

بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ



دانشگاه فنی حرس فرمای استان خراسان رضوی  
دانشکده فنی شب و نظریه

**پایان نامه دوره: کاردانی**

**رشته: کامپیوتر**

**گرایش: نرم افزار**

**موضوع:**

وب اپلیکیشن یا برنامه مدیریت و برنامه ریزی کارهای روزانه یا اهداف به صورت  
فهرست

**استاد راهنما:**

جناب استاد دکتر محمد عبداللهی

**نام دانشجو:**

محمدرضا آژیده

سال تحصیلی: ۱۴۰۱-۱۴۰۰

## چکیده

زندگی همه آنقدر شلوغ شده که طبیعی است گاهی اوقات بسیاری از کارهایمان را فراموش می‌کنیم. اما به جز این افراد کسانی هم هستند به صورت ذاتی دچار حواس پرتی هستند و افرادی هم می‌خواهند اهداف خود را دنبال کنند. اگر شما هم یکی از این افراد هستید شاید استفاده از برنامه‌های مدیریت کارهای روزانه بد نباشد.

چه برای مدیریت کارهای روزانه و چه برای تهیه یک لیست خرید شما قطعاً به یک برنامه مدیریت و برنامه ریزی نیاز دارید.

تودو یک برنامه کاربردی برای یادآوری و مدیریت کارهای روزانه و اهداف است که بسیار ساده می‌باشد.

کلمات کلیدی:

اپلیکیشن، مدیریت، برنامه ریزی، تودو، یادآوری

## فهرست مطالب

مقدمه ..... ۱

### فصل اول: تعریف مسئله

مقدمه ..... ۲

۱-۱ عنوان و مسئله پروژه ..... ۳

۱-۲ اهداف پروژه ..... ۳

۱-۳ روش انجام پروژه و ابزار ..... ۳

### فصل دوم: نحوه کار با سیستم

مقدمه ..... ۵

۲-۱ صفحه خانه ..... ۶

۲-۲ صفحه ثبت نام ..... ۶

۲-۳ صفحه فهرست کار ها ..... ۷

۲-۴ صفحه اضافه کردن وظیفه ..... ۷

### فصل سوم: پیاده سازی سیستم

مقدمه ..... ۹

۳-۱ جنگو چیست؟ ..... ۱۰

۳-۱-۱ تاریخچه ..... ۱۰

۳-۱-۲ امکانات ..... ۱۰

۳-۲ نحوه اجرا کردن برنامه ..... ۱۱

۳-۳ پوشه های پروژه ..... ۱۱

۱۳.....	۳-۴ فایل های هر پوشه
۱۴.....	۳-۵ فایل مدل ها
۱۵.....	۳-۶ فایل ویوهای اکانت ها
۱۵.....	۳-۶-۱ ویو ثبت نام
۱۶.....	۳-۶-۲ ویو ورود
۱۷.....	۳-۶-۳ ویو خروج
۱۷.....	۳-۷ فایل ویوهای لیست ها
۱۷.....	۳-۷-۱ ویو خانه
۱۷.....	۳-۷-۲ ویو لیست ها
۱۸.....	۳-۷-۳ ویو اضافه کردن کار
۱۹.....	۳-۷-۴ ویو حذف کار
۱۹.....	۳-۸ فایل مسیر ها
۲۰.....	۳-۹ فایل سند اصلی
۲۱.....	۳-۱۰ فایل ادمین
۲۲.....	۳-۱۱ کد های جاوااسکریپت

#### فصل چهارم: تجزیه و تحلیل سیستم

۲۳.....	مقدمه
۲۴.....	۴-۱ ERD
۲۵.....	۴-۲ DFD

## فصل پنجم: نتیجه گیری و پیشنهادات

مقدمه	۲۶
۵-۱ جمع بندی	۲۷
۵-۲ پیشنهاد ها	۲۷
منابع	۲۸

## مقدمه

فهرست کردن کارها یکی از مهمترین کارهایی است که برای مدیریت اهداف هر فرد نیاز است و ما می توانیم با برنامه نویسی که هدف آن تولید برنامه های کاربردی جهت مکانیزه کردن سیستم های عملیاتی است، چنین برنامه ای تولید کنیم تا افراد بدون استفاده از کاغذ بتوانند آن را انجام دهند.

در چنین پروژه ای نیاز است تا با اهداف، روند تولید و نحوه کار با سیستم آشنا شویم.

# فصل اول

## تعریف مسئله

در این فصل درباره عنوان مسئله و اینکه چرا چنین پروژه و برنامه ای مورد نیاز است و اهداف و کاربرد آن چیست صحبت می کنیم، همچنین ابزار هایی که برای تولید این پروژه استفاده شده اند را مورد بررسی قرار می دهیم.



## ۱-۱ عنوان و مسئله پروژه

تودو لیست یک برنامه یا پروژه ای است که می توان کار های روزمره را فهرست کرد و خط زد.

این برنامه تحت وب می باشد و با رایانه یا موبایل می شود به آن دسترسی داشت.

## ۱-۲ اهداف پروژه

با توجه به این که افراد نیاز دارند تا وظایف روزمره و اهدافشان را جایی بنویسند و بررسی کنند تا بتوانند زندگی و کار خود را به خوبی مدیریت کنند، این برنامه کمک بسیاری به این افراد میکند و هدف کلی از این پروژه نوشتن، بررسی و زمانبندی اهداف است که افراد را هدفمند تر، با برنامه تر و منظم تر می کند و از این نظر برای پیشرفت فردی یا حتی گروهی می تواند بسیار مفید باشد

## ۱-۳ روش انجام پروژه و ابزار

برای انجام چنین پروژه ای نیاز به ابزارها، زبان های برنامه نویسی و فریمورک های مختلفی است که آن ها را مورد بررسی قرار می دهیم.

