Fatura Ödeme Sistemi

3.Grup

Ferdi Koca

Fatih Sezer

Büşra Yalman Beyaz

Yasir Akkaya

Furkan Ermiş

Adem Karagülle

Proje Raporu

Rapor Tarihi	Proje Adı	Hazırlayan
09.10.2023	Fatura Ödeme Sistemi	Ferdi Koca Fatih Sezer Büşra Yalman Beyaz Yasir Akkaya Furkan Ermiş Adem Karagülle

Proje Özeti

Hazırlanmış olan Fatura Ödeme Sistemi; kullanıcıların farklı fatura türleri için fatura eklemelerini, düzenlemelerini, silmelerini ve ödeme işlemlerini gerçekleştirebilecekleri bir web tabanlı uygulamadır. Ayrıca kullanıcıların bakiye bilgilerini yönetmelerini sağlar. Proje, kullanıcı dostu bir arayüz ve işlevsel özelliklerle tasarlanmıştır. Ferdi Koca, Fatih Sezer, Büşra Beyaz, Yasir Akkaya, Furkan Ermiş, Adem Karagülle tarafından 4 Ekim – 9 Ekim 2023 tarihleri arasında geliştirişmiştir. Bu projede daha önceden Adem Karagülle tarafından oluşturulan CSS Kütüphanesinden yararlanılmıştır.

GÖREV	YAPILAN İŞLEMLER
Fatura ve kullanıcı sınıfları oluşturuldu.	Faturaların ve kullanıcıların bilgilerinin tutulması amacıyla bill ve user sınıfları oluşturuldu. User sınıfına addBill ve editBill metodları eklendi.
Pop-up oluşturuldu.	Pop-up ile bakiye ve fatura bölümlerine yönlendiren butonlar eklendi.
InitialState fonksiyonu oluşturuldu.	Sayfa ilk yüklendiğinde hangi elementlerin görünüp görünmeyeceği ayarlandı.
whenLogIn, whenLogOut fonksiyonları oluşturuldu.	Kullanıcı sisteme giriş/çıkış yaptığı esnada hangi elementlerin görünür olup hangilerinin olmayacağı belirlendi.
updateTable fonksiyonu oluşturuldu.	Kullanıcının faturalarına dair ödendi/ödenmedi bilgisi ve hepsi seçenekleri ile görüntülenmesi sağlandı. Düzenleme, ödeme ve silme ikonları her satıra eklendi.
payBill fonksiyonu oluşturuldu.	Kullanıcı tabloda ödeme ikonuna tıkladığı zaman bu fonksiyon çağırılır. Ve kullanıcıya ait bakiye miktarına bağlı ödeme işlemi gerçekleştirilir.
editBill fonksiyonu oluşturuldu.	Kullanıcı tabloda düzenleme ikonuna tıkladığı zaman bu fonksiyon çağırılır. İlgili tablo satırı düzenlenecek şekilde fatura bilgileri ekranında doldurulur. Güncel fatura kullanıcının eski faturası ile değiştirildi.
deleteBill fonksiyonu oluşturuldu.	Kullanıcı tabloda delete ikonuna bastığı zaman ilgili fatura kullanıcının fatura listesinden silinir.
Giriş butonu event'i eklendi.	Kullanıcı giriş yaparken bilgilerine "users" listesinden ulaşıldı.
Kayıt butonu oluşturuldu.	Yeni kullanıcının eklenmesi sağlandı.
Çıkış yap butonu eklendi.	Sistemden çıkış yapılması sağlandı.
Fatura ekle butonu eklendi.	Kullanıcının girdiği fatura bilgileri kullanıcıya ait fatura listesine eklendi. Bu işlem createBill fonksiyonu ile gerçekleştirildi.
updateAmount fonksyionu oluşturuldu.	Kullanıcının güncel bakiyesinin ilgili yerlere eklenmesi sağlandı.
Bakiye ekleme event'i oluşturuldu.	Kullanıcının sisteme istediği miktarda para yükleyebilmesi sağlandı.

