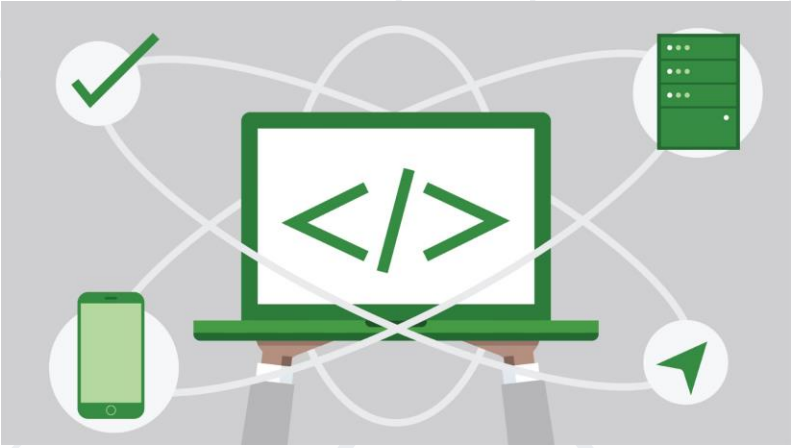
Django Middleware, Sessions, Cookies



SoftUni Team

Technical Trainers

Software University

SoftUni

Software University

https://softuni.bg

Have a Question?

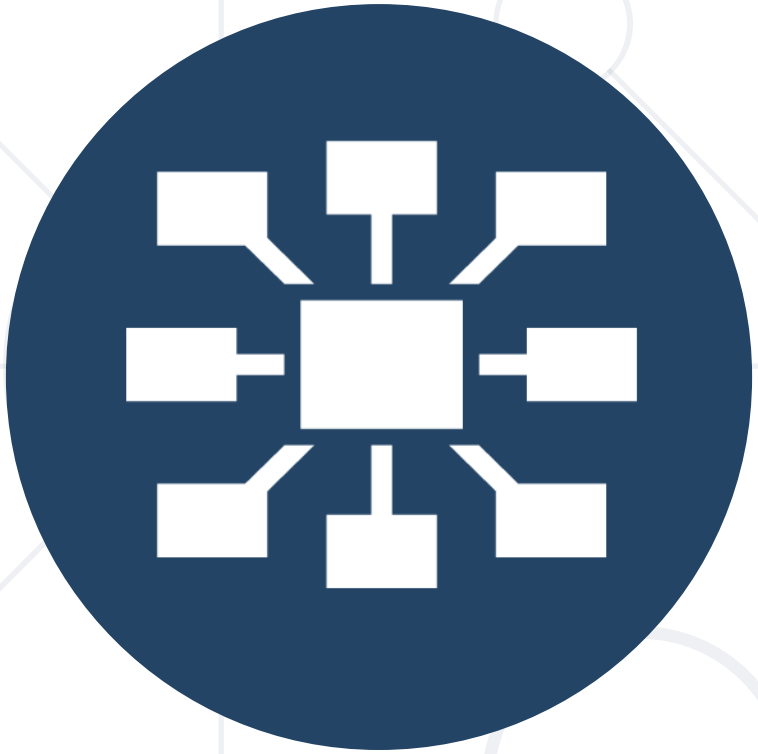
Software University

sli.do

#python-web

Table of Contents

1. **Middleware** in Django
   - Django **Middleware Classes**
   - **Common** Django Middleware
   - **Custom** Middleware
2. **Django Sessions**
   - Session **data**
3. **Cookies**



Django's Middleware Framework

Middleware Framework

- **Middleware** is a **framework** of **hooks** into Django's **request**/**response** processing
  - These hooks allow you to **process requests** and **responses globally** at **different stages**
- Middleware **components** are **processed upon every request and response** that Django handles
  - This **ensures** that certain **functionalities** or **modifications** can be **applied consistently** across the **entire** application