- **Html**: یک زبان نشانه گذاری و ضروری که برای به وجود آوردن ساختار صفحات وب الزامی است.
- **CSS**: یک زبان برای استایل دهی و شکل دادن به رابط کاربری عناصر صفحات وب است.
- **Java Script**: یک زبان برنامه نویسی وب که برای کنترل رفتار کاربر کاربرد دارد.
- **Bootstrap**: یک فریمورک برای CSS می باشد که برای واکنش گرا شدن سایت بسیار کاربرد دارد.

- Django: یک فریمورک تحت وب و متن باز است که به زبان پایتون نوشته شده که مزایایی چون سرعت، امنیت و مقیاس پذیری را دارد که می توانیم با کمترین خط یک وب اپلیکیشن کاربردی تولید کنیم.

## فصل دوم

### نحوه کار با سیستم

در این فصل تصاویری از سایت خواهیم دید و با نحوه استفاده از برنامه یا وبسایت آشنا می شویم که چگونه از امکانات آن استفاده کنیم.

## ۲-۱ صفحه خانه

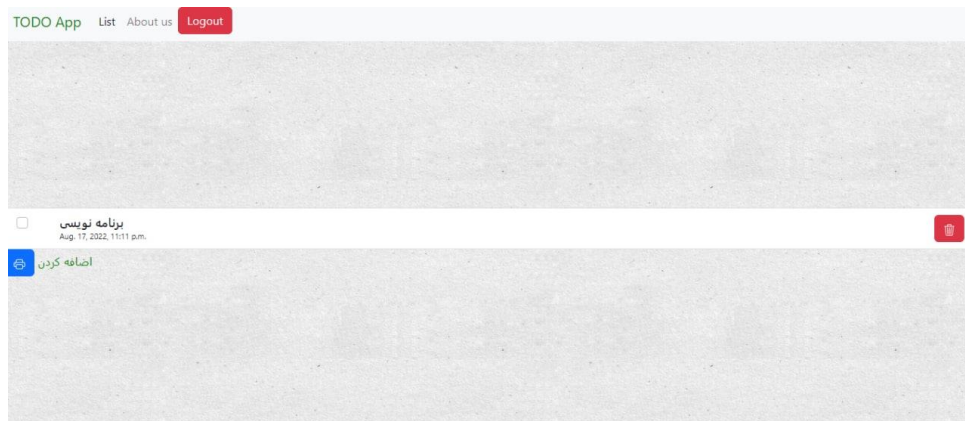


صفحه خانه یا صفحه اول وبسایت دارای رابط کاربری بسیار ساده است که کار را برای کاربران ساده کرده است و کافیه با فشردن دکمه شروع بتوانند ثبت نام کنند و از امکانات سایت استفاده کنند. در چنین برنامه ای تنها چیزی که مهم است بخش تعریف کارها می باشد و در صفحه اصلی از موارد اضافی جلوگیری شده و با دکمه شروع کاربران را هرچه سریعتر به بخش برنامه ریزی وظایف خود منتقل می کند.

## ۲-۲ صفحه ثبت نام

در این بخش کاربران می توانند ثبت نام کنند و تنها چیزی که نیاز دارند یک نام کاربری و گذرواژه می باشد و تنها نکته ای که حائز اهمیت است این است که گذرواژه باید حداقل ۸ کاراکتر داشته باشد و ساده نباشد و برای امنیت حساب هر شخص موضوع مهمی است.

## ۲-۳ صفحه کارها



در این صفحه افراد ثبت نام موفقیت آمیز داشته اند و می توانند از امکانات اصلی برنامه استفاده کنند و کارهای خود را تعریف کنند، خط بزنند و حذف کنند. دو دکمه در پایین فهرست کارها وجود دارد که افراد می توانند کار جدیدی اضافه کنند و حتی می توانند از تمام فهرست پرینت بگیرند. در بالای صفحه در بخش Navbar یک دکمه خروج ظاهر می شود که افراد با فشردن آن می توانند از حساب کاربری خود خارج شوند.

## ۲-۴ صفحه اضافه کردن وظایف

این صفحه با فشردن دکمه اضافه کردن در صفحه لیست کار ها ظاهر می شود و در این بخش می توانیم کار جدیدی را تعریف کنیم.

این صفحه یک فرم دارد که فیلد یکم آن یعنی کار، اجباری می باشد و فیلد دوم آن که برای مشخص کردن تاریخی خاص برای کار خود می باشد، اختیاری است و با فشردن دکمه افزودن به صفحه کار ها منتقل می شویم و فهرست کار های حساب کاربری خود بروزرسانی می شود.

## فصل سوم

### پیاده سازی سیستم

در این فصل با نحوه کارکرد کد های برنامه و ساختار کلی پروژه آشنا می شویم که یک پروژه جنگو و فایل هایش چطور کار می کنند، اینکه چگونه داده ها ذخیره و پردازش می شوند، مسیریابی چگونه است، چگونه پردازشی در اسناد HTML انجام می شود و مسائلی از این قبیل مورد بررسی قرار می گیرند.

### ۳-۱ جنگو چیست؟

جنگو یک چارچوب نرم افزاری یا فریمورک تحت وب و متن باز است که به زبان پایتون نوشته شده است و از معماری Model-View-Template(MVT) پیروی می کند.

هدف اصلی جنگو ساخت آسان سایت های پیچیده و وابسته به دیتابیس است و بر پایه قابلیت استفاده مجدد و قابل اتصال بودن اجزای مختلف، توسعه سریع و اصل خودت را تکرار نکن طراحی شده است. جنگو از پایتون استفاده می کند، حتی برای تنظیمات، فایل ها و مدل های اطلاعات.

از وب سایت های مشهوری که از جنگو استفاده می کنند، می توان به اینستاگرام و پینترست اشاره داشت.

#### ۳-۱-۱ تاریخچه

جنگو در پاییز ۲۰۰۳ توسط Simon Wilison و Adrian Holovaty در حین ایجاد برنامه در شرکت Lawrence-Journal-World متولد شد. سپس در سال ۲۰۰۵ تحت اجازه نامه بی اس دی منتشر شد. نام جنگو از جنگو راینهارت، نوازنده گیتار جاز گرفته شده است. در ژوئن سال ۲۰۰۸، بنیاد نرم افزاری جنگو برای توسعه و حفظ جنگو شکل گرفت.

