

# Taskify

Çoklu Görev Oluşturucu  
React.js Projesi

Eylm SEYHAN

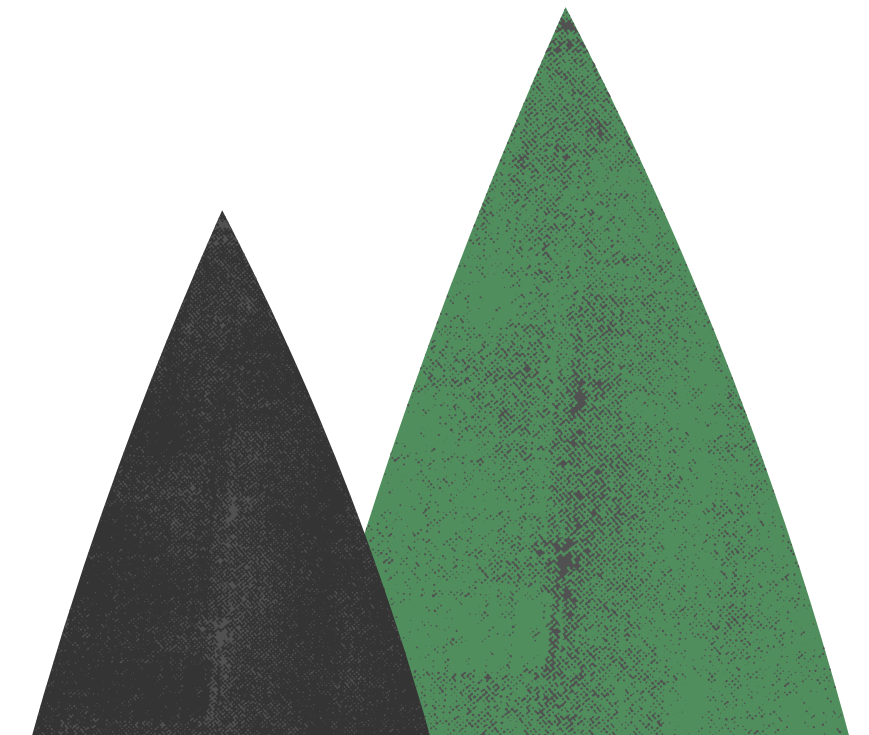




# Dosya Yapısı

```
planner
├── node_modules
├── public
├── src
│   ├── assets
│   ├── components
│   │   ├── common
│   │   │   ├── Navbar
│   │   │   │   ├── LogoutModal.jsx
│   │   │   │   ├── Navbar.css
│   │   │   │   ├── Navbar.jsx
│   │   │   │   ├── NotificationBadge.jsx
│   │   │   │   ├── NotificationModal.jsx
│   │   │   │   ├── ProfileDropdown.jsx
│   │   │   │   ├── ProfileModal.jsx
│   │   │   │   ├── Footer.css
│   │   │   │   ├── Footer.jsx
│   │   │   │   ├── Sidebar.css
│   │   │   │   └── Sidebar.jsx
│   │   │   ├── forms
│   │   │   │   ├── DeleteConfirmationModal.jsx
│   │   │   │   ├── LoginForm.css
│   │   │   │   ├── LoginForm.jsx
│   │   │   │   ├── RegisterForm.jsx
│   │   │   │   ├── SubtaskForm.jsx
│   │   │   │   └── TaskForm.jsx
│   │   │   ├── lists
│   │   │   │   ├── FilteredTaskList.css
│   │   │   │   ├── FilteredTaskList.jsx
│   │   │   │   ├── TaskList.css
│   │   │   │   └── TaskList.jsx
│   │   │   └── tasks
│   │   │       ├── TaskCard.jsx
│   │   │       └── TaskFilter.jsx
│   └── tasks
│       ├── TaskFilter.jsx
│       └── TaskManagement.jsx
├── context
│   └── TaskContext.jsx
├── pages
│   ├── Calendar.css
│   ├── Calendar.jsx
│   ├── Dashboard.css
│   └── Dashboard.jsx
├── Home.css
├── Home.jsx
├── Login.css
├── Login.jsx
├── Register.jsx
├── Schedule.css
├── Schedule.jsx
├── App.css
├── App.jsx
├── App.test.js
├── ErrorBoundary.jsx
├── firebase.js
├── index.css
├── index.jsx
├── logo.svg
├── reportWebVitals.js
├── setupTests.js
├── .gitignore
├── craco.config.js
├── package-lock.json
└── package.json
```

```
planner
├── src
│   ├── components
│   │   └── tasks
│   │       ├── TaskFilter.jsx
│   │       └── TaskManagement.jsx
│   ├── context
│   │   └── TaskContext.jsx
│   ├── pages
│   │   ├── Calendar.css
│   │   ├── Calendar.jsx
│   │   ├── Dashboard.css
│   │   └── Dashboard.jsx
│   ├── Home.css
│   ├── Home.jsx
│   ├── Login.css
│   ├── Login.jsx
│   ├── Register.jsx
│   ├── Schedule.css
│   └── Schedule.jsx
│   ├── App.css
│   ├── App.jsx
│   ├── App.test.js
│   ├── ErrorBoundary.jsx
│   ├── firebase.js
│   ├── index.css
│   ├── index.jsx
│   ├── logo.svg
│   ├── reportWebVitals.js
│   ├── setupTests.js
│   ├── .gitignore
│   ├── craco.config.js
│   ├── package-lock.json
│   └── package.json
```



