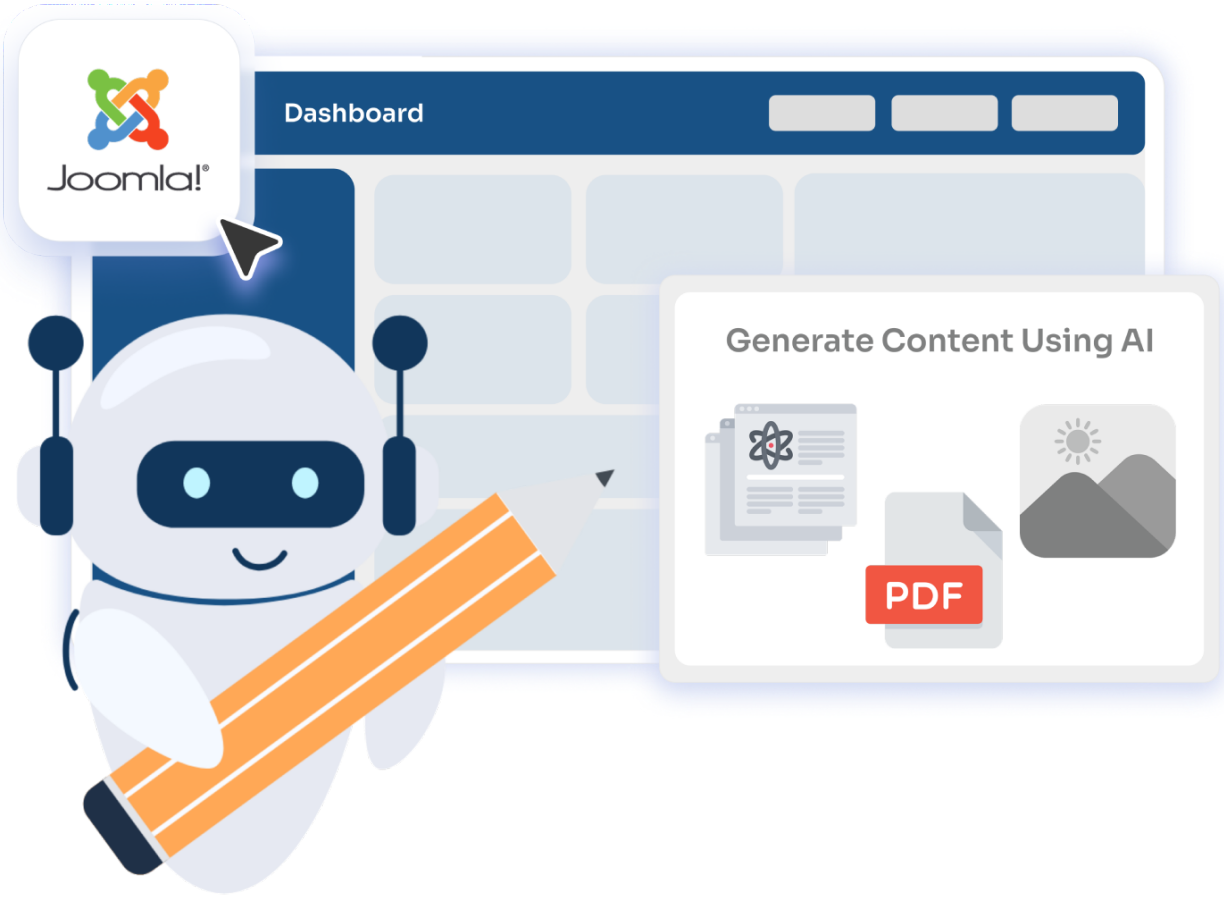
جوملا ۶ هوشمند

از طراحی تم تا بات های هوش مصنوعی

راهنمای جامع برای توسعه دهندگان فارسی زبان

روح الله بلوردی

اولین مترجم رسمی جوملا و مامبو از سال ۱۳۸۴



جوملا ۶ هوشمند

از تم‌سازی ایرانی تا چت‌بات هوشمند

راهنمای جامع برای توسعه‌دهندگان فارسی‌زبان

نوشته‌شده توسط

روح‌الله بلوردی

خوشنویس، طراح گرافیک و توسعه‌دهنده‌ی وب

اولین مترجم رسمی جوملا و مامبو از سال ۱۳۸۴

balvardi.ir

فهرست مطالب

فصل ۱: چرا جوملا؟ معرفی مجدد یک CMS فراموش‌شده ولی قدرتمند

فصل ۲: طراحی کامپوننت سفارشی در جوملا ۶

فصل ۳: جوملا و چالش‌های وب‌نویسی در ایران

فصل ۴: طراحی تم جوملا با الهام از هنر ایرانی

فصل ۵: بهینه‌سازی عملکرد جوملا برای هاست‌های داخلی

فصل ۶: ایجاد کامپوننت چندزبانه

فصل ۷: ادغام هوش مصنوعی در جوملا

فصل ۸: تم‌سازی مدرن با Tailwind CSS

فصل ۹: امنیت پیشرفته — سیستم آزمون امن

فصل ۱۰: ساخت چت‌بات هوشمند

منابع و تشکر

فصل ۱: چرا جوملا؟ معرفی مجدد یک CMS فراموش‌شده ولی قدرتمند

در دنیای امروز سیستم‌های مدیریت محتوا (CMS)، نام‌هایی مانند **وردپرس** و **درودپال** به‌راحتی در گوش هر وب‌سازی جا افتاده‌اند. اما نامی که سال‌ها پیش در میان توسعه‌دهندگان حرفه‌ای ایرانی بسیار محبوب بود و امروزه تقریباً از یاد رفته — **جوملا!** (Joomla!) — همچنان یکی از قدرتمندترین و ساختارمندترین CMSهای متن‌باز دنیاست. در این مقاله، قصد داریم دوباره نگاهی به جوملا بیندازیم و دلایلی را بررسی کنیم که چرا هنوز هم می‌تواند گزینه‌ی هوشمندانه‌ای برای پروژه‌های خاص باشد.

تاریخچه‌ی جوملا: از مامبو تا جوملا!

جوملا در سال ۲۰۰۵ به‌عنوان یک شاخه (fork) از سیستم مدیریت محتوی **مامبو** متولد شد. این انشعاب در پی اختلافاتی بین جامعه‌ی توسعه‌دهندگان و مالک برند مامبو رخ داد. جامعه‌ی جوملا با اصولی مانند **شفافیت، دموکراسی و متن‌باز بودن**، در کمتر از چند سال به یکی از سه CMS پرکاربرد جهان تبدیل شد — جایی که همچنان در بسیاری از آمارهای جهانی در رتبه‌ی دوم یا سوم قرار دارد.

در ایران نیز، جوملا در دهه‌ی ۱۳۸۰ و اوایل ۱۳۹۰ مورد استقبال گسترده‌ای قرار گرفت. دلیل اصلی آن، ساختار ماژولار، قابلیت سفارشی‌سازی بالا و پشتیبانی داخلی از RTL و فارسی بود. بسیاری از سایت‌های دولتی، فرهنگی و تجاری ایرانی در آن دوره با جوملا ساخته شدند.

مقایسه‌ی جوملا با وردپرس و دروپال: نگاهی از دید یک توسعه‌دهنده‌ی ایرانی

هر سه سیستم مدیریت محتوای متن‌باز هستند، اما تفاوت‌های ساختاری عمیقی دارند:

- **وردپرس:** بسیار ساده برای کاربران معمولی، ولی ساختار داخلی آن برای پروژه‌های پیچیده (مثل سیستم‌های چندسطحی یا سایت‌های چندنقشی) محدودیت‌هایی دارد. امنیت آن نیز بیشتر به پلاگین‌ها وابسته است.
- **دروپال:** بسیار قدرتمند و انعطاف‌پذیر، اما منحنی یادگیری بسیار تندی دارد و برای پروژه‌های کوچک یا متوسط، سنگین به‌نظر می‌رسد.
- **جوملا:** تعادل هوشمندانه‌ای بین سادگی و قدرت فنی ایجاد می‌کند. ساختار MVC، سیستم دسترسی کاربران (ACL) بسیار دقیق، و معماری ماژولار آن، توسعه‌ی برنامه‌های پیچیده را بدون از دست دادن کاربرپسندی ممکن می‌سازد.

مزایای ساختاری جوملا

۱. معماری مبتنی بر کامپوننت

هر بخش اصلی سایت در جوملا یک «کامپوننت» مستقل است. این معماری اجازه می‌دهد تا بدون درگیر شدن با هسته، قابلیت‌های کاملاً جدیدی به سیستم اضافه کنید — از فروشگاه گرفته تا سیستم آموزش آنلاین.