#### ۳-۱-۲ امکانات

هسته اصلی جنگو از معماری MVC تشکیل شده و در این هسته امکانات بسیاری گنجانده شده. نمونه هایی از امکانات هسته و توزیع جنگو عبارتند از:

- سرور مستقل و داخلی
- اعتبارسنجی برای ذخیره سازی و انتقال اطلاعات
- سیستم قالب بندی صفحات
- رابط کاربری پنل مدیریت
- دسترسی به پایگاه داده بدون استفاده از کوئری SQL



## ۳-۲ نحوه اجرا کردن برنامه

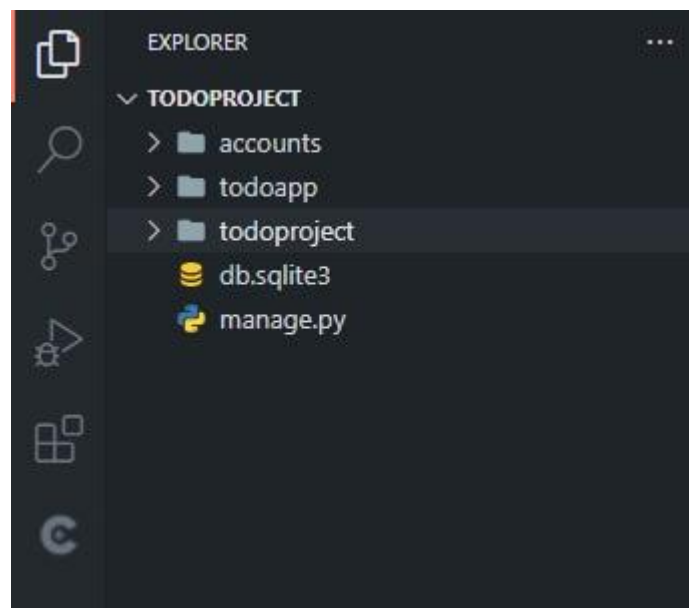
برای اجرا کردن برنامه، نیاز به یک مرورگر، نصب فریمورک جنگو و زبان پایتون بر روی سیستم خود و یک محیط ویرایشگر قدرتمند است که بتوان پروژه های بزرگ را مدیریت کنیم.

دو نمونه ویرایشگر قدرتمند که می توان از آن ها استفاده کرد VS Code و PyCharm می باشد و کافی است درون ویرایشگر پوشه پروژه را باز کنیم و یک ترمینال یا محیط فرمان را باز کنیم و دستور زیر را وارد کنیم:

```
Python /manage.py runserver
```

با این دستور، سرور اجرا می شود و بعد نیاز است که درون مرورگر آدرس محلی کامپیوتر خود را وارد کنیم تا برنامه اجرا شود.

## ۳-۳ پوشه های هر پروژه



یک پروژه جنگو با دستور زیر در ترمینال ساخته می شود و پوشه ها و فایل هایی به صورت پیش فرض ایجاد می کند:

```
Django-admin startproject todoproject
```

بعد از آن نیاز است که وب اپلیکیشن های خود را ایجاد کنیم تا پوشه ای برای آن ها ساخته شود. با استفاده از دستور زیر این کار صورت می گیرد:

```
Python /manage.py startapp todoapp
```

```
Python /manage.py startapp accounts
```

حالا پروژه از سه پوشه و دو فایل تشکیل شده که آن ها را بررسی می کنیم.

پوشه `todoproject`: فایل های کلی پروژه درون این پوشه وجود دارد از جمله فایل تنظیمات.

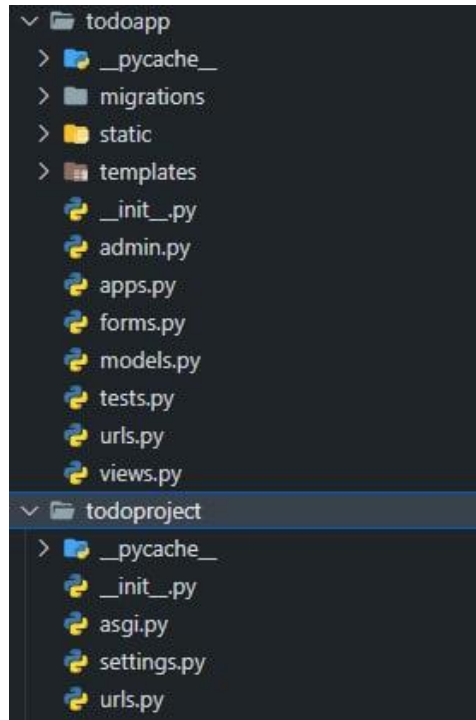
پوشه `todoapp`: این پوشه مربوط به اپلیکیشن لیست ها می شود که فایل های مربوط به اپلیکیشن تودو لیست در این پوشه وجود دارد.

پوشه `accounts`: این پوشه مربوط به اپلیکیشن اکانت ها می شود که برای مدیریت ثبت نام، ورود و خروج ایجاد شده است.

فایل `manage.py`: یکی از فایل های مهم پروژه که وقتی می خواهیم در ترمینال دستوری که پیش ساخته خود جنگو هستند وارد کنیم از این فایل کمک می گیریم و برای اجرا شدن دستور ما باید نام این فایل را کنار دستور وارد کنیم.

فایل `db.sqlite3`: این فایل پایگاه داده ما هست که از پایگاه داده `SQLite` استفاده می کند.