Günlere Göre İş Dağılımı

4 Ekim - 6 Ekim: Temel Özelliklerin Geliştirilmesi

- Kullanıcı giriş ve kayıt işlemlerinin tamamlanması.
- Kullanıcıların fatura ekleyebilmesi ve bu faturaları düzenleyebilmesi.
- Kullanıcıların bakiyelerini görüntüleyebilmesi ve bakiye ekleyebilmesi.
- Ödenmiş ve ödenmemiş faturaları filtreleme özelliğinin eklenmesi.

7 Ekim - 8 Ekim: Ek Özelliklerin Eklenmesi

- Fatura ödeme işleminin tamamlanması ve bakiyeden düşme işleminin gerçekleştirilmesi.
- Kullanıcının oturumunu kapatma ve tekrar giriş yapma yeteneğinin eklenmesi.
- Fatura silme işleminin tamamlanması.
- Kullanıcı arabirimine tasarımsal iyileştirmeler eklenmesi.

9 Ekim: Test ve Dökümantasyon

- Uygulamanın hata ayıklama ve test aşamalarının tamamlanması.
- Kodlarla ilişkili yorum satırlarının eklenmesi.
- Uygulamanın görev ve yapılan işlemler tablosu ile dökümantasyonunun tamamlanması.
- Son kontrollerin yapılması ve proje hazır hale getirilmesi.

Kullanıcının fatura bilgilerinin tutulması için bill adlı sınıf oluşturuldu. Bu sınıfta, faturanın tipi, adı, tarihi, ücreti ve ödenip ödenmediği bilgileri tutulur.

```
// Fatura sınıfın oluşturulması
class bill {
    constructor(billType, billName, billDate, billCost, isPaid) {
        this.billType = billType;
        this.billName = billName;
        this.billDate = billDate;
        this.billCost = billCost;
        this.isPaid = isPaid;
    }
}

```

User sınıf tanımlaması; kullanıcılara ait username, password, amount, ve billList gibi özelliklere sahip bir kullanıcı nesnesi oluşturur. Ayrıca bu sınıf, kullanıcılara özgü faturaları eklemek ve fatura listesindeki belirli bir faturayı güncellemek için iki metot içerir.

```
// kullanıcı sınıfının oluşturulması
class user {
    constructor(username, password) {
        this.username = username;
        this.password = password;
        this.amount = 0;
        this.billList = [];
    }

    // kullanıcılara özel fatura ekler.
    addBill(bill) {
        this.billList.push(bill);
    }
    editBill(index, editedbill) { // indexini aldığımız faturanın fatura listesinde güncelleme işlemini yapar
        this.billList.splice(index, 1, editedbill);
    }
}
```

users değişkenine atanmış olan boş array sayesinde uygulamaya kaydolan kişilerin verileri tutulur.

```

let users = []; //
users.push(new user("ferdi", "1234"));

```

Bu fonksiyon, kullanıcının giriş yaptığında sayfada çeşitli görünürlük ayarlarını değiştirerek kullanıcı işlemine uygun davranır ve ilgili sayfaya yönlendirmek için kullanılır.

```
// kullanıcı sisteme giriş yaptığında çağrılan fonksiyon
function whenLogIn() {
    toggleRegister.style.display = "none";
    toggleBill.style.display = "block";
    navButtonGroup.style.display = "flex";
    popUpElement.style.display = "block";
}
```

updateTable fonksiyonu, belirli bir şarta göre kullanıcının faturalarını filtrelemek ve sonuçları bir tabloya yazdırmak için kullanılır.

```
// ödenme durumuna göre faturaların tabloda gösterilmesi sağlanır. kullanıcı ödenmiş, ödenmemiş ve tüm faturaları görüntüleyebilmektedir.
function updateTable(filterCondition = "Hepsi") {
    billTable.innerHTML = "";
    var selectedList;

    if (filterCondition == "Ödenmiş Faturalar") {
        selectedList = currentUser.billList.filter((element) => element.isPaid == "Ödendi");
    } else if (filterCondition == "Ödenmemiş Faturalar") {
        selectedList = currentUser.billList.filter((element) => element.isPaid == "Ödenmedi");
    } else {
        selectedList = currentUser.billList;
    }
    createTableRow(selectedList);
}
```