Middleware Framework

- Each **middleware component** is **responsible** for **performing** a **specific function**
- Django provides a set of **built-in** **middleware components** that cover **common** scenarios
- Users can also write their **own** **custom** **middleware** to address **specific application** **requirements**
- **Middleware** plays a **crucial role** in **extending** and **customizing** Django's **request/response** processing

Built-in Middleware

- Django provides a **set** of **built-in middleware**

```
MIDDLEWARE = [
    'django.middleware.security.SecurityMiddleware',
    'django.contrib.sessions.middleware.SessionMiddleware',
    'django.middleware.common.CommonMiddleware',
    'django.middleware.csrf.CsrfViewMiddleware',
    'django.contrib.auth.middleware.AuthenticationMiddleware',
    'django.contrib.messages.middleware.MessageMiddleware',
    'django.middleware.clickjacking.XFrameOptionsMiddleware',
]
```

Middleware Class

- In Django, a Middleware is a **regular Python class** that **processes requests** and **responses** **globally** during the **execution** of a Django project
- **Middleware classes** do **not** have to **subclass** **anything specific**
- To use a **middleware class**, you need to **register** its **path** in the **`MIDDLEWARE`** **setting** in the project's **`settings.py`** file
  - This allows Django to **include** the **middleware** in the **request**/**response** **processing** **pipeline**

Middleware Class

- There are several **key middleware classes**
  - **Process Request**
    - Executed at the **beginning** of the **request phase**
    - Useful for tasks such as authentication, setting global variables, etc
  - **Process View**
    - Called just **before** a **view function** is called
    - It's beneficial for modifying the view function or its arguments

Middleware Class

- **Process Template Response**
  - Allows you to **modify** the **response** **after** the view has been **executed** **but before** it's **rendered**
  - Useful for adding extra context to templates
- **Process Response**
  - Operates on the **response** **after** it has been **processed** by the view
  - It can be used for tasks like setting headers or modifying content