**۳-۴ فایل های هر پوشه**



بعد از ایجاد پروژه و اپ ها، در هر پوشه چندین فایل به صورت پیشفرض ساخته می شود که بعد برنامه نویس بتواند درون آن ها کد نویسی کند. چندین زیر پوشه و فایل مهم که نیاز به برنامه نویسی دارند را مورد بررسی قرار می دهیم:

فایل `views.py`: در این فایل باید توابعی بنویسیم که درخواستی را به سرور میفرستد و پاسخی ارسال می کند و این پاسخ می تواند محتوای سند HTML، عمل `redirect` و هر چیزی باشد و به طور کلی کاربرد آن اجرای دستورات و ارسال آن به سند و صفحه مورد نظر است.

فایل `urls.py`: در این فایل باید کدهای مربوط به مسیریابی سایت را انجام بدهیم و هر مسیر مربوط به یک صفحه خاص می شود که باید آن را توابعی که در فایل `views.py` نوشتیم، بگیریم.

فایل `Models.py`: در این فایل باید کدهای مربوط به پایگاه داده پروژه را بنویسیم و جداول و فیلدهایی با استفاده از کدنویسی ایجاد کنیم.

فایل `Setting.py`: این فایل در پوشه `todoproject` قرار دارد و تنظیمات کلی پروژه از جمله ثبت وب اپلیکیشن ها در این پوشه صورت می گیرد.

فایل `admin.py`: در این فایل کدهای مربوط پنل ادمین نوشته می شود و باید اطلاعات پایگاه داده خود را به آن ارسال کنیم تا پنل ادمین را برای ما ایجاد کند.

پوشه `templates`: در این زیر پوشه تمامی اسناد HTML نگه داری می شود و مهمترین فایل آن `base.html` است که بقیه سند های HTML از آن ارث بری می کنند.

پوشه `statics`: در این زیر پوشه تمامی عکس ها، ایکون ها، ویدیو ها، صداها و فایل CSS در آن نگه داری می شود.

### ۳-۵ فایل مدل ها:



```
models.py x
todoapp > models.py
1 from django.db import models
2 from django.contrib.auth.models import User
3
4 class TodoItem(models.Model):
5     work = models.CharField(max_length=100)
6     checked = models.BooleanField(default=False)
7     owner = models.ForeignKey(User, on_delete=models.CASCADE, null=True)
8     date = models.DateTimeField(blank=True, null=True)
9
10     def __str__(self):
11         return self.work
12
13
```

در این فایل که برای ایجاد پایگاه داده کاربرد دارد، در ابتدا خود جنگو چند شی ایمپورت کرده است تا بتوانیم از امکانات آن استفاده کنیم.

در اینجا کلاسی به نام `TodoItem` داریم که حکم جدول پایگاه داده را دارد و درون آن چندین متغیر داریم که هر کدام حکم فیلد های آن جدول را دارند.

این کلاس از `models.Model` ارث بری می کند که کار ما را برای ایجاد جدول ساده می کند و امکانات خوبی در اختیار برنامه نویس قرار می دهد و کافی است از کلاس `models` نوع داده ای آن فیلد را بنویسیم. چندین فیلد در این جدول ساخته می شود که آن هارا مورد بررسی قرار می دهیم.

`work`: این فیلد کار یا همان وظیفه ای است که کاربر باید تعریف کند.

checked: این فیلد که از نوع Boolean است، برای بررسی وضعیت تمام بودن یا نبودن کار است.

owner: این فیلد که از نوع کلید خارجی است و به پایگاه داده کاربران که به صورت خودکار ایجاد شده وصل می شود و برای تعیین مالک آن کار است.

date: این فیلد که از نوع تاریخ و زمان است برای گرفتن زمان کار مورد نظر است.

نکته ای که حائز اهمیت این است که بعد از نوشتن این کد ها نیاز است تا جدول را به پایگاه داده اضافه کنیم و این کار را با دستورات جنگو در ترمینال انجام می دهیم:

Python /manage.py makemigrations

Python /mange.py migrate

بدین ترتیب تمامی تغییرات در پایگاه داده صورت می گیرد.

## ۳-۶ ویو فایل اکانت ها

در این فایل کد های مربوط به هر یک از عملیاتی که در صفحه مختص به آن است را می نویسیم. هر ویو باید به صورت تابع نوشته شود و request را به صورت پارامتر ارسال کند.

این فایل مربوط به ویو اکانت ها می باشد که کار ورود، خروج و ثبت نام را کنترل می کند.

### ۳-۶-۱ ویو ثبت نام

```
def Signup_View(request):
    if request.method == 'POST':
        form = UserCreationForm(request.POST)
        if form.is_valid():
            user = form.save()
            login(request, user)
            return redirect('todoapp:todo_list')

    form = UserCreationForm()
    return render(request, 'accounts/signup.html', {'form': form})
```

در این بخش کد های مربوط به ثبت نام را می نویسیم.

در ابتدا بررسی می کنیم اگر روشی که اطلاعات فرم را ارسال می کنیم به صورت POST است یا خیر و بعد اگر اینطور بود متغیری با مقدار UserCreationForm می سازیم و با این کار فرمی برای ایجاد کاربر می سازیم و حالا اگر این فرم معتبر بود، آن را save می کنیم و اطلاعات کاربر جدید داخل پایگاه داده ذخیره می شود و بعد این user را login می کنیم و در نهایت با استفاده از redirect او را به صفحه لیست ها منتقل می کنیم و در نهایت این کد را می نویسیم:

```
Return render(request, 'accounts/signup.html', {"form":form})
```

پارامتر اول همان درخواست است، پارامتر دوم باید مسیر سند HTML که می خواهیم آن را رندر کند را می نویسیم و در پارامتر سوم آن متغیر form را به آن سند منتقل می کنیم که بعدا در آن سند بتوانیم بین تگ های HTML از کد ها استفاده کنیم.