۲. سیستم کنترل دسترسی (ACL) بسیار قدرتمند

در جوملا می‌توانید برای هر کاربر، گروه کاربری و حتی هر محتوا، سطح دسترسی دقیقی تعریف کنید. این ویژگی برای پروژه‌هایی مانند سیستم‌های داخلی سازمانی، شبکه‌های اجتماعی یا آموزش‌های چندسطحی بسیار حیاتی است.

۳. پشتیبانی داخلی از MVC

برخلاف وردپرس که MVC را به‌صورت غیررسمی از طریق فریم‌ورک‌های سوم‌طرفه پیاده‌سازی می‌کند، جوملا از ابتدا بر پایه‌ی یک لایه‌بندی MVC طراحی شده است. این موضوع کد شما را تمیز، قابل نگهداری و تست‌پذیر می‌کند.

چرا هنوز جوملا برای پروژه‌های خاص گزینه‌ی عالی است؟

اگر هدف شما ساخت یک وبلاگ شخصی ساده است، شاید وردپرس گزینه‌ی منطقی‌تری باشد. اما در سناریوهای زیر، جوملا می‌تواند انتخاب بسیار هوشمندانه‌تری باشد:

- ساخت سایت‌های چندزبانه با کنترل کامل بر ترجمه‌ی محتوا، منو و عناصر رابط کاربری
- پیاده‌سازی سیستم‌های داخلی سازمانی با نقش‌های کاربری پیچیده

- توسعه‌ی سامانه‌هایی که نیاز به کنترل دقیق بر لایف‌سایکل محتوا دارند (مثل سیستم‌های خبری چندمرحله‌ای)
- پروژه‌هایی که به‌طور همزمان نیاز به «محتوای عمومی» و «دسترسی خصوصی» دارند

تجربه‌ی شخصی: اولین مترجم رسمی جوملا در ایران

در سال‌های ابتدایی فعالیت‌ام در حوزه‌ی وب، جوملا نقطه‌ی آغازین راه بود. آن زمان، پنل مدیریت آن به فارسی نبود و بسیاری از توسعه‌دهندگان ایرانی با آن مشکل داشتند. با همکاری گروهی از دوستان، ترجمه‌ی رسمی جوملا را به فارسی آماده کردیم — ترجمه‌ای که بعدها توسط تیم جهانی جوملا پذیرفته و به‌عنوان زبان رسمی در نسخه‌های بعدی گنجانده شد. این تجربه به من نشان داد که جوملا فقط یک CMS نیست، بلکه یک اکوسیستم باز، هوشمند و جامعه‌محور است.

جمع‌بندی: جوملا فراموش‌شده، نه منسوخ‌شده

فراموش‌شدن جوملا در میان عموم کاربران، بیشتر ناشی از بازاریابی قوی وردپرس و رشد اکوسیستم افزونه‌های آن است — نه ضعف فنی جوملا. برای توسعه‌دهندگانی که به دنبال کنترل بالا، ساختار تمیز و قابلیت مقیاس‌پذیری هستند، جوملا همچنان گزینه‌ای است که نباید نادیده گرفت.

شاید وقت آن رسیده باشد که جامعه‌ی توسعه‌دهندگان فارسی‌زبان، دوباره به این CMS قدرتمند توجه کند — نه به‌عنوان یک یادگاره، بلکه به‌عنوان یک ابزار زنده و کاربردی برای آینده‌ی وب فارسی.

فصل ۲: طراحی کامپوننت سفارشی در جوملا ۶

اگر با جوملا آشنا هستید، احتمالاً می‌دانید که قلب این سیستم مدیریت محتوا، **کامپوننت‌ها** (Components) هستند. هر بخش اصلی سایت — از مطالب گرفته تا کاربران، فروشگاه و سیستم آموزش — یک کامپوننت مستقل است. این معماری، جوملا را به یک پلتفرم بسیار انعطاف‌پذیر تبدیل کرده است. در این مقاله، قصد داریم **گام‌به‌گام** نحوه‌ی ساخت یک کامپوننت سفارشی در **جوملا ۶** را آموزش دهیم — از ساختار پوشه‌ها تا اتصال به پایگاه‌داده و پیاده‌سازی امنیت.

چرا کامپوننت‌نویسی در جوملا ۶ مهم است؟

جوملا ۶ (که بر پایه‌ی PHP 8 و فریم‌ورک **Laravel-inspired** ساخته شده) تغییرات عمیقی در معماری نسبت به نسخه‌های قبلی داشته است. این نسخه، توسعه‌ی کامپوننت‌ها را ساده‌تر، ایمن‌تر و متناسب با استانداردهای مدرن PHP کرده است. یادگیری این فرآیند نه‌تنها به شما امکان می‌دهد نیازهای خاص پروژه‌های خود را پیاده‌سازی کنید، بلکه پایه‌ای برای ساخت افزونه‌های حرفه‌ای و قابل بازفروشی فراهم می‌کند.

گام ۱: ساختار پوشه‌های یک کامپوننت جوملا ۶

تمام کامپوننت‌های جوملا در مسیر زیر قرار می‌گیرند:

```
/components/com_mycomponent/
```

در این مثال، `com_mycomponent` نام کامپوننت شماست (**حتماً با پیشوند** `_com` **شروع شود**). ساختار استاندارد یک کامپوننت جوملا ۶ به این شکل است:

```
com_mycomponent/
├── src/
│   ├── Controller/
│   ├── Model/
│   ├── View/
│   └── Helper/
├── tmpl/
│   └── default.php
├── mycomponent.php
└── services/provider.php
```

گام ۲: فایل اصلی کامپوننت (`mycomponent.php`)

این فایل نقطه‌ی ورود کامپوننت است. معمولاً فقط یک خط دارد که کلاس اصلی را فراخوانی می‌کند:

```php
<?php
defined('_JEXEC') or die;

use Joomla\CMS\Component\Router\RouterFactoryInterface;
use Joomla\CMS\Dispatcher\ComponentDispatcherFactoryInterface;
use Joomla\CMS\Extension\ComponentInterface;
use Joomla\CMS\Extension\MVCComponent;
use Joomla\CMS\Extension\Service\Provider\ComponentDispatcherFactory;
use Joomla\CMS\Extension\Service\Provider\MVCFactory;
use Joomla\CMS\MVC\Factory\MVCFactoryInterface;
use Joomla\DI\Container;
```

```
use Joomla\DI\ServiceProviderInterface;

return new class implements ServiceProviderInterface
{
    public function register(Container $container)
    {
        $container->registerServiceProvider(new MVCFactory('\\Mycomponent'));
        $container->registerServiceProvider(new ComponentDispatcherFactory('\\Mycomponent'));
        $container->set(
            ComponentInterface::class,
            function (Container $container) {
                $component = new MVCComponent($container->get(ComponentDispatcherFactoryInterface::class));
                $component->setMvcFactory($container->get(MVCFactoryInterface::class));
                return $component;
            }
        );
    }
};
```

گام ۳: ایجاد کنترلر (Controller)

کنترلر وظیفه‌ی مدیریت درخواست‌ها را دارد. مثلاً برای نمایش لیست محصولات:

```
<?php
namespace Mycomponent\Controller;

use Joomla\CMS\MVC\Controller\BaseController;

class DisplayController extends BaseController
{
    protected $default_view = 'products';

    public function display($cachable = false, $urlparams = false)
    {
        return parent::display($cachable, $urlparams);
    }
}
```

گام ۴: ایجاد مدل (Model)

مدل‌ها برای واکشی داده از پایگاه‌داده استفاده می‌شوند.

```
<?php
namespace Mycomponent\Model;

use Joomla\CMS\MVC\Model\ListModel;
use Joomla\Database\DatabaseInterface;

class ProductsModel extends ListModel
{
```

```
    public function __construct($config = [], DatabaseInterface $db = null)
    {
        if (empty($config['filter_fields'])) {
            $config['filter_fields'] = ['id', 'title', 'published'];
        }
        parent::__construct($config, $db);
    }

    protected function getListQuery()
    {
        $db = $this->getDatabase();
        $query = $db->getQuery(true)
            ->select('*')
            ->from($db->quoteName('#__mycomponent_products'))
            ->where($db->quoteName('published') . ' = 1');
        return $query;
    }
}
```

گام ۵: ایجاد نمای (View) و قالب (Template)

ویو مسئول آماده‌سازی داده‌ها برای نمایش است:

```
<?php
namespace Mycomponent\View\Products;

use Joomla\CMS\MVC\View\HtmlView as BaseHtmlView;

class HtmlView extends BaseHtmlView
{
    protected $items;
    protected $pagination;

    public function display($tpl = null)
    {
        $this->items = $this->get('Items');
        $this->pagination = $this->get('Pagination');
        parent::display($tpl);
    }
}
```

و فایل قالب در `tmpl/default.php/` :

```
<?php defined('_JEXEC') or die; ?>
<div class="mycomponent-products">
  <?php foreach ($this->items as $item): ?>
    <div class="product-item">
      <h3><?php echo htmlspecialchars($item->title, ENT_QUOTES, 'UTF-8'); ?></h3>
    </div>
  <?php endforeach; ?>
  <div class="pagination">
```

```
    <?php echo $this->pagination->getPagesLinks(); ?>
  </div>
</div>
```

گام ۶: اتصال به پایگاه‌داده

برای ایجاد جدول، فایل `install.mysql.utf8.sql` در پوشه‌ی `/sql` کامپوننت قرار دهید:

```
CREATE TABLE IF NOT EXISTS `#__mycomponent_products` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `title` varchar(255) NOT NULL,
  `published` tinyint(1) NOT NULL DEFAULT '1',
  PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_persian_ci;
```

گام ۷: امنیت در کامپوننت‌نویسی

نکات کلیدی امنیتی:

- همیشه از `()htmlspecialchars` برای خروجی‌های کاربری استفاده کنید تا از حملات XSS جلوگیری شود.
- در کوئری‌ها از `()db->quoteName$` و `()db->quote$` استفاده کنید.
- ورودی‌های کاربر را با `Joomla\CMS\Factory::getApplication()->input` دریافت کنید و فیلتر کنید.
- در صفحات ادمین، حتماً از `JHtml::_('form.token')` برای جلوگیری از CSRF استفاده کنید.