payBill fonksiyonu, kullanıcının bir faturayı ödemesini sağlar. Yeterli bakiye yoksa uyarı verir; yeterli bakiye varsa, fatura ödenir ve bakiye güncellenir.

```
// ödeme iconuna tıklandığında event listenerin içinde çağrılan fonksiyon
function payBill(index) {
    const bill = currentUser.billList[index];
    if (Number(bill.billCost > currentUser.amount)) {
        alert("Yetersiz Bakiye \nLütfen Yükleme Yapınız!");
    } else {
        currentUser.amount -= Number(bill.billCost);
        bill.isPaid = "Ödendi";
    }
    updateAmount();
    updateTable();
}
```

editBill fonksiyonu, belirtilen indeksteki faturanın bilgilerini alarak form alanlarına doldurur ve faturayı güncellemek için bir düzenleme işlemi gerçekleştirir.

```
// edit iconuna tıklandığında event listenerin içinde çağrılan fonksiyon
function editBill(index) {
    var bill = currentUser.billList[index];
    billType.value = bill.billType;
    billCost.value = bill.billCost;
    billDate.value = bill.billDate;
    billName.value = bill.billName;
    isPaid.checked = bill.isPaid == "Ödendi" ? true : false;
    billBtn.textContent = "Fatura Güncelle";
    billIndex = index;
}
```

deleteBill fonksiyonu, belirtilen indeksteki faturayı kullanıcının fatura listesinden kaldırır ve tabloyu günceller.

```

// delete iconuna tıklandığında event listenerin içinde çağrılan fonksiyon
function deleteBill(index) {
    currentUser.billList.splice(index, 1);
    updateTable();
}

```

createBill fonksiyonu, inputların boş değer girilmemesi için bir if koşuluyla kontrol edilir. Yeni bir fatura oluşturur ve kullanıcının fatura listesine ekler. Eğer gerekli bilgiler eksikse, bir uyarı mesajı gösterir. Oluşturma işlemi tamamlandığında tablo güncellenir ve fatura giriş alanları temizlenir.

```
291
292  function createBill() {
293      if (
294          billCost.value >= 0 &&
295          billCost.value != "" &&
296          billName.value != "" &&
297          billDate.value != ""
298      ) {
299          currentUser.addBill(
300              new bill(
301                  billType.value,
302                  billName.value,
303                  billDate.value,
304                  billCost.value,
305                  isPaid.checked ? "Ödendi" : "Ödenmedi"
306              )
307          );
308      } else {
309          alert("Eksik değer girdiniz.");
310      }
311      updateTable();
312      clearBillInputs();
313  }
314
```

findUser fonksiyonu, verilen username ve password bilgilerini kullanarak users listesinde bir kullanıcı arar. Eğer bu bilgilere sahip bir kullanıcı bulunursa, bu kullanıcı döndürülür.

```
325
326  // kullanıcı bulunduğunda kullanıcıyı döndürür bulamazsa undefined döndürür
327  function findUser(username, password) {
328      return users.find(
329          (user) => user.username == username && user.password == password
330      );
331  }
332
```

updateAmount fonksiyonu, mevcut kullanıcının bakiyesini ve adını sayfadaki belirli bölgelere yansıtarak günceller ve bazı giriş alanlarını temizler.

```
357
358  const updateAmount = () => {
359      wallet.innerText = currentUser.amount;
360      nameOfAmount.innerText = currentUser.username;
361      navbarAmount.innerText = "Bakiye: " + currentUser.amount;
362      inputWallet.value = "";
363      creditCard.value = "";
364  };
365
```