## ۲-۶-۳ ویو ورود

```
def Login_View(request):
    if request.method == 'POST':
        form = AuthenticationForm(data= request.POST)
        if form.is_valid():
            user = form.get_user()
            login(request, user)
            return redirect('todoapp:todo_list')

    form = AuthenticationForm()
    return render(request, 'accounts/login.html', {'form':form})
```

در این بخش ویو مربوط به ورود را می نویسیم که تقریبا مثل ویو ثبت نام است فقط در متغیر form مقدار AuthenticationForm را وارد کنیم که به معنای احراز هویت است و اگر این احراز هویت معتبر بود بعد کاربری که در فرم وارد شده را با دستور form.get\_user() می گیریم و بعد با login(request,user) به صفحه کار ها منتقل می کنیم

## ۳-۶-۳ ویو خروج

```
def Logout_View(request):
    if request.method == 'POST':
        logout(request)
        return redirect('/')

```

در این ویو عمل خروج را می نویسیم که در صورتی که کاربر وارد حساب خود شده باشد با دستور `logout(request)` او را به صفحه اصلی سایت منتقل می کنیم

### ۳-۷ ویو فایل لیست ها

در این فایل ویو های مربوط به فهرست کار ها را می نویسیم که عبارتند از نمایش کارها، اضافه کردن کارها، حذف کردن کارها و خانه اصلی سایت.

#### ۳-۷-۱ ویو خانه

```
6 # صفحه خانه
7 def HomeView(request):
8     return render(request, 'todoapp/home.html')
9

```

در این ویو کافیه با استفاده از دستور `render` که در بخش های قبل توضیح داده شد سند مربوط به صفحه اصلی سایت را رندر کنیم تا در سایت نمایش داده شود.

#### ۳-۷-۲ ویو لیست ها

```

6 # صفحه کار ها
7 @login_required(login_url='accounts/login')
8 def todo_List_View(request):
9     user = request.user
10    query = TodoItem.objects.filter(owner = user)
11    if request.method == 'POST':
12        checked_list = request.POST.getlist('checked')
13        checked_list = [int(i) for i in checked_list]
14        for todo_item in query:
15            if todo_item.id in checked_list:
16                TodoItem.objects.filter(id= todo_item.id).update(checked= True)
17            else:
18                TodoItem.objects.filter(id= todo_item.id).update(checked= False)
19        return redirect('/list')
20    todo_list_len = len(query)
21    return render(request, 'todoapp/todo_list.html',{'todolist':query, 'todo_list_len':todo_list_len})
22

```

در این ویو ابتدا با `request.user` آن کاربری که وارد شده است را دریافت می کنیم و بعد در متغیر `query` چنین دستوری می نویسیم که از درون پایگاه داده، آبجکت هایی را دریافت کند که مالک آن همین کاربر وارد شده فعلی می باشد. سپس `id` هر چک باکس را دریافت می کنیم و اگر آیتم انتخاب شده تیک خورده بود، فیلد `checked` آن را `True` کند یا برعکس.

در نهایت با استفاده از `len(query)` تعداد کار ها را دریافت می کنیم که اگر تعداد آن برابر صفر باشد متن چیزی وجود ندارد را نشان دهد که فعلا این متغیر را به سند `HTML` خود ارسال می کنیم که بعدا استفاده کنیم.

نکته ای که وجود دارد این است که ما در بالا با استفاده از دکوریتور `@login_required` اجبار کردیم که برای استفاده از این ویو و صفحه، حتما باید کاربر لاگین شده باشد در غیر این صورت این صفحه برای کاربر نشان داده نخواهد شد.

### ۳-۷-۳ ویو اضافه کردن کار

```
# صفحه اضافه کردن کار جدید
@login_required(login_url='accounts/login')
def todo_item_create(request):
    user = request.user
    if request.method == 'POST':
        form = TodoItemForm(request.POST)
        if form.is_valid():
            instance = form.save(commit=False)
            instance.owner = user
            instance.save()
            return redirect('todoapp:todo_list')
    form = TodoItemForm()
    return render(request, 'todoapp/create_todo_items.html', {'form': form})
```

در این ویو، بیشتر کد شبیه به کد های بخش های قبل است که نیاز به توضیح ندارد اما نکته ای که وجود دارد این است که در متغیر فرم از `TodoItemForm` استفاده شده که به این معنی است که فیلد های مورد نظر را از جدول ایجاد شده خود در پایگاه داده بگیر و به فایل `HTML` مورد نظر که فرم ما در آن جا قرار دارد منتقل کن و اگر اطلاعات فرم صحیح بود، اطلاعات را ذخیره کن و به پایگاه داده منتقل کن.

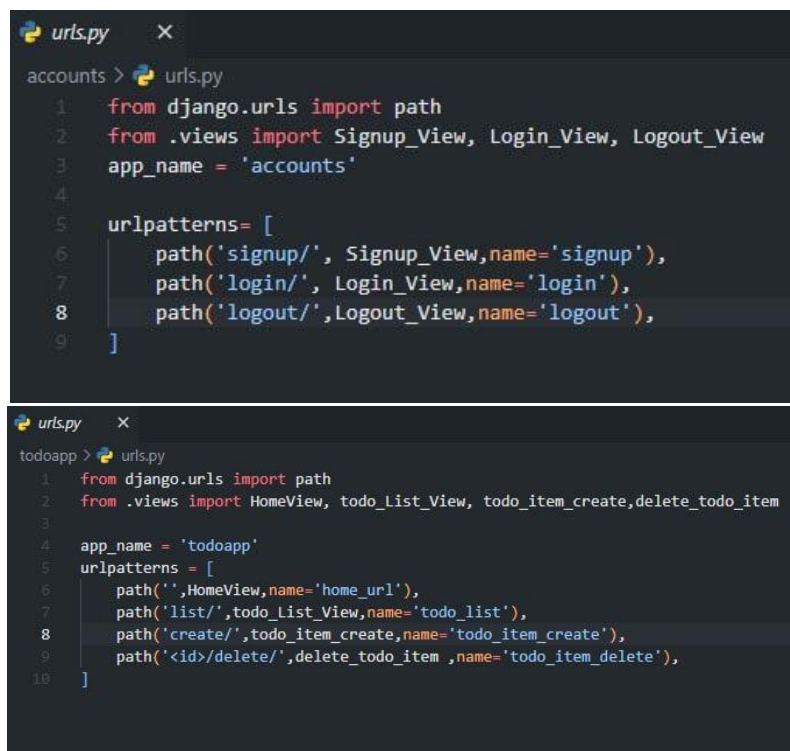
### ۳-۷-۴ ویو حذف کار



```
# حذف یک کار
def delete_todo_item(request, id):
    try:
        item = TodoItem.objects.get(id = id)
    except:
        return redirect('todoapp:todo_list')
    if item.owner == request.user:
        item.delete()
        return redirect('todoapp:todo_list')
    else:
        return redirect('todoapp:todo_list')
```