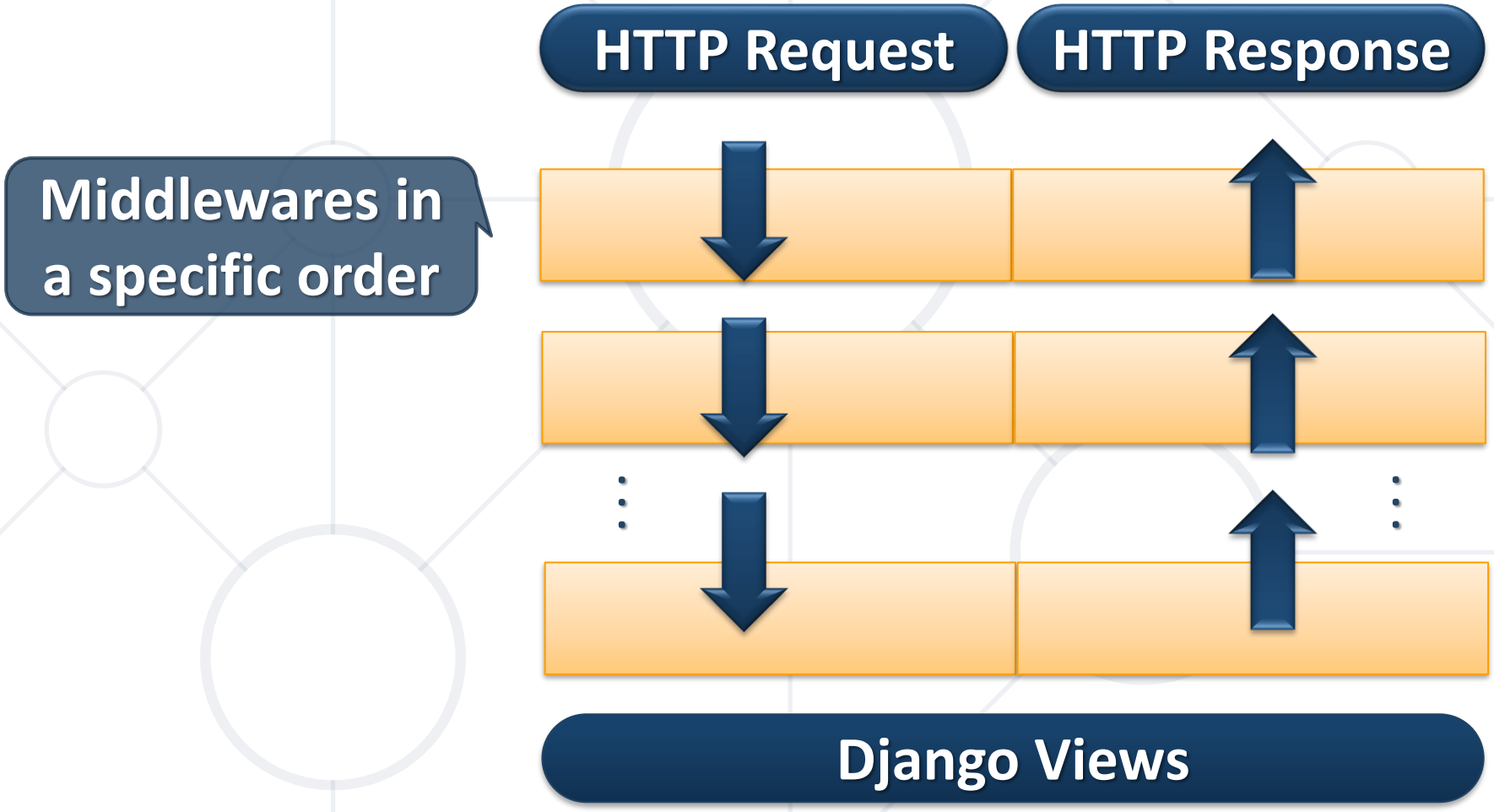
Middleware Ordering

- **Middleware order matters** in Django
- The **order** in which middleware is **listed** in the `MIDDLEWARE` setting **determines** the **order** of **execution**
- **Understanding** this **order** is **crucial** for **managing dependencies** and **ensuring** each **middleware functions** as expected

Middleware Ordering

- Middleware classes are **executed twice** during the **request-response life cycle**
  - Once **before** the **view** is **called**, and once **after** the **view** has **processed** the **request**
- In the **request phase**, **middleware** classes are **executed** in the **order** they are **defined** in the **`MIDDLEWARE`** setting, **from top to bottom**
- During the **response phase**, **middleware** classes are **executed** in the **reverse order** from which they were executed during the request phase

Django Middleware Concept



Common Django Middlewares

- Django comes with several **built-in middlewares** with **specific roles**
  - `CommonMiddleware`: Adds various **HTTP headers** for better security and caching
  - `SecurityMiddleware`: Helps **secure** your application by adding **security-related headers**
  - `SessionMiddleware`: Manages **sessions** for **users**
  - `CsrfViewMiddleware`: Adds **CSRF protection** to view

More built-in Middleware classes: *https://docs.djangoproject.com/en/5.0/ref/middleware/*

Custom Middleware

- You can **create** your **custom** **middleware** by following some rules
  - Define a class with methods **corresponding** to **different** **stages** of **request** **processing**
  - The middleware you create **should** **take** a **request** and **return** a **response**, similar to a view
  - Within this middleware, you can **perform** **actions** **before** the view is called **during** the **request phase** or **after** the view has **processed** the **request** during the **response phase**
  - Ensure to **include** it in the `MIDDLEWARE` setting

Custom Middleware

```
# custom_middleware.py

class MyMiddleware:
    def __init__(self, get_response):
        self.get_response = get_response

    def __call__(self, request):
        # Code to execute before the view or next middleware

        response = self.get_response(request)

        # Code to execute after the view or next middleware

        return response
```

The get_response callable is a function that takes a request and returns a response