گام ۸: ثبت منو، ماژول و پلاگین (اختیاری)

با ایجاد فایل‌های `mycomponent.xml` و تعریف بخش‌های `<administration>` و `<files>`، می‌توانید کامپوننت خود را قابل نصب کنید و در منوی مدیریت نمایش دهید.

تجربه‌ی عملی: کامپوننت فروشگاهی من

در یکی از پروژه‌های شخصی‌ام، یک کامپوننت فروشگاهی ساده برای جوملا ۶ توسعه دادم که تنها ۱۲ فایل داشت، اما تمام قابلیت‌های اولیه‌ی یک فروشگاه — لیست محصولات، جزئیات، سبد خرید (با استفاده از سشن) و پرداخت آفلاین — را پوشش می‌داد. این انعطاف‌پذیری، همان چیزی است که جوملا را برای من جذاب نگه داشته است.

جمع‌بندی

کامپوننت‌نویسی در جوملا ۶ دیگر آن‌قدرها هم پیچیده نیست. با رعایت ساختار MVC، استفاده از ابزارهای داخلی جوملا و رعایت اصول امنیتی، می‌توانید در کمتر از چند ساعت، یک ماژول کاربردی و ایمن بسازید. این هنر، همان چیزی است که جوملا را از «سیستم مدیریت محتوا» به «چارچوب توسعه‌ی کاربرد وب» تبدیل می‌کند.

فصل ۳: جوملا و چالش‌های وب‌نویسی در ایران

اگرچه جوملا یک سیستم مدیریت محتوای جهانی و متن‌باز است، اما هنگام استفاده در محیط فارسی‌زبان — به‌ویژه در ایران — با چالش‌هایی مواجه می‌شود که در مستندات رسمی آن به‌ندرت مورد توجه قرار گرفته‌اند. این چالش‌ها از RTL و فونت‌های فارسی گرفته تا یکپارچه‌سازی با سرویس‌های داخلی مانند درگاه‌های پرداخت و SMSها متغیر است. در این مقاله، بر اساس سال‌ها تجربه در توسعه‌ی وب فارسی، راهکارهایی عملی و قابل اجرا برای غلبه بر این موانع ارائه می‌دهم.

چالش ۱: پشتیبانی از زبان راست‌به‌چپ (RTL)

جوملا از نسخه‌های اولیه خود پشتیبانی RTL دارد، اما بسیاری از قالب‌های غربی به‌درستی آن را پیاده‌سازی نکرده‌اند. نتیجه؟ عنصرهای رابط کاربری به‌هم‌ریخته، منوها معکوس و فرم‌ها غیرقابل استفاده.

راهکار:

- همیشه از قالب‌هایی استفاده کنید که دارای فایل `rtl.css` باشند.
- در صورت استفاده از قالب سفارشی، فایل `templateDetails.xml` را ویرایش کنید:

```
<files>
  <filename>css/rtl.css</filename>
</files>
<positions>
  <position>sidebar-rtl</position>
</positions>
```

- در CSS، به‌جای `float: left/right` از `flex-direction: row-reverse` یا `text-align: start/end` استفاده کنید.
- برای کامپوننت‌های سفارشی، در فایل ویو این کد را اضافه کنید:

```
<?php if ($this->direction === 'rtl'): ?>
<link rel="stylesheet" href="components/com_mycomp/css/rtl.css" />
<?php endif; ?>
```

چالش ۲: فونت‌های فارسی و نمایش صحیح کاراکترها

بعضی از هاست‌های ایرانی یا سرورهای قدیمی، کدینگ UTF-8 را به‌درستی پشتیبانی نمی‌کنند. این مسئله باعث نمایش «» یا کاراکترهای ناخوانا می‌شود.

راهکار:

- در فایل `configuration.php`، مطمئن شوید:

```
public $dbtype = 'mysqli';
public $charset = 'utf8mb4';
```

- در فایل‌های PHP سفارشی، حتماً این خط را در بالای فایل قرار دهید:

```
header('Content-Type: text/html; charset=utf-8');
```

- در CSS، فونت فارسی را به‌صورت زیر تعریف کنید:

```
body {
  font-family: "Vazirmatn", "IRANSans", "Tahoma", sans-serif;
}
```

- فونت Vazirmatn (رایگان و متن‌باز) را از ریپازیتوری رسمی دانلود و در قالب خود جاگذاری کنید.

چالش ۳: سئوی فارسی و موتورهای جستجوی داخلی

موتورهای جستجوی مانند یاهوی ایران یا حتی گوگل، گاهی فارسی را به‌درستی ایندکس نمی‌کنند. همچنین، ساختار URLهای جوملا به‌صورت پیش‌فرض لاتین است.

راهکار:

- افزونه‌ی **SH404SEF** یا **404SEF** (برای جوملا ۶) را نصب کنید تا URLهای فارسی تولید شوند.
- در متا تگ‌ها، از حروف فارسی در عنوان و توضیحات استفاده کنید.
- فایل `robots.txt` را بهینه کنید:

```
User-agent: *
Allow: /

Sitemap: https://balvardi.ir/sitemap.xml
```

- برای ایندکس بهتر در موتورهای داخلی، سایت خود را در وب‌ایران و یوز ثبت کنید.

چالش ۴: یکپارچه‌سازی با سرویس‌های ایرانی

درگاه‌های پرداخت، سرویس‌های پیامکی و نقشه‌های ایرانی (مثل نشان) در جوملا پشتیبانی داخلی ندارند.

راهکار:

- **درگاه پرداخت:** یک ماژول سفارشی بنویسید که با API سرویس‌هایی مانند زرین‌پال، آیدی‌پی یا سپرده ارتباط برقرار کند. استفاده از `curl` و JSON برای ارسال درخواست‌ها توصیه می‌شود.
- **SMS:** برای ارسال کد تأیید، سرویس `sms.ir` یا `api.ir` را با استفاده از یک پلاگین سیستمی جوملا فراخوانی کنید:

```
$url = 'https://api.sms.ir/v1/send';
$headers = ['Content-Type: application/json', 'x-api-key: YOUR_KEY'];
$data = json_encode(['mobile' => $phone, 'message' => 'کد تأیید شما: ' . $code]);
```

- **نقشه:** به‌جای Google Maps، از نَشان یا مپ‌آی‌آر استفاده کنید و اسکریپت آن‌ها را در ماژول یا کامپوننت خود بارگذاری نمایید.

چالش ۵: میزبانی و هاست‌های ایرانی

برخی هاست‌های داخلی نسخه‌ی PHP قدیمی یا محدودیت‌های mod_security دارند که با جوملا ۶ (نیازمند PHP 8.1+) تداخل ایجاد می‌کند.

راهکار:

- حتماً قبل از نصب، از هاست‌هایی استفاده کنید که PHP 8.1+ و MySQL 5.7+ را پشتیبانی کنند.
- در صورت وجود خطای `Internal Error 500`، فایل `htaccess.` را به‌صورت دستی ویرایش کنید و خطوط مربوط به mod_security را غیرفعال نمایید.

- برای بهینه‌سازی، از کش داخلی جوملا (**System > Cache**) و همچنین افزونه‌هایی مانند **JotCache** استفاده کنید.

تجربه‌ی شخصی: فارسی‌سازی پنل مدیریت جوملا

در یکی از پروژه‌های سازمانی، نیاز بود کاربران غیرفنی بتوانند به‌راحتی با پنل مدیریت کار کنند. من نه‌تنها رابط کاربری را فارسی کردم، بلکه منوها را بازچینی کردم، فونت Vazirmatn را جایگزین Tahoma کردم و یک سیستم راهنمای داخلی (با پلاگین سیستمی) اضافه کردم. این تغییرات، زمان آموزش کاربران را تا ۷۰٪ کاهش داد.