در این ویو علاوه بر پارامتر request، پارامتر id را هم ارسال می کنیم که اگر کاربر بر روی دکمه حذف کلیک کرد، با استفاده از دستور `TodoItem.objects.get(id=id)` که کوئری select می باشد، id آن کار را پیدا کند و با استفاده از `item.delete()` آن رکورد را به طور کلی حذف کند و بعد به صفحه فعلی عمل انتقال را انجام دهد.

### ۳-۸ فایل مسیریابی



```
urls.py
accounts > urls.py
1 from django.urls import path
2 from .views import Signup_View, Login_View, Logout_View
3 app_name = 'accounts'
4
5 urlpatterns = [
6     path('signup/', Signup_View, name='signup'),
7     path('login/', Login_View, name='login'),
8     path('logout/', Logout_View, name='logout'),
9 ]

urls.py
todoapp > urls.py
1 from django.urls import path
2 from .views import HomeView, todo_list_View, todo_item_create, delete_todo_item
3
4 app_name = 'todoapp'
5 urlpatterns = [
6     path('', HomeView, name='home_url'),
7     path('list/', todo_list_View, name='todo_list'),
8     path('create/', todo_item_create, name='todo_item_create'),
9     path('<id>/delete/', delete_todo_item, name='todo_item_delete'),
10 ]
```

در این فایل ها کد های مربوط به مسیریابی را می نویسیم. تصویر اول مربوط به فایل `urls.py` اکانت ها است و تصویر دوم مربوط به اپ لیست ها.

در این فایل در ابتدا باید ویو های خود را از فایل `view.py` ایمپورت کنیم تا بعد از آن ها استفاده کنیم.

در این فایل لیستی به نام `urlpatterns` ایجاد کرده ایم و هر اندیس از آن را از متدی به اسم `path` استفاده کرده ایم که کار مسیریابی هر صفحه یا ویو را انجام می دهد.

پارامتر اول مربوط به مسیری که بخش `url` مرورگر وارد می کنیم است، پارامتر دوم همان ویویی که می خواهیم به آن مسیر نسبت داده شود است و پارامتر سوم هم نام آن صفحه را می نویسیم.

## ۹-۳ فایل سند اصلی

```
base.html X
todoapp > templates > base.html > html > body
11 <title>{% block title %}{% endblock %}</title>
12 </head>
13 <body>
14 <!-- Navbar -->
15 <nav class="navbar navbar-expand-lg bg-light">
16 <div class="container-fluid">
17 <a style="color: green;" class="navbar-brand" href="#">TODO App</a>
18 <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarSupportedContent" aria-controls="navbarSupportedContent" aria-expanded="false">
19 <span class="navbar-toggler-icon"></span>
20 </button>
21 <div class="collapse navbar-collapse" id="navbarSupportedContent">
22 <ul class="navbar-nav me-auto mb-2 mb-lg-0">
23 <li class="nav-item">
24 <div>
25 <div>
26 <div>
27 <div>
28 <div>
29 <div>
30 <div>
31 <div>
32 <div>
33 <div>
34 <div>
35 <div>
36 <div>
37 <div>
38 <div>
39 <div>
40 <div>
41 <div>
42 <div>
43 <div>
44 <div>
45 <div>
46 <div>
47 <div>
48 <div>
49 <div>
50 <div>
51 <div>
52 <div>
53 <div>
54 <div>
55 <div>
```

در این فایل صفحه آرای سایت انجام شده و تگ های `HTML` و کلاس های آماده `Bootstrap` استفاده شده است.

نکته ای که مهم است این است که جنگو، تگ هایی در اختیار ما قرار داده که بتوانیم آن ها را در بین تگ های HTML استفاده کنیم.

برای مثال تگ های { %block content% } و { %block title% } این امکان را می دهند تا دیگر فایل های HTML مجبور به نوشتن تگ های تکراری نباشند و بتوانند با استفاده از تگ { %extends 'base.html'% } از این فایل ارث بری کنند.

از دیگر تگ هایی که در این صفحه اشاره شده، می توانیم به موارد زیر اشاره کنیم:

{ %if user.is\_authenticated% } : این تگ به این معنا است که اگر کاربری احراز هویت کرده بود، دکمه خروج را نمایش بده در غیر این صورت که با تگ { %else% } مشخص شده، دکمه ثبت نام و ورود را نمایش بده.

{ %url 'accounts:login'% } : این تگ در href لینک ها نوشته می شود و آدرس مشخصی به آن لینک می دهد. در این مثال این آدرس در واقع همان اسمی است که در پارامتر سوم متد path در فایل urls.py داده بودیم.

### ۱۰-۳ فایل ادمین

```
1 from django.contrib import admin
2
3 from .models import TodoItem
4
5 class TodoItemAdmin(admin.ModelAdmin):
6     pass
7
8 admin.site.register(TodoItem, TodoItemAdmin)
```

در این فایل با استفاده امکانات آماده ای که جنگو در ارتباط با پنل ادمین به ما داده است، یک پنل ادمین بسازیم و پایگاه داده خود را به آن وصل کنیم.