# Home sayfası



Anasayfa Özellikler Fiyatlandırma İletişim

## Taskify'a Hoş Geldiniz

Taskify, takım görevlerinizi düzenlemenize ve yönetmenize yardımcı olan bir platformdur. Görevlerinizi ekleyin, düzenleyin ve takım üyelerinizle işbirliği yaparak projelerinizi yönetin. İş süreçlerinizi daha verimli hale getirin ve takımınızla birlikte başarıya ulaşın.

[-> Giriş Yap](#)

[-> Kayıt Ol](#)

### Görev Yönetimi

Takımınızın görevlerinizi kolayca ekleyin ve yönetin.

### Takım İşbirliği

Takım üyelerinizle işbirliği yapın ve projeleri birlikte yönetin.

### İlerleme Takibi

Görevlerinizin ilerlemesini kolayca takip edin.

## Kullanıcı Yorumları



Taskify sayesinde tüm görevlerimi kolayca yönetebiliyorum. İşbirliği yapmak çok daha kolay hale geldi.

- İpek Öztürk



Taskify, takım projelerinde düzeni sağlamak için harika bir araç. Herkese tavsiye ederim!

- Ebrar Seda Gündüz

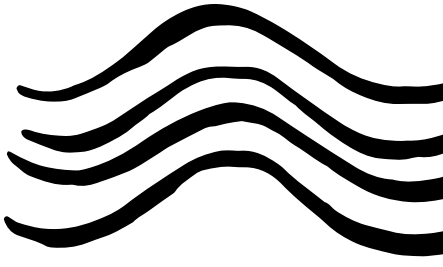


Görevlerimi takip etmek ve tamamlandıklarında işaretlemek hiç bu kadar kolay olmamıştı. Taskify mükemmel!

- Zeynep Helin Aydın

Kütüphaneler ve Veri Tanımları: React, Ant Design bileşenleri, logo ve stil dosyaları içe aktarılır. **testimonials** dizisi ile kullanıcı yorumları tanımlanır.

Yönlendirme: **useNavigate** ile navigasyon işlevi tanımlanır.



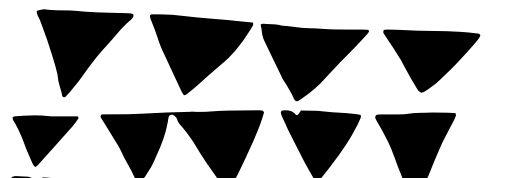
# Login ve LoginForm

`handleLogin` işlevi ile başarılı giriş sonrası yönlendirme yapılır.

- `handleSubmit` , form gönderildiğinde çalışır. Bu işlev:
- Form verilerini alır ve `signInWithEmailAndPassword` ile Firebase Authentication'a gönderir.
  - Giriş başarılı olursa, başarılı mesaj gösterir ve kullanıcıyı dashboard sayfasına yönlendirir.
  - Giriş başarısız olursa, hata mesajı gösterir ve hatayı konsola yazdırır.

Taskify'a giriş yap.