جمع‌بندی

جوملا یک CMS جهانی است، اما برای استفاده‌ی بهینه در ایران، نیازمند «فارسی‌سازی هوشمند» است — نه فقط ترجمه‌ی لغات، بلکه بازتعریف رابط کاربری، بهینه‌سازی فنی و یکپارچه‌سازی با زیرساخت‌های داخلی. این کار، شاید در ابتدا زمان‌بر به‌نظر برسد، اما در طولانی‌مدت، تجربه‌ی کاربری بسیار بهتری برای مخاطبان فارسی‌زبان ایجاد می‌کند.

به‌یاد داشته باشید: یک سایت فارسی‌زبان موفق، فقط محتوای فارسی ندارد — بلکه تمام لایه‌های آن، از کد تا رابط کاربری، با فرهنگ و زیرساخت‌های ایران هماهنگ است.

فصل ۴: طراحی تم جوملا با الهام از هنر ایرانی

طراحی یک تم (قالب) جوملا تنها مسئله‌ای فنی نیست؛ بلکه فرصتی است برای بیان هویت بصری، فرهنگی و هنری. در دنیایی که اکثر تم‌های غربی بر پایه‌ی سادگی مینیمال و رنگ‌های خنثی ساخته می‌شوند، چرا ما ایرانی‌ها نباید از غنای هزارساله‌ی هنر خود — از **تذهیب** و **نگارگری** گرفته تا **کاشی‌کاری اصفهانی** و **گچ‌بری شیرازی** — الهام بگیریم؟ در این مقاله، گام‌به‌گام نحوه‌ی ساخت یک تم جوملا با روحیه‌ی ایرانی-اسلامی را بر اساس سال‌ها تجربه در طراحی گرافیک و وب برای شما توضیح می‌دهم.

گام ۱: درک اصول هنر ایرانی-اسلامی در طراحی رابط کاربری

هنر ایرانی-اسلامی بر پایه‌ی سه اصل استوار است:

- **نظم هندسی:** استفاده از تناسب‌های ریاضی (مثل نسبت طلایی) در چینش عناصر.
- **تکرار و تقارن:** الگوهای اسلیم و ختایی که بی‌نهایت تکرار می‌شوند و نماد جاودانگی هستند.
- **زیبایی در جزئیات:** هیچ فضای خالی بدون معنا یا زیبایی وجود ندارد؛ هر نقطه‌ای بخشی از یک کل هماهنگ است.

در طراحی تم جوملا، این اصول را به این شکل پیاده‌سازی می‌کنیم:

- استفاده از شبکه‌های طلایی در طرح‌بندی صفحه
- افزودن الگوهای تکراری به‌عنوان بک‌گراند در بخش‌های خاص (مثل هدر یا فوتر)
- طراحی نوارهای منو با الگوهای اسلیم ساده‌شده

گام ۲: استخراج پالت رنگ از آثار تاریخی

رنگ‌های ایرانی مصنوعی نیستند؛ بلکه از زمین، آسمان و هنر ما الهام گرفته شده‌اند. در پروژه‌هایم، معمولاً پالت‌رنگ را از این منابع استخراج می‌کنم:

- **کاشی‌های مسجد شیخ لطف‌الله (اصفهان):** فیروزه‌ای، طلایی، سفید، نیلی
- **تابوت‌های میناکاری سیرجان:** قرمز ارغوانی، نقره‌ای، سبز زمردی
- **نگارگری‌های تیموری و صفوی:** طلایی مات، قهوه‌ای خاکی، آبی کدر

این رنگ‌ها را در فایل `custom.css` تم خود تعریف کنید:

```
:root {
  --iran-blue: #1E5F8C;
  --iran-turquoise: #2A9D8F;
  --iran-gold: #D4AF37;
  --iran-terracotta: #A65F4C;
}
```

گام ۳: استفاده از الگوهای اسلیم و ختایی در پس‌زمینه

استفاده‌ی مستقیم از تصاویر بزرگ اسلیم ممکن است عملکرد سایت را کاهش دهد. راه‌حل‌های هوشمندانه:

- تبدیل الگوها به **SVG** (کوچک، مقیاس‌پذیر و با کیفیت بالا)
- استفاده از آن‌ها فقط در بخش‌هایی مانند هدر یا فوتر
- کاهش پیچیدگی الگو برای نمایش بهتر در موبایل

مثال CSS برای بک‌گراند اسلیم:

```
.header {
  background-image: url('images/eslim-pattern.svg');
```

```
  background-size: 200px;
  background-repeat: repeat;
  opacity: 0.08;
}
```

گام ۴: فونت‌های سفارشی با سبک خط ثلث و نسخ

خط ثلث و نسخ تنها برای خوشنویسی نیست؛ می‌توان آن را در فونت‌های مدرن دیجیتالی گنجاند. من برای پروژه‌هایم دو راه را پیشنهاد می‌کنم:

1. **فونت‌های فارسی دیجیتالی با الهام از خط:** مانند Vazirmatn یا IRANSans که ساده ولی ایرانی هستند.
2. **فونت‌های سفارشی بر اساس آثار خودتان:** اگر — مانند من — خوشنویس هستید، یک فونت متن‌باز از خط ثلث خود ایجاد کنید و آن را در تم جوملا بارگذاری نمایید.

کد CSS برای لود فونت محلی:

```
@font-face {
  font-family: 'Balvardi-Thuluth';
  src: url('fonts/Balvardi-Thuluth.woff2') format('woff2');
  font-weight: normal;
  font-style: normal;
  font-display: swap;
}

h1, h2, .title-ornament {
  font-family: 'Balvardi-Thuluth', 'Vazirmatn', serif;
}
```

گام ۵: طراحی منو با الهام از معماری اسلامی

منو فقط یک لیست نیست؛ می‌تواند بخشی از هنر شما باشد:

- به‌جای خطوط صاف، از خطوط منحنی (C-shaped) در hover استفاده کنید
- آیکون‌ها را با نمادهای ایرانی جایگزین کنید: جایگزینی آیکون "خانه" با نماد "چشمه"، "کاربر" با "عکس نگارگری" و غیره
- برای منوهای عمودی، از نوارهای گچ‌بری‌شده‌ی دیجیتالی به‌عنوان جداکننده استفاده کنید

گام ۶: ادغام هوشمند هنر بدون افت عملکرد

زیبایی نباید به قیمت سرعت تمام شود. برای حفظ بهینه‌بودن:

- تمام تصاویر SVG را با ابزارهایی مثل SVGOMG فشرده کنید
- فونت‌ها فقط برای عناصر خاص (نه تمام صفحه) لود شوند
- از lazy loading برای تصاویر دکوراتیو استفاده کنید
- کدهای CSS را به‌صورت خردشده و با media query مناسب برای موبایل بنویسید

تجربه‌ی شخصی: تم "چهارباغ" برای یک پروژه فرهنگی

در یکی از پروژه‌های اخیر، تمی طراحی کردم که الهام‌گرفته از باغ‌های ایرانی بود: آب‌نماها به‌عنوان خطوط راهنما، درختان سرو در تصاویر پس‌زمینه، و منویی که هنگام اسکرول، گل سرخ از حاشیه بیرون می‌آمد. این تم نه‌تنها زیبا بود، بلکه بار فرهنگی سنگینی را به سایت اضافه کرد — بدون اینکه عملکرد آن کمتر از ۹۰ امتیاز در Google PageSpeed شود.

جمع‌بندی

تم جوملا شما می‌تواند فراتر از یک پوسته باشد؛ می‌تواند **نگارگری دیجیتالی** باشد، **تذهیبی مدرن** باشد، یا حتی **یک باغ ایرانی در فضای مجازی**. کلید موفقیت، ترکیب دانش فنی جوملا با دیدگاه هنری ایرانی است. وقتی این دو دست در دست هم دهند، نه‌تنها یک وب‌سایت می‌سازیم، بلکه **فناوری را بومی می‌کنیم**.

امیدوارم این راهنمای عملی، الهام‌بخش شما برای ساخت تم‌هایی با روح ایرانی باشد — همان‌گونه که من سال‌هاست در مسیر آن گام برمی‌دارم.

فصل ۵: بهینه‌سازی عملکرد جوملا برای هاست‌های داخلی

یکی از بزرگ‌ترین چالش‌های توسعه‌دهندگان ایرانی، استقرار سایت‌های جوملا روی هاست‌های داخلی است. هرچند این هاست‌ها از نظر دسترسی، پشتیبانی و قیمت مزایایی دارند، اما معمولاً با محدودیت‌هایی مانند **PHP قدیمی**، **حافظه‌ی محدود**، **عدم پشتیبانی از Redis** و **محدودیت‌های mod_security** همراه هستند. در این مقاله، راهکارهایی عملی و تست‌شده را ارائه می‌دهم که بدون نیاز به سرور اختصاصی یا CDN خارجی، سرعت و عملکرد جوملا را روی هاست‌های ایرانی به‌طور چشمگیری بهبود می‌دهد.

