Django Website Performance Optimization Checklist

1️⃣ Database Optimization

☐ Query কমাও - Use `select_related()` for ForeignKey/OneToOneField - Use `prefetch_related()` for ManyToMany or reverse relation - Avoid N+1 query problem

☐ Index ব্যবহার করো

```
class Product(models.Model):
    name = models.CharField(max_length=255, db_index=True)
```

☐ Bulk operations ব্যবহার করো - Use `bulk_create()`, `bulk_update()` instead of saving in loops

☐ Pagination - বড় list একসাথে load না করে paginate করো

2️⃣ Caching

☐ View-level caching

```
from django.views.decorators.cache import cache_page

@cache_page(60*15)  # 15 মিনিট cache
def product_list(request):
    ...
```

☐ Template fragment caching - Cache template parts that change infrequently

☐ Low-level caching - Use Redis or Memcached for expensive queries

---

3️⃣ Static Files & Media

☐ CSS/JS Minify & Bundle - Use `django-compressor`

☐ Image Optimization - Use compressed formats (e.g., WebP) - Avoid very large images

☐ CDN ব্যবহার করো - Serve static files via CDN for faster load

☐ Lazy Loading - Lazy load images or heavy resources

4️⃣ Middleware & Settings

☐ DEBUG=False in production

☐ GZip Middleware

```
MIDDLEWARE = [
    'django.middleware.gzip.GZipMiddleware',
    ...
]
```

☐ Remove unnecessary middleware

5️⃣ Profiling & Monitoring

☐ Query profiling - Use `django-debug-toolbar` to find slow queries

☐ Performance monitoring - Use `NewRelic` or `Sentry` in production

6️⃣ Extra Tips

☐ Avoid heavy calculations in templates

☐ Use raw SQL if ORM is too slow

☐ Consider database connection pooling if needed

✅ Quick Summary: Query কমাও ✅, Cache ব্যবহার করো ✅, Static Optimize করো ✅, Heavy Calculation avoid করো ✅