Custom Middleware - Factory Function

```
# Middleware Factory
def custom_middleware(get_response):
    def middleware(request):
        # Code to execute before the view or next middleware

        response = get_response(request)
        # Code to execute after the view or next middleware

        return response
    return middleware
```

```
# Middleware Configuration in settings.py
MIDDLEWARE = [
    # Other middlewares...
    'path.to.your.custom_middleware',  # Add your custom middleware
    ]
```

Live Demo

Custom Middleware

Django's Session Framework

Usage and Control

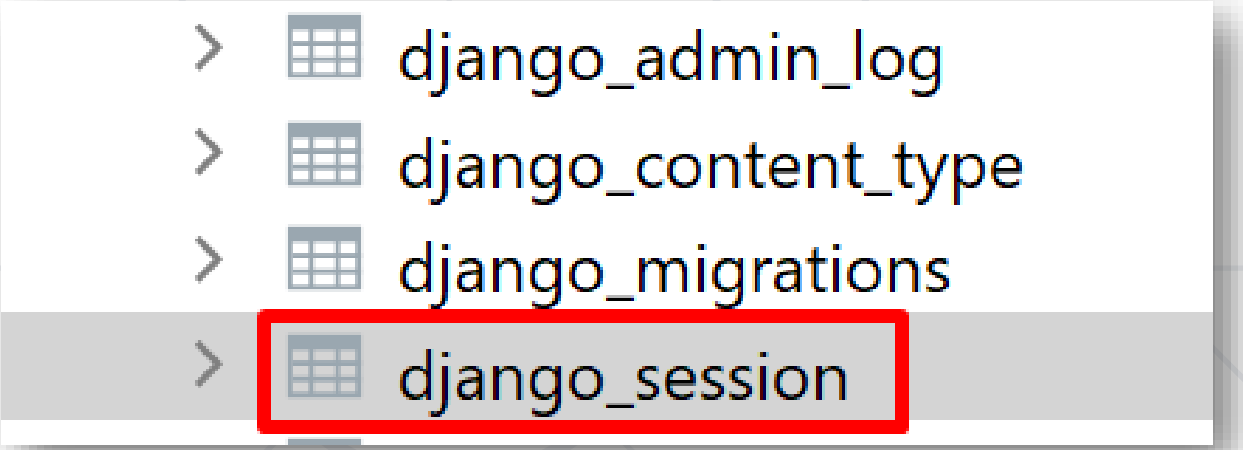
What are Sessions?

- Communication between the **client** and **server** occurs **independently** for each **interaction**, **devoid** of any **inherent connection** between successive **messages**
  - There exists **no concept** of a "sequence" or behavior influenced by prior messages
- **Sessions** play a **pivotal role** in **managing** and **preserving** **arbitrary data** **unique** to each **site visitor**
  - They **facilitate** the tracking of the "**state**" between the **site** and a specific **browser**, **ensuring** that data is **accessible** to the site whenever the browser **establishes** a **connection**

Importance of Sessions

- **Sessions** enable the creation of **personalized** and **dynamic** user **experiences** by **storing** and **managing** data **associated** with **individual** users
- They are instrumental in **tracking** user **authentication**, **preserving** **preferences**, **recording** **browsing activities**, and **storing** **information** entered into form fields
- In essence, **sessions** provide a means to **maintain** and **tailor** the **user's** **interaction** with a website, **ensuring** a **seamless** and **customized** engagement

Django Session Table



The session data is serialized and stored as a string

The default expiration date is 14 days from its initiation

WHERE | ORDER BY

	session_key	session_data	expire_date
1	ieqjlgmo58gm3a90dl…	.eJxVjDsOwyAQBe9CHS…	2024-02-14 16:39:02.919056