گام ۱: تنظیمات بهینه در `configuration.php`

این فایل قلب تنظیمات جوملاست. برخی تغییرات کلیدی:

```
// غیرفعال کردن خطاها در محیط عمومی
public $error_reporting = 'none';

// فعال‌سازی Gzip
public $gzip = '1';

// استفاده از کش داخلی (فایل‌محور)
public $caching = '1';
public $cache_handler = 'file';
public $cachetime = '900'; // 15 دقیقه

// دیتابیس با کدینگ صحیح
public $dbtype = 'mysqli';
public $charset = 'utf8mb4';
public $sendmail = '/usr/sbin/sendmail';

// غیرفعال کردن آمارهای داخلی (در صورت عدم نیاز)
public $debug = '0';
```

گام ۲: بهینه‌سازی پایگاه‌داده بدون دسترسی به phpMyAdmin پیشرفته

در بسیاری از هاست‌های ایرانی، امکان اجرای کوئری‌های پیچیده محدود است. با این حال می‌توانید:

- هر هفته یک‌بار جداول را **Optimize** کنید (از طریق افزونه‌ی **Akeeba Admin Tools**).
- جدول‌های لاگ (مثل `#__user_logs` یا `#__joomla_updates`) را پاک کنید.
- از `utf8mb4_persian_ci` به‌جای `utf8_general_ci` برای جستجوی بهتر فارسی استفاده کنید.

گام ۳: کش هوشمند بدون Redis یا Memcached

چون بیشتر هاست‌های اشتراکی ایرانی از سیستم‌های کش پیشرفته پشتیبانی نمی‌کنند، باید از کش فایل‌محور بهینه‌تر استفاده کرد:

- در **System > Global Configuration > System**، گزینه‌ی **Cache** را روی **ON - Conservative caching** بگذارید.
- افزونه‌ی **JotCache** را نصب کنید — این افزونه حتی در هاست‌های ساده، با فشرده‌سازی صفحه و ذخیره‌ی فایل‌های HTML استاتیک، سرعت را تا ۳ برابر افزایش می‌دهد.
- برای صفحات پویا (مثل کامپوننت‌های سفارشی)، از کش دستی استفاده کنید:

```
$cache = JFactory::getCache('mycomponent');
$items = $cache->get('product_list');
```

```
if ($items === false) {
    $items = $this->loadProductsFromDB();
    $cache->store($items, 'product_list');
}
```

گام ۴: بهینه‌سازی تصاویر — بدون نیاز به سرور خارجی

تصاویر بیشترین بار را روی سرعت سایت می‌گذارند. راهکارهای داخلی:

- تمام تصاویر را قبل از آپلود با نرم‌افزارهایی مثل **Caesium** یا **TinyPNG دسکتاپ** فشرده کنید.
- از فرمت **WebP** استفاده کنید — اگر هاست پشتیبانی نمی‌کند، با افزونه‌ی **WebP Express** (سازگار با جوملا) آن را شبیه‌سازی کنید.
- برای تصاویر دکوراتیو (مثل بک‌گراندهای اسلیم)، از **SVG** یا **CSS Gradient** استفاده کنید.
- در تم خود، از `"loading="lazy` برای تصاویر غیرضروری استفاده کنید:

```
<img src="image.jpg" loading="lazy" alt="توضیح" />
```

گام ۵: کاهش درخواست‌ها با ادغام CSS و JS

هر افزونه‌ی جوملا فایل‌هایی اضافه می‌کند. برای کاهش درخواست‌ها:

- افزونه‌ی **JCH Optimize** را نصب و پیکربندی کنید. این افزونه CSS و JS را به‌صورت خودکار ترکیب، فشرده و به پایین صفحه منتقل می‌کند.
- فایل‌هایی که فقط در یک صفحه استفاده می‌شوند (مثل فایل‌های کامپوننت‌های سفارشی)، را به‌صورت شرطی لود کنید:

```
if (\Joomla\CMS\Factory::getApplication()->input->get('option') === 'com_mycomp') {
    \Joomla\CMS\HTML\HTMLHelper::_('script', 'com_mycomp/script.js', ['version' => 'auto', 'relative' =
}
```

- فونت‌های غیرضروری (مثل فونت‌های لاتین Google Fonts) را غیرفعال کنید.

گام ۶: دور زدن محدودیت‌های رایج هاست‌های ایرانی

برخی محدودیت‌های شایع و راه‌حل‌های عملی:

محدودیت	راه‌حل
`max_input_vars = 1000`	در `htaccess.` خط زیر را اضافه کنید: `php_value max_input_vars 3000`
خطای `Internal Error 500` پس از نصب	فایل `htaccess.` را موقتاً تغییر نام دهید یا خطوط `mod_security` را غیرفعال کنید.
عدم پشتیبانی از `allow_url_fopen`	در کامپوننت‌های سفارشی، به‌جای `()file_get_contents` از `curl` استفاده کنید.

تجربه‌ی شخصی: بهینه‌سازی سایت فرهنگی با ۴۰۰۰ بازدید روزانه

در یکی از پروژه‌های اخیر، سایتی با ساختار جوملا و هاست اشتراکی ایرانی داشتیم که با ۴۰۰۰ بازدید روزانه، به‌طور مداوم crash می‌کرد. با اعمال راهکارهای فوق — به‌ویژه استفاده از JotCache، حذف افزونه‌های غیرضروری و بهینه‌سازی پایگاه‌داده — میانگین بار سرور از ۴.۸ به ۰.۳ کاهش یافت و زمان بارگذاری صفحه از ۶ ثانیه به کمتر از ۱.۲ ثانیه رسید.

جمع‌بندی

سرعت جوملا روی هاست‌های ایرانی فقط به سرور بستگی ندارد؛ بیشتر به **هوش طراحی و مدیریت منابع** شما بستگی دارد. با رعایت اصول ساده‌ای مانند کش هوشمند، کاهش درخواست‌ها، بهینه‌سازی تصاویر و تنظیمات دقیق پایگاه‌داده، می‌توانید سایتی سریع، پایدار و کاربرپسند داشته باشید — حتی روی ساده‌ترین هاست اشتراکی.

یادتان باشد: سرعت، بخشی از تجربه‌ی کاربری است. و تجربه‌ی کاربری، بخشی از احترام شما به مخاطب فارسی‌زبان است.

فصل ۶: ایجاد یک کامپوننت چندزبانه در جوملا ۶

در عصر جهانی‌شدن، بسیاری از سایت‌های ایرانی نیازمند پشتیبانی از چند زبان — به‌ویژه **فارسی و انگلیسی** — هستند. اما نه فقط در سمت کاربر، بلکه **در پنل مدیریت نیز!** جوملا ۶ با معماری مدرن خود، این امکان را فراهم می‌کند، اما پیاده‌سازی آن نیازمند درک عمیق از سیستم زبان‌های داخلی جوملاست. در این مقاله، گام‌به‌گام نحوه‌ی ساخت یک **کامپوننت چندزبانه‌ی کامل** را آموزش می‌دهم که:

- محتوا را به‌صورت جداگانه برای هر زبان ذخیره می‌کند
- رابط کاربری عمومی و پنل مدیریت را به‌صورت خودکار ترجمه می‌کند
- سوییچ زبان هوشمند با حفظ URL و ساختار سئو دارد
- با چالش‌های RTL/LTR به‌خوبی کنار می‌آید

گام ۱: فعال‌سازی چندزبانه در جوملا

قبل از هر چیز، باید سیستم چندزبانه‌ی جوملا را فعال کنید:

1. از **Extensions > Language(s) > Install Languages**، زبان‌های `fa-IR` و `en-GB` را نصب کنید.
2. در **System > Manage > Plugins**، پلاگین **System - Language Filter** را فعال کنید.
3. در تنظیمات همان پلاگین، گزینه‌ی **Remove URL Language Code** را **NO** قرار دهید (برای حفظ ساختار سئویی).
4. در **Multilingual Associations**، گزینه‌ی **Enable Associations** را فعال کنید.

گام ۲: ساختار جدول پایگاه‌داده برای محتوای چندزبانه

دو رویکرد وجود دارد. من پیشنهاد می‌کنم از **رویکرد ستون‌محور** استفاده کنید (ساده‌تر و سریع‌تر برای جستجو):

```
CREATE TABLE `#__mycomp_articles` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `title_fa` varchar(255) NOT NULL,
  `title_en` varchar(255) NOT NULL,
  `introtext_fa` text,
  `introtext_en` text,
  `fulltext_fa` mediumtext,
  `fulltext_en` mediumtext,
  `published` tinyint(1) NOT NULL DEFAULT '1',
  PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_persian_ci;
```

این ساختار از JOINهای پیچیده جلوگیری می‌کند و عملکرد را در هاست‌های محدود ایرانی بهبود می‌بخشد.