همچنین با دستور زیر در ترمینال می توانیم یک مدیر تعریف کنیم:

Python /manage.py makesuperusers

## ۱۱-۳ کد های جاوااسکریپت

```
<script>
let check_input = document.getElementsByName('checked')
let form =document.getElementById('todo_list_form')
let print_btn = document.getElementById('print_paper')
check_input.forEach(Element=>{
  Element.addEventListener('click',function(){
    form.submit()
  })
});
print_btn.addEventListener('click',function(){
  print()
})
</script>
```

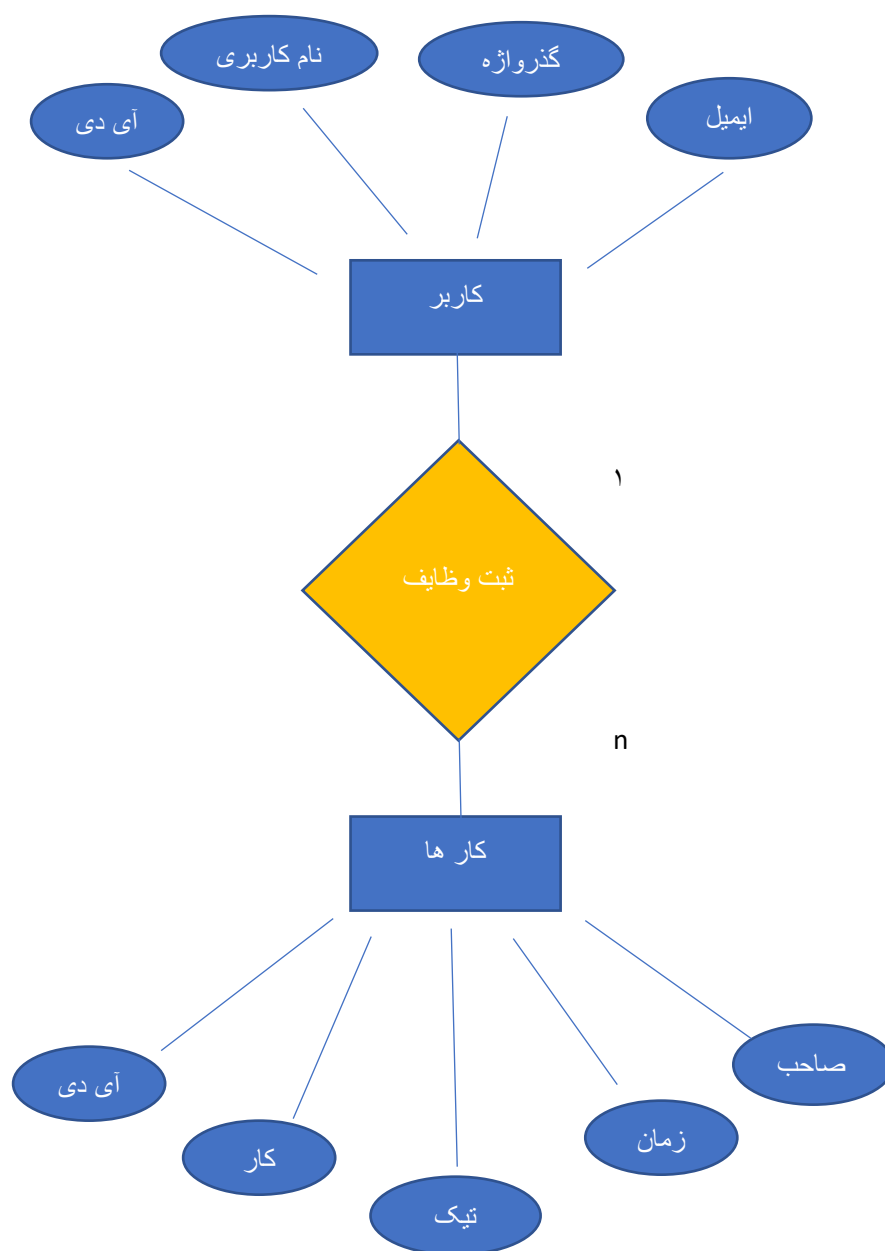
در این کد با استفاده از دو رویداد کلیک، دو کار متفاوت را انجام می دهیم. نوشتن رویداد ها با استفاده از `addEventListener` صورت می گیرد.