Giriş Yap



# Register ve RegisterForm

Kayıt başarılı olduktan sonra kullanıcıyı dashboard'a yönlendiren **handleRegister** işlevi tanımlanır

**onFinish** işlevi, kullanıcı bilgilerini alır ve Firebase Authentication ile kullanıcı oluşturur (`createUserWithEmailAndPassword`). Kayıt başarılı olursa, kullanıcının bilgileri Firestore veritabanına kaydedilir (`firestore.collection('users').doc(user.email).set`). Bu bilgiler arasında ad, e-posta ve şifre yer alır.

Eğer e-posta zaten kullanılıyorsa, kullanıcıya giriş yapması önerilir. Bu durumda, **signInWithEmailAndPassword** kullanılarak giriş işlemi gerçekleştirilir.

Başarılı veya başarısız giriş işlemleri mesaj olarak bildirilir.

Taskify'a kayıt ol.

Kayıt Ol



# Dashboard



Hoş geldiniz Eylem  

Görevler Görev Filtrele

Tamamlanmış Görevler

0

Bekleyen Görevler

0

Devam Eden Görevler

0

Görev Ekle




Henüz planlanmış bir görev yok

Takvim

selectedMenu, filteredTasks, ve taskStats gibi state'ler useState ile tanımlanır. Görevleri yüklemek için **TaskContext** kullanılır.

useEffect kullanılarak görevler yüklendiğinde **loadTasks** çağrılır. İkinci useEffect bloğunda, yüklenen görevler filtrelenir ve tamamlanmış, bekleyen ve devam eden görev sayıları hesaplanarak **taskStats** state'i güncellenir.

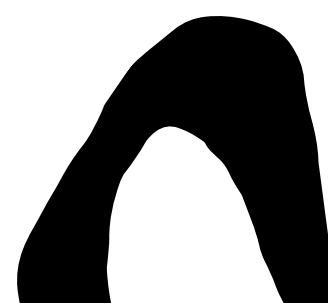
**handleMenuClick** fonksiyonu, kullanıcı menü öğesine tıkladığında seçili menüyü günceller. **handleFilter** fonksiyonu, görevleri seçili atanmış kişilere ve tarih aralığına göre filtreler ve **filteredTasks** state'ini günceller.



"Görevler" menüsü seçildiğinde, görev istatistikleri ve TaskManagement bileşeni gösterilir.

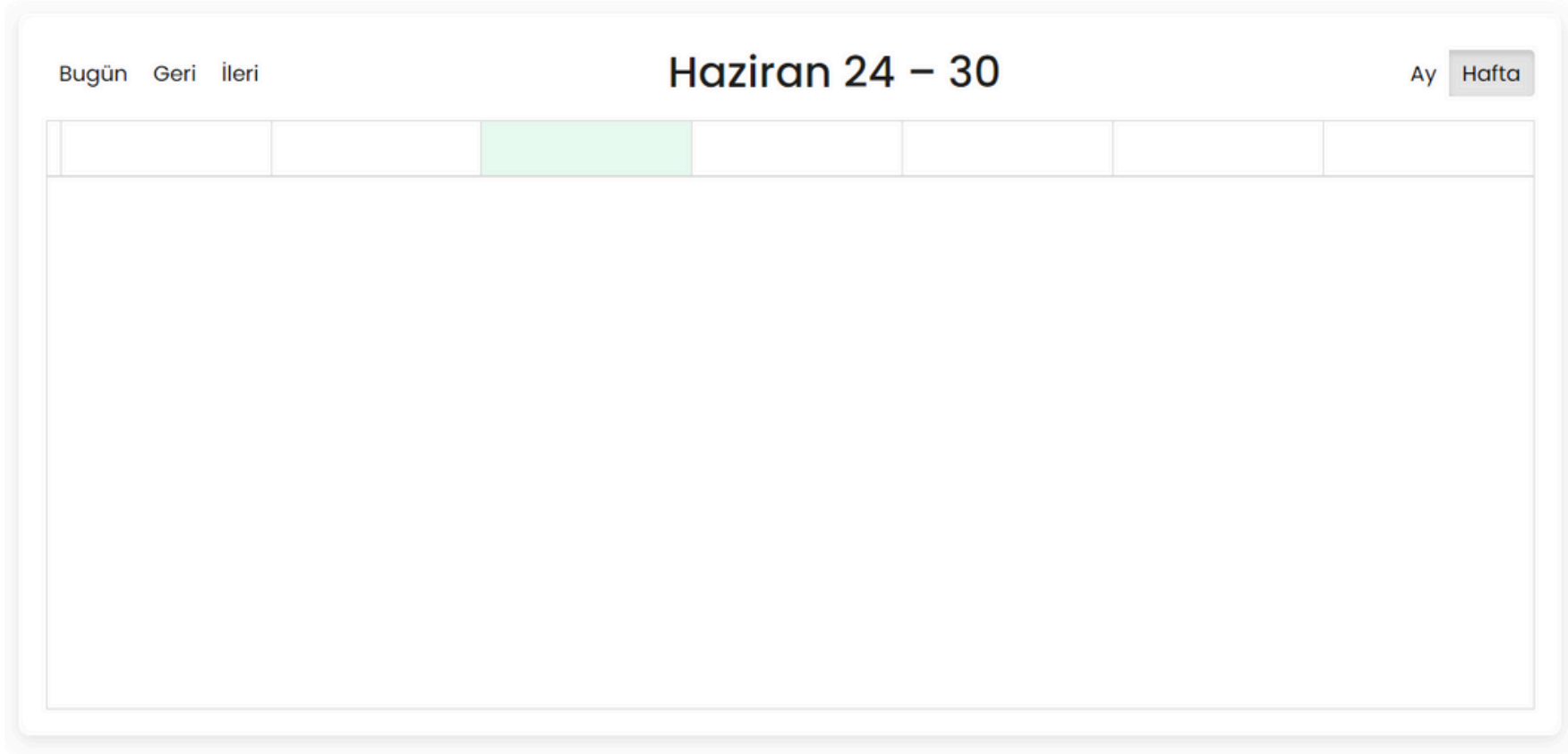
"Görev Filtrele" menüsü seçildiğinde, TaskFilter ve FilteredTaskList bileşenleri gösterilir.