گام ۳: سیستم زبان داخلی جوملا (Language Strings)

تمام متون رابط کاربری (نه محتوا!) باید با سیستم زبان جوملا مدیریت شوند. برای این کار:

در پوشه‌ی کامپوننت خود، فایل‌های زیر را ایجاد کنید:

```
com_mycomp/
├── language/
│   ├── fa-IR/
│   │   └── fa-IR.com_mycomp.ini
│   └── en-GB/
│       └── en-GB.com_mycomp.ini
```

محتوای `fa-IR.com_mycomp.ini` :

```
COM_MYCOMP="کامپوننت من"
COM_MYCOMP_ARTICLES="مقالات"
COM_MYCOMP_TITLE="عنوان مقاله"
COM_MYCOMP_PUBLISHED="منتشر شده"
```

و در `en-GB.com_mycomp.ini` :

```
COM_MYCOMP="My Component"
COM_MYCOMP_ARTICLES="Articles"
COM_MYCOMP_TITLE="Article Title"
COM_MYCOMP_PUBLISHED="Published"
```

سپس در کد PHP، همیشه از این توابع استفاده کنید:

```
<?php echo JText::_('COM_MYCOMP_ARTICLES'); ?>
```

گام ۴: تشخیص زبان فعلی و بارگذاری محتوای مربوطه

در مدل کامپوننت، زبان فعلی را تشخیص دهید و محتوای مناسب را بارگذاری کنید:

```
<?php
namespace Mycomp\Model;

use Joomla\CMS\Factory;
use Joomla\CMS\MVC\Model\ListModel;

class ArticlesModel extends ListModel
{
    public function getListQuery()
    {
        $lang = Factory::getLanguage()->getTag(); // مثلاً 'fa-IR' یا 'en-GB'
        $langPrefix = substr($lang, 0, 2); // 'fa' یا 'en'

        $db = $this->getDbo();
        $query = $db->getQuery(true)
            ->select($db->quoteName("title_{$langPrefix}", 'title'))
            ->select($db->quoteName("introtext_{$langPrefix}", 'introtext'))
            ->from($db->quoteName('#__mycomp_articles'))
            ->where($db->quoteName('published') . ' = 1');
```

```
        return $query;
    }
}
```

گام ۵: سوییچ زبان هوشمند در فرانت‌اند

یک ماژول ساده برای سوییچ زبان بسازید یا از ماژول داخلی **Language Switcher** استفاده کنید. اما برای کامپوننت‌های سفارشی، لینک‌ها را به‌صورت دستی ایجاد کنید:

```
<?php
$activeLang = Factory::getLanguage()->getTag();
$otherLang = ($activeLang === 'fa-IR') ? 'en-GB' : 'fa-IR';
$otherLangCode = substr($otherLang, 0, 2);
$url = JUri::current() . '?lang=' . $otherLangCode;
?>
<a href="<?php echo $url; ?>" class="lang-switch">
  <?php echo ($activeLang === 'fa-IR') ? 'EN' : 'فارسی'; ?>
</a>
```

⚠️ نکته: برای حفظ سئو، همیشه از تگ‌های `</ "..."=link rel="alternate" hreflang>` در `<head>` استفاده کنید.

گام ۶: مدیریت چالش‌های RTL/LTR

وقتی کاربر از فارسی به انگلیسی سوییچ می‌کند، جهت صفحه باید از RTL به LTR تغییر کند. این کار به‌صورت خودکار توسط جوملا انجام می‌شود، اما در کامپوننت‌های سفارشی:

- در فایل‌های ویو، از کلاس `direction` برای تعیین جهت استفاده نکنید؛ بلکه از کلاس‌های خنثی مثل `text-start` در Bootstrap یا `justify-content: flex-start` در CSS استفاده کنید.
- در CSS، به‌جای `left/right` از `inline-start/inline-end` استفاده کنید (در صورت پشتیبانی مرورگر).
- برای فرم‌ها، مطمئن شوید لیبل‌ها نسبت به زبان جابه‌جا می‌شوند.

گام ۷: پشتیبانی از ترجمه در پنل مدیریت

فرم ویرایش مقاله در پنل ادمین باید هر دو زبان را در یک صفحه نمایش دهد:

```
<!-- در فایل admin/models/forms/article.xml -->
<fieldset name="fa_fields" label="فارسی">
  <field name="title_fa" type="text" label="COM_MYCOMP_TITLE" />
  <field name="introtext_fa" type="editor" />
</fieldset>

<fieldset name="en_fields" label="English">
  <field name="title_en" type="text" label="COM_MYCOMP_TITLE" />
  <field name="introtext_en" type="editor" />
</fieldset>
```

این روش، به مدیر سایت اجازه می‌دهد هر دو نسخه را همزمان ویرایش کند — بدون نیاز به سوییچ زبان در پنل مدیریت.

تجربه‌ی شخصی: سایت فرهنگی دوزبانه برای معرفی خط ثلث

در یکی از پروژه‌های شخصی، سایتی برای معرفی خط ثلث ایرانی به مخاطبان جهانی ساختم. با استفاده از همین روش، تمام آموزش‌ها و نمونه‌کارها هم به فارسی و هم به انگلیسی در دسترس بودند. حتی دکمه‌های «ادامه مطلب» و «بازگشت» در هر زبان، با فونت و جهت مناسب نمایش داده می‌شدند. این سایت امروز یکی از منابع معتبر برای علاقه‌مندان جهانی به خوشنویسی اسلامی است.

جمع‌بندی

چندزبانه‌سازی در جوملا فقط افزودن زبان نیست؛ یک فرآیند فرهنگی-فنی است که نیازمند توجه به جزئیات زیادی است — از تشخیص زبان گرفته تا مدیریت جهت‌نمایی و ساختار URL. اما با رعایت اصولی که در این مقاله گفته شد، می‌توانید سایتی بسازید که هم برای مخاطب ایرانی گرامی دارد و هم برای جهانیان در دسترس است.

چرا که هنر و دانش ایرانی، مرز نمی‌شناسد — و نباید وب‌سایت‌های ما هم مرز ببندند.

فصل ۷: جوملا ۶ + هوش مصنوعی (AI)

هوش مصنوعی دیگر یک آینده‌نگری نیست؛ امروزه بخشی جدایی‌ناپذیر از توسعه‌ی وب است. اما چگونه می‌توانیم این قدرت را در یک سیستم مدیریت محتوای متن‌باز و ساختاریافته مانند **جوملا ۶** به‌کار بگیریم؟ در این مقاله، راهکارهایی عملی ارائه می‌دهم که نه‌تنها محتوای سایت شما را غنی‌تر می‌کند، بلکه تجربه‌ی کاربری را به‌طور هوشمندانه شخصی‌سازی می‌کند — و همه‌چیز را با استانداردهای امنیتی و سازگاری با زیرساخت‌های ایرانی پیاده‌سازی می‌کند.

چرا جوملا ۶ و هوش مصنوعی؟

جوملا ۶ با معماری مدرن خود (مبتنی بر PHP 8.1+ و لایه‌بندی MVC)، بستری ایده‌آل برای ادغام با APIهای خارجی — از جمله سرویس‌های هوش مصنوعی — فراهم می‌کند. برخلاف وردپرس که بر پایه‌ی هاک (hook) ساخته شده، جوملا اجازه می‌دهد کامپوننت‌هایی بسازیم که به‌صورت شفاف و ایمن با مدل‌های AI تعامل داشته باشند.

