burada veri tabanımızı görebiliriz fişler ve ciroları ayrı olarak işlenmiştir. Ayrımı olmayan tablolar eksik ve tam olarak tamamlanmamış tablolardır ya ciro kısmı eksik yada tarih kısımları eksik olan veriler çoğunlukla karşılaştığımız problemleri kısımlardır.

Şekil 20

The image shows a login form with the following elements:

- Header:** Login
- Email field:** A text input field containing 'deneme@gmail.com'.
- Password field:** A text input field with masked characters (dots).
- Login button:** A blue button labeled 'Login'.

Şekil 21

```
app.post("/auth", function (req, res) {
  const password = req.body.password;

  db.query("SELECT * FROM user_password WHERE password=?", [password], (err, results) => {
    if (err) throw err;

    if (results.length > 0) {
      res.status(200).sendFile(path.join("C:/Users/ataav/Downloads/SONWEB/SONWEB", "dashboard.html"));
    } else {
      res.status(200).sendFile(path.join("C:/Users/ataav/Downloads/SONWEB/SONWEB", "401.html"));
    }

    // Note: No need to call res.end() here, res.sendFile and res.send already end the response.
  });
});
```

Burada login sayfamızı görebiliriz. Tek bir yatırımcıya özel olarak tasarlanmıştır. O nedenle yönetici girişi tasarımı yoktur.

4)SONUÇ

Projemizi bitirdiğimizde işletmenin bütün taleplerini yerine getirdik. İşletme bu programı restoranında uygulamaya koymaya başlayacaktır. Proje yapım aşamasında veri analizi yapılırken verilerin dağınık olarak tutulması işimizi zorlaştırdı fakat bunun üstesinden geldik ve hedef olarak belirlediğimiz bütün analizleri yaptık. Backend kısmında zor anlar yaşadık çünkü Node.js daha önce hiç kullanmadık ama kendimizi geliştirmek adına ve daha güncel bir program kullanarak aynı zamanda kendimize yatırım yaptık. Arayüz kısmında bir problem yaşanmadı. Bu projeyi yaptıktan sonra zincir restoranların problemlerini daha yakından gördük ve yardımcı olduk. İşin işleyişini görürken bunun yanında değişen başarı ölçütlerini keşfettik. İşletmenin bu uygulamayı restoranında kullanmaya karar vermesi bizi ayrıca mutlu etmiştir.