Takvim ve çizelge bölümleri TaskCalendar ve Schedule bileşenleriyle gösterilir.






## Takvim



react-big-calendar  
kütüphanesi  
kullanılarak takvim  
oluşturulur

TaskContext kullanılarak  
görevler yüklenir  
ve useEffect ile sayfa  
yüklendiğinde çağrılır

## Çizelge

Görev Adı	Açıklama	Başlangıç Tarihi	Bitiş Tarihi	Atanan Kişiler
 No data				

TaskContext kullanılarak görevler alınır ve  
getUsers işlevi ile kullanıcılar yüklenir.

useEffect ile kullanıcılar sayfa yüklendiğinde  
getirilir ve  
setUsers ile state'e kaydedilir.

# TaskContext.jsx

## TaskProvider Bileşeni:

tasks, taskNotifications, ve currentUser durumları (state) useState ile tanımlanır. loadTasks işlevi, Firestore'dan görevleri almak ve tasks durumunu güncellemek için tanımlanır.

## Kullanıcı Bilgisini Yükleme:

useEffect kullanılarak bileşen yüklendiğinde fetchCurrentUser işlevi çağrılır. Bu işlev, şu anda oturum açmış kullanıcıyı alır, kullanıcının bilgilerini Firestore'dan getirir ve currentUser durumunu günceller. Kullanıcı bilgileri alındıktan sonra loadTasks işlevi çağrılarak görevler yüklenir.

# SideBar.jsx



**scrollToSection** fonksiyonu yazarak belirli bölümlere yumuşak kaydırma sağladık. Ant Design'ın Menu bileşenleri ve ikonları kullanılarak, "Görevler", "Takvim" ve "Çizelge" menü öğeleri eklenmiştir.

# TaskFilter.jsx

Görevler **Görev Filtrele**

Kişiler:  Tarih Aralığı:  →

Kader  
Eylül  
Eylem  
Ahmet  
Efe

Henüz planlanmış bir görev yok

**RangePicker:** Ant Design'dan tarih aralığı seçici.  
**Option:** Ant Design'dan seçenek bileşeni, Select bileşeni kullandık.

Form gönderildiğinde **handleFinish** fonksiyonu kullanılıyor. Atanan kişileri ve tarihi **onFilter**'a geçiriyor


# TaskManagement.jsx

Görev Ekle ×

\* Görev Adı

Açıklama

\* Tarih Aralığı

Start date → End date 

\* Atanan Kişiler

+ Alt Görev Ekle

\* Durum

İptal Ekle

TaskManagement bileşeni,  
görevleri eklemek, düzenlemek ve  
silme için kullanılıyor