Session Structure

Unique Session ID

```
"hje85d3" : {
            user_id: 789,
            username: "FirstUser"
          },
"af354dd" : {
            user_id: 456,
            username: "SecondUser"
          },
"fg78e5s" : {
            user_id: 654,
            username: "ThirdUser"
          }
```

key-value pairs with user data

Enabling Sessions

- **Sessions** are generally **enabled automatically** when you create a new Django project
- In your project's **`settings.py`**, you include **`'django.contrib.sessions'`** in the **`INSTALLED_APPS`** list to ensure the necessary **Django sessions framework** is available

```
INSTALLED_APPS = [
    ...
    'django.contrib.sessions',
    ...
]
MIDDLEWARE = [
    ...
    'django.contrib.sessions.middleware.SessionMiddleware',
    ...
]
```

Accessing Session Data

- The `HttpRequest` **object** (request) in a Django view **comes** with a **session attribute**, functioning as a **dictionary-like object**
- This **attribute enables** the **storage** and **retrieval** of **user-specific data** across different **requests**
- In your views, you have the flexibility to **read**, **write**, or **edit** `request.session` **attribute** at any point, allowing **seamless management** of **session data**

Using Session Data

```python
from django.shortcuts import render

def index(request):
    # Retrieve the current value of 'num_visits' from the session, defaulting to 0
    num_visits = request.session.get('num_visits', 0)

    # Increment the value and store it back in the session
    num_visits += 1
    request.session['num_visits'] = num_visits

    # Create a context with the number of current visits
    context = {'num_visits': num_visits}

    # Render the template with the context
    return render(request, 'index.html', context)
```

Session Data Management Guidelines

- Use **strings** as dictionary **keys**
- **Avoid keys** that **start** with an **underscore**
- Do **not overwrite session data** with a **new object**
- Store **only essential information** in the session
- **Avoid** storing **sensitive information**
- Before **accessing** session data, **check** if the **session key exists**
  - Use `request.session.get('key', default)`

More about Django Sessions: *https://docs.djangoproject.com/en/5.0/topics/http/sessions/*

Live Demo

Working with Session Data

Cookies

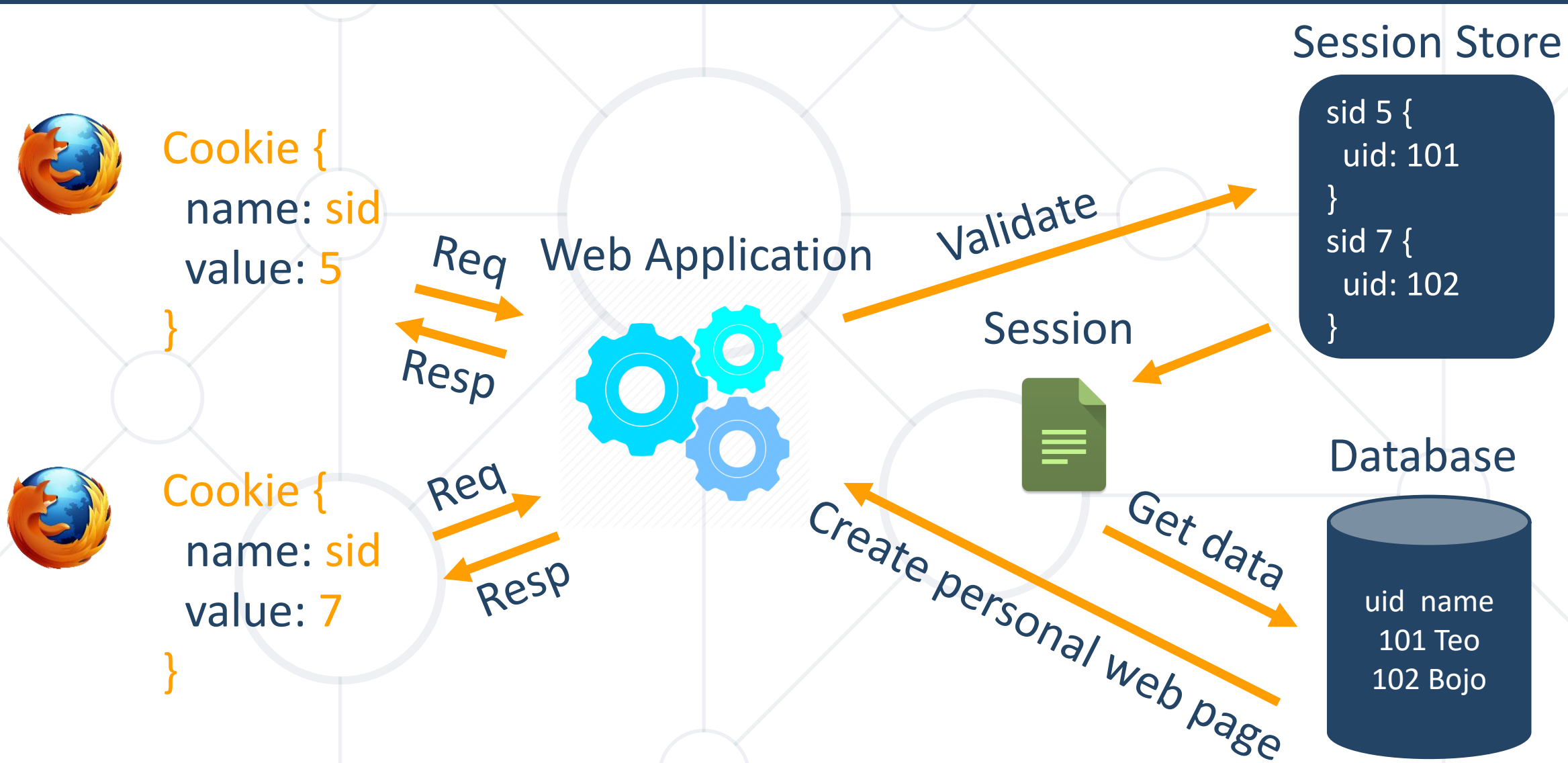
What Are Cookies?