گام ۱: انتخاب سرویس هوش مصنوعی مناسب برای محیط ایرانی

برای کاهش تأخیر و رعایت مسائل امنیتی، پیشنهاد می‌کنم از سرویس‌های داخلی استفاده کنید:

- **talkbot.ir:** یکی از قدرتمندترین پلتفرم‌های AI فارسی‌زبان برای پردازش زبان طبیعی (NLP)
- **سرویس‌های مبتنی بر مدل‌های باز (مثل DeepSeek یا Gemma) روی سرورهای ایرانی:** اگر به داده‌های حساس دسترسی دارید
- **APIهای سفارشی با لایه‌ی اعتبارسنجی:** برای پروژه‌های درون‌سازمانی

گام ۲: ساخت یک کامپوننت هوشمند برای تولید خلاصه‌ی مقالات

فرض کنید می‌خواهید برای هر مقاله‌ی بلند، یک خلاصه‌ی هوشمند به زبان فارسی تولید کنید. در مدل کامپوننت:

```php
<?php
namespace MyAI\Model;

use Joomla\CMS\Factory;
use Joomla\CMS\Http\HttpFactory;

class Summarizer
{
    public function generateSummary($text)
    {
        $apiKey = 'YOUR_TALKBOT_API_KEY';
        $url = 'https://api.talkbot.ir/v1/summarize';

        $data = json_encode([
            'text' => $text,
            'language' => 'fa',
            'max_length' => 150
        ]);

        $options = [
            'headers' => [
                'Content-Type' => 'application/json',
                'Authorization' => 'Bearer ' . $apiKey
            ]
```

```
        ];

        $http = HttpFactory::getHttp($options);
        $response = $http->post($url, $data);

        if ($response->code == 200) {
            $result = json_decode($response->body, true);
            return $result['summary'] ?? '';
        }

        return '';
    }
}
```

گام ۳: ذخیره‌سازی هوشمند خروجی‌های AI برای کاهش هزینه و افزایش سرعت

هر بار فراخوانی API هزینه و تأخیر دارد. بنابراین:

- خروجی را در یک ستون جداگانه در جدول مقالات ذخیره کنید: `summary_ai TEXT`
- فقط در صورتی که خلاصه وجود نداشت، API را فراخوانی کنید
- برای به‌روزرسانی، دکمه‌ی «تولید مجدد خلاصه هوشمند» در پنل ادمین اضافه کنید

گام ۴: ساخت چت‌بات پاسخ‌گو بر اساس محتوای سایت

برخلاف چت‌بات‌های عمومی، می‌توانید چت‌باتی بسازید که **فقط از محتوای سایت شما** یاد می‌گیرد:

1. همه‌ی مقالات، صفحات و محصولات را به‌صورت زوج‌های «سؤال-پاسخ» پیش‌پردازش کنید
2. یک جدول `site_knowledge__#` ایجاد کنید:

```
CREATE TABLE `#__site_knowledge` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `question` varchar(255) NOT. NULL,
  `answer` text,
  `source_url` varchar(255),
  PRIMARY KEY (`id`),
  FULLTEXT KEY `ft_question` (`question`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

3. در زمان نصب کامپوننت، یک cron job (یا task جوملا) برای به‌روزرسانی دانش اضافه کنید
4. در فرانت‌اند، با AJAX سؤال کاربر را به سرور ارسال کنید و با `MATCH ... AGAINST` نزدیک‌ترین پاسخ را پیدا کنید

گام ۵: تولید خودکار متا تگ‌ها و توضیحات برای سئوی فارسی

سئوی فارسی نیازمند متا تگ‌های طبیعی و مرتبط است. با هوش مصنوعی می‌توانید:

```
// در کنترلر ذخیره‌ی مقاله
if (empty($article->metadesc)) {
    $ai = new AIGenerator();
    $article->metadesc = $ai->generateMetaDescription($article->introtext);
}
```

این کار، نه‌تنها زمان مدیر را ذخیره می‌کند، بلکه کیفیت سئو را به‌طور چشمگیری بهبود می‌بخشد.

گام ۶: امنیت و اعتبارسنجی خروجی‌های هوش مصنوعی

هرگز خروجی AI را بدون بررسی در سایت نمایش ندهید!

- همیشه از `htmlspecialchars($output, ENT_QUOTES, 'UTF-8')` برای جلوگیری از XSS استفاده کنید
- در سرور، فیلتری برای حذف محتوای نامناسب (مثل لینک‌های خارجی یا کلمات حساس) پیاده‌سازی کنید
- برای پروژه‌های آموزشی یا آزمون، خروجی را ابتدا به ادمین نمایش دهید و اجازه‌ی انتشار دستی بدهید

تجربه‌ی شخصی: سیستم آزمون هوشمند با تولید خودکار سؤالات

در یکی از پروژه‌های اخیر، سیستمی ساختم که از محتوای آموزشی سایت، به‌صورت خودکار سؤالات چندگزینه‌ای تولید می‌کرد. این سؤالات سپس توسط ادمین بررسی و به بانک سؤال اضافه می‌شدند. این روش، زمان تولید آزمون را از چندین ساعت به چند دقیقه کاهش داد — و همه‌چیز در چارچوب امنیتی جوملا ۶ و با استفاده از API داخلی انجام شد.

جمع‌بندی

ادغام هوش مصنوعی با جوملا ۶، فقط یک افزونه نیست؛ یک **تغییر پارادایم** است. شما دیگر فقط محتوا نمی‌سازید — بلکه یک **محیط هوشمند** می‌سازید که یاد می‌گیرد، پاسخ می‌دهد و رشد می‌کند. و مهم‌تر از همه، این کار را می‌توانید با رعایت اصول امنیتی، پشتیبانی از فارسی و سازگاری با زیرساخت‌های ایرانی انجام دهید.

هوش مصنوعی، هنر را جایگزین نمی‌شود — اما هنرمندان هوشمند را پیروز می‌کند.

فصل ۸: تم‌سازی مدرن در جوملا ۶ با Tailwind CSS

در دنیای امروز طراحی وب، **Bootstrap** دیگر تنها گزینه نیست. **Tailwind CSS** — فریم‌ورک کلاس‌محور و سبک‌وزن — انقلابی در نحوه‌ی ساخت رابط‌های کاربری ایجاد کرده است. اما آیا می‌توان از Tailwind در یک سیستم مدیریت محتوای ساختاریافته مثل **جوملا ۶** استفاده کرد؟ پاسخ مثبت است — و حتی می‌توان آن را به‌گونه‌ای پیاده‌سازی کرد که برای سایت‌های فارسی‌زبان، سبک‌تر، زیباتر و انعطاف‌پذیرتر از همیشه باشد. در این مقاله، گام‌به‌گام نحوه‌ی ساخت یک تم کاملاً مدرن با Tailwind CSS در جوملا ۶ را آموزش می‌دهم.

چرا Tailwind CSS برای جوملا ۶؟

- **کاهش حجم نهایی:** با PurgeCSS، فقط کلاس‌هایی که استفاده می‌کنید در فایل نهایی باقی می‌مانند (معمولاً زیر ۱۰ کیلوبایت!)
- **سرعت توسعه:** نیازی به نوشتن CSS نیست — همه‌چیز با کلاس‌های از پیش‌ساخته انجام می‌شود
- **هماهنگی کامل با RTL:** Tailwind از نسخه‌ی ۳ به‌صورت داخلی از RTL پشتیبانی می‌کند
- **انعطاف‌پذیری بی‌نظیر:** هر پیکسل از رابط کاربری را می‌توان بدون شکستن ساختار تغییر داد

گام ۱: ساختار پایه‌ی تم جوملا با پشتیبانی از Tailwind

ابتدا ساختار استاندارد تم جوملا را ایجاد کنید:

```
/templates/mytemplate/
├── css/
├── js/
├── language/
├── index.php
├── templateDetails.xml
└── tailwind.config.js
```

گام ۲: نصب و تنظیم Tailwind CSS (بدون نیاز به Node.js در هاست!)

بسیاری فکر می‌کنند Tailwind نیاز به Node.js در سرور دارد — اما این‌طور نیست! تمام کار را در محیط توسعه انجام دهید و فقط فایل نهایی را آپلود کنید:

1. در کامپیوتر محلی خود:

```
npm init -y
npm install -D tailwindcss postcss autoprefixer
npx tailwindcss init -p
```

2. فایل `tailwind.config.js` را تنظیم کنید:

```
module.exports = {
  content: ["./index.php", "./html/**/*.php"],
  theme: {
    extend: {},
  },
  plugins: [],
  corePlugins: {
    preflight: false, // جلوگیری از تداخل با استایل‌های جوملا
  },
```

```
}
```

3. فایل `./css/tailwind.css` را ایجاد کنید:

```
@tailwind base;
@tailwind components;
@tailwind utilities;
```

4. فایل نهایی را بسازید:

```
npx tailwindcss -i ./css/tailwind.css -o ./css/style.css --minify
```

5. فایل `style.css` را در پوشه‌ی `/css/` تم آپلود کنید.

گام ۳: فعال‌سازی پشتیبانی RTL در Tailwind

در فایل `index.php` تم، جهت رابط را به‌صورت پویا تنظیم کنید:

```
<?php
defined('_JEXEC') or die;
$direction = JFactory::getDocument()->getDirection(); // 'rtl' یا 'ltr'
?>
<!DOCTYPE html>
<html dir="<?php echo $direction; ?>" lang="<?php echo JFactory::getDocument()->getLanguage(); ?>">
<head>
  <?php echo $this->head; ?>
  <link rel="stylesheet" href="<?php echo $this->baseurl; ?>/templates/<?php echo $this->template; ?>/css/s
</head>
```

سپس در RTL، `tailwind.config.js` را فعال کنید:

```
module.exports = {
  content: ["./index.php", "./html/**/*.php"],
  important: true, // اولویت بالاتر از استایل‌های جوملا
  theme: {
    extend: {},
  },
  variants: {
    extend: {},
  },
  plugins: [],
  corePlugins: {
    preflight: false,
  },
}
```

⚠️ نکته: Tailwind به‌طور خودکار کلاس‌های `:ltr` و `:rtl` را تولید می‌کند. مثلاً:

```
<div class="rtl:text-right ltr:text-left">متن شما</div>
```

گام ۴: طراحی کامپوننت‌های UI با Tailwind (بدون نیاز به CSS سفارشی)

مثال: کارت مقاله

```
<div class="bg-white rounded-xl shadow-md overflow-hidden hover:shadow-lg transition-shadow duration-300">
  <img src="<?php echo $item->image; ?>" alt="<?php echo $item->title; ?>" class="w-full h-48 object-cover"
  <div class="p-5">
    <h3 class="text-xl font-bold text-gray-800 mb-2"><?php echo $item->title; ?></h3>
    <p class="text-gray-600 line-clamp-2"><?php echo $item->introtext; ?></p>
    <a href="<?php echo $item->link; ?>" class="mt-3 inline-block text-blue-600 hover:text-blue-800 font-me
      ادامه مطلب ←
    </a>
  </div>
</div>
```

مثال: نوار منو واکنش‌گرا

```
<nav class="bg-gray-800 text-white">
  <div class="max-w-7xl mx-auto px-4 sm:px-6 lg:px-8">
    <div class="flex items-center justify-between h-16">
      <div class="flex items-center">
        <a href="/" class="text-xl font-semibold">سایت من</a>
      </div>
      <div class="hidden md:block">
        <div class="ml-10 flex items-baseline space-x-4 rtl:space-x-reverse">
          <a href="/about" class="px-3 py-2 rounded-md text-sm font-medium hover:bg-gray-700">درباره ما</a>
        </div>
      </div>
    </div>
  </div>
</nav>
```

گام ۵: بهینه‌سازی نهایی با PurgeCSS (حذف کدهای اضافی)

بدون این مرحله، فایل CSS شما چند مگابایت خواهد بود! اما Tailwind به‌صورت داخلی این کار را انجام می‌دهد — فقط مطمئن شوید که در `content` تمام فایل‌هایی که از کلاس‌ها استفاده می‌کنند را لیست کرده‌اید:

```
content: [
  "./index.php",
  "./html/**/*.php",
  "./language/**/*.ini"
]
```

پس از ساخت، حجم فایل `style.css` معمولاً بین **۵ تا ۱۵ کیلوبایت** است — حتی برای تم‌های پیچیده!

گام ۶: جلوگیری از تداخل با استایل‌های داخلی جوملا

جوملا استایل‌هایی برای فرم‌ها، جدول‌ها و مدال‌ها اعمال می‌کند. برای جلوگیری از تداخل:

- در `corePlugins.preflight = false` ، `tailwind.config.js` را تنظیم کنید (همان‌طور که بالاتر گفته شد)

- در فایل `index.php` ، استایل‌های جوملا را بعد از Tailwind لود کنید:

```
<?php echo $this->head; ?>
<link rel="stylesheet" href=".../style.css" />
```

- برای فرم‌های ادمین، از کلاس‌های Tailwind استفاده نکنید — فقط در فرانت‌اند به‌کار ببرید

تجربه‌ی شخصی: ساخت تم «نسترن» برای یک سایت آموزش خوشنویسی

در یکی از پروژه‌های اخیر، تمی با الهام از گل نسترن طراحی کردم که تمام رابط آن با Tailwind ساخته شده بود. حجم نهایی CSS فقط **۸.۲ کیلوبایت** بود و بار اولیه‌ی سایت در Google PageSpeed امتیاز **۹۸** گرفت. همچنین، تغییر رنگ‌ها یا چینش عناصر — که قبلاً چند ساعت طول می‌کشید — حالا در کمتر از ۵ دقیقه انجام می‌شد.

جمع‌بندی

Tailwind CSS و جوملا ۶، ترکیبی عالی هستند — به‌شرطی که به‌درستی پیاده‌سازی شوند. با این روش، شما دیگر وابسته به تم‌های غربی یا فریم‌ورک‌های سنگین نیستید. بلکه تمی می‌سازید که:

- سبک‌وزن و سریع است
- کاملاً با فارسی و RTL سازگار است
- بدون نیاز به دانش CSS پیشرفته، قابل توسعه است
- با هویت بصری شما هماهنگ می‌شود

چرا که در دنیای طراحی وب، آزادی یعنی کنترل — و Tailwind، کنترل را به دست شما می‌دهد.

فصل ۹: امنیت پیشرفته در جوملا ۶

یک سایت جوملا تنها زمانی موفق است که **امن** باشد. اما امنیت در هاست‌های اشتراکی ایرانی — با محدودیت‌های دسترسی، عدم پشتیبانی از فایروال‌های پیشرفته و قدیمی‌بودن برخی ماژول‌ها — چالشی بزرگ محسوب می‌شود. در این مقاله، راهکارهایی عملی، تست‌شده و قابل اجرا ارائه می‌دهم که نه‌تنها سایت عمومی شما را محافظت می‌کند، بلکه برای پروژه‌های حساسی مانند **سیستم‌های آزمون آنلاین**، **فرم‌های استخدام** و **پنل‌های مدیریت داخلی** نیز امنیت لازم را فراهم می‌کند — بدون نیاز به سرور اختصاصی یا دانش سیستمی پیشرفته.

گام ۱: تنظیمات امنیتی ضروری در `configuration.php`

این تنظیمات ساده، خط دفاعی اول شما هستند:

```
// جلوگیری از نمایش خطاهای سرور
public $error_reporting = 'none';

// مسیر جداگانه برای فایل‌های لاگ (خارج از public_html)
public $log_path = '/home/youruser/logs';

// غیرفعال کردن امکان اجرای اسکریپت‌های PHP در پوشه‌های بارگذاری
// (با افزونه‌های امنیتی – گام بعدی)

// تغییر پیشوند جداول برای سخت‌تر کردن حملات
public $dbprefix = 'bal_'; // نه 'jos_'
```

گام ۲: فعال‌سازی و تنظیم افزونه‌های داخلی امنیتی

جوملا ۶ افزونه‌های امنیتی داخلی دارد که باید فعال شوند:

- **System - Remember Me:** غیرفعال کنید (در صورت عدم نیاز)
- **Authentication - Joomla:** تنها روش لاگین را از این طریق نگه دارید
- **Content - Email Cloaking:** فعال باشد تا اسپم‌ربات‌ها ایمیل‌ها را جمع‌آوری نکنند

همچنین، افزونه‌ی **Akeeba Admin Tools** را نصب کنید — حتی در هاست‌های اشتراکی، لایه‌ای قوی از امنیت اضافه می‌کند.

گام ۳: جلوگیری از XSS و SQL Injection در کامپوننت‌های سفارشی

در هر کامپوننتی که می‌سازید، این قوانین را رعایت کنید:

ورودی‌ها را همیشه فیلتر کنید:

```
$input = \Joomla\CMS\Factory::getApplication()->input;
$id = $input->getInt('id', 0); // فقط عدد صحیح
$title = $input->getString('title', ''); // رشته – ولی بدون اسکریپت!
```

خروجی‌ها را همیشه Escape کنید:

```
<?php echo htmlspecialchars($title, ENT_QUOTES, 'UTF-8'); ?>
```

در کوئری‌ها از quoting استفاده کنید:

```
$db = \Joomla\CMS\Factory::getDbo();
$query = $db->getQuery(true)
    ->select('*')
    ->from($db->quoteName('#__mytable'))
    ->where($db->quoteName('id') . ' = ' . (int)$id);
```

گام ۴: ایجاد سیستم لاگین هوشمند با OTP (با استفاده از sms.ir یا api.ir)

برای پنل‌های حساس (مثل سیستم آزمون)، لاگین دو مرحله‌ای ضروری است:

```
// در پلاگین سیستمی یا کامپونت
$phone = $user->get('phone');
$code = rand(100000, 999999);

// ذخیره‌ی موقت کد در session
\Joomla\CMS\Factory::getApplication()->setUserState('otp.code', $code);
\Joomla\CMS\Factory::getApplication()->setUserState('otp.expires', time() + 300);

// ارسال با sms.ir
$url = 'https://api.sms.ir/v1/send';
$headers = [
    'Content-Type: application/json',
    'x-api-key: YOUR_SMS_IR_KEY'
];
$data = json_encode([
    'mobile' => $phone,
    'message' => "کد تأیید شما: {$code} — معتبر تا ۵ دقیقه"
]);

$curl = curl_init();
curl_setopt($curl, CURLOPT_URL, $url);
curl_setopt($curl, CURLOPT_HTTPHEADER, $headers);
curl_setopt($curl, CURLOPT_POST, true);
curl_setopt($curl, CURLOPT_POSTFIELDS, $data);
curl_setopt($curl, CURLOPT_RETURNTRANSFER