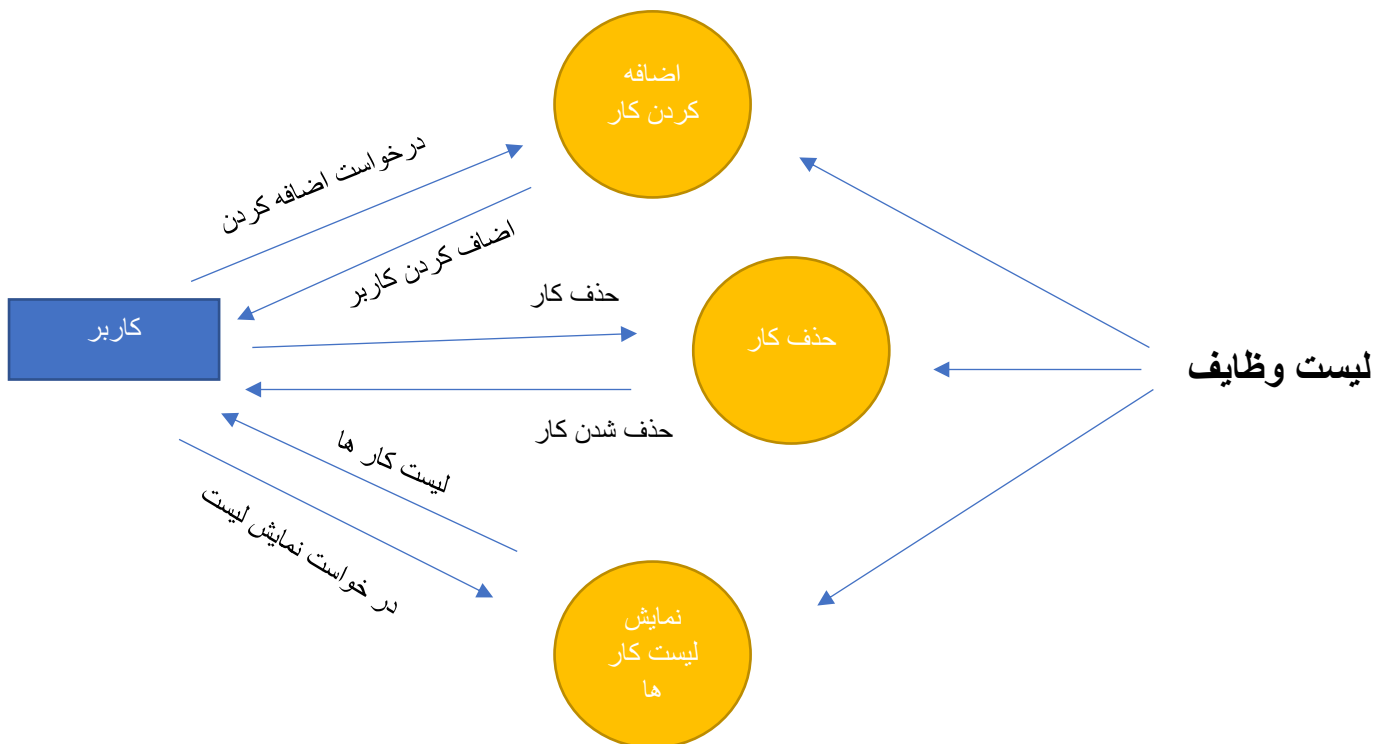
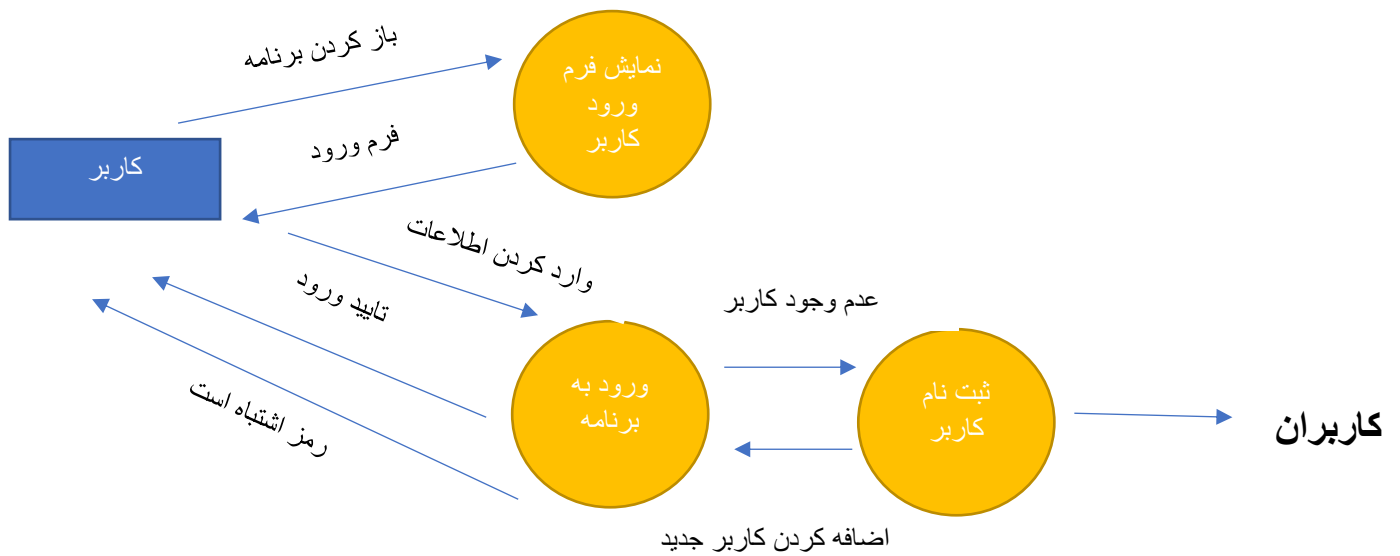
در ابتدا چک باکس و فرم و دکمه پرینت را دریافت می کنیم. در رویداد اول، با استفاده از `form.submit()` کار ثبت خودکار را انجام می دهیم تا هنگام تیک زدن کار های خود نیاز به دکمه ای نداشته باشیم. در رویداد دوم با استفاده از دستور `print()` از صفحه یک پرینت می گیریم.

## فصل چهارم

### تجزیه و تحلیل سیستم

در این فصل با رسم اشکال قصد داریم نحوه کارکرد پایگاه داده و ارتباطات آن ها را مورد بررسی قرار دهیم.





## فصل پنجم

### نتیجه گیری و پیشنهادات



## ۵-۱ نتیجه گیری

فریمورک جنگو فرصت های خوبی را در اختیار توسعه دهندگان وب و صاحبان کسب و کار قرار می دهد. ساخت وب اپلیکیشن هایی با کمترین خط و امکاناتی که کار توسعه و برنامه نویسی را راحت می کند، در وقت صرفه جویی می کند و بسیار مقرون به صرفه است.

همچنین برنامه های مدیریت زمان بسیار به افراد کمک می کنند که وقت و زندگی خود را بهتر مدیریت کنند و به اهداف خود در زندگی برسند و افراد را تشویق به منظم بودن می کنند.

## ۵-۲ پیشنهادات

به علاقه مندان برنامه نویسی تحت وب و توسعه دهندگان پیشنهاد می کنم از فریمورک جنگو برای نوشتن کدهای سمت سرور خود استفاده کنند. گرچه محبوبیت این فریمورک در کشور ما نسبت به دیگر زبان های حوزه خود کمتر است اما در حال رشد است و روز به روز شرکت های بسیاری از آن استفاده می کنند چون بسیار قدرتمند تر و ساده تر است و به زبان پایتون نوشته شده است که دارای کتابخانه ها و امکانات بسیاری است.

منابع:

[https://en.wikipedia.org/wiki/Django \(web framework\)](https://en.wikipedia.org/wiki/Django_(web_framework))

<https://docs.djangoproject.com/en/4.1/>