- **Cookies** serve as a **mechanism** for **storing** **information** on a **user's** **device**
- They are **small files** of **plain text**, **devoid** of executable code, **sent** by the **server** to the **client's** **browser**
  - Once **received**, these files are **stored** on the **client's device**, whether it be a computer, tablet, or another platform
- Each **cookie** **holds** a **small piece** of **data** **specific** to a **particular client** and **website**
- **Importantly**, **cookies** are **exclusively** **stored** on the **client-side machine**, **ensuring** that the **information** is **accessible** and **retrievable** **only** by the **user's device**

Cookies in Django

- In the context of Django, a **cookie** containing a **special session ID** is **employed** to **uniquely identify each browser** and its **associated session** with the **website**
- This **session ID facilitates** the seamless **management** of **user interactions** and **stateful information** throughout the **user's visit** to the site
- Django's **cookie-based session management** enhances **security** and allows **efficient user-specific** data **handling**

Relation with Cookies



What's inside?

- The cookie file contains a table of **key-value** pairs

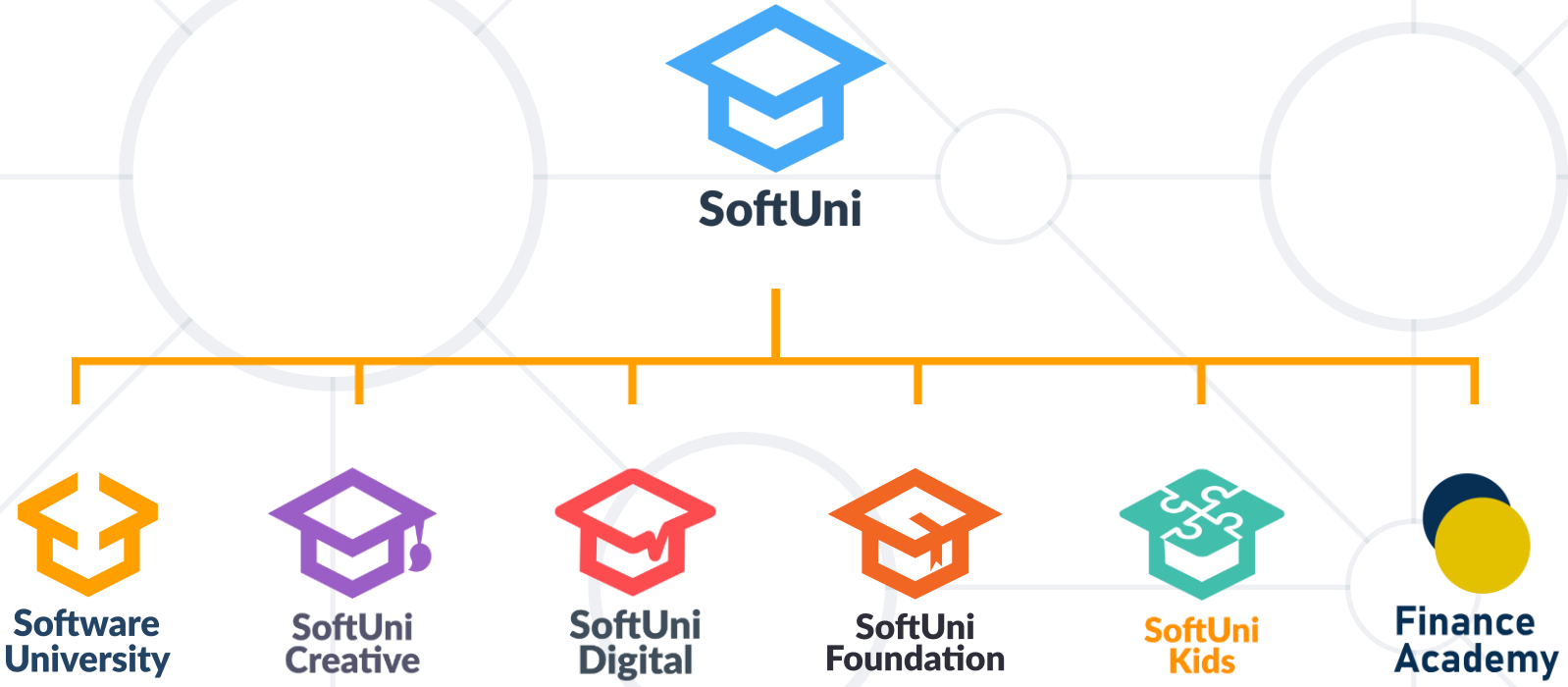
Name:	ELOQUA
Content:	GUID=50B3A712CDAA4A208FE95CE1F2BA7063
Domain:	.oracle.com
Path:	/
Send for:	Any kind of connection
Accessible to script:	Yes
Created:	Monday, August 15, 2016 at 11:38:50 PM
Expires:	Wednesday, August 15, 2018 at 11:38:51 PM

Remove

Summary

- Django **Middleware**
  - **Built-in** Middleware
  - **Custom** Middleware
- Django **Sessions**
  - Using **session data**
- **Cookies**

Questions?



SoftUni Diamond Partners

Software University

SmartIT

Coca-Cola HBC Bulgaria

SOFTWARE GROUP

DXC TECHNOLOGY

createX

Postbank
Решения за твоето утре

SUPER HOSTING .BG

DRAFT KINGS

VIVACOM

INDEAVR
Serving the high achievers

AMBITIONED

PHAR VISION

Trainings @ Software University (SoftUni)

- Software University – High-Quality Education, Profession and Job for Software Developers
  - softuni.bg, softuni.org
- Software University Foundation
  - softuni.foundation
- Software University @ Facebook
  - facebook.com/SoftwareUniversity

License

- This course (slides, examples, demos, exercises, homework, documents, videos and other assets) is **copyrighted content**
- Unauthorized copy, reproduction or use is illegal
- © SoftUni – https://about.softuni.bg/
- © Software University – https://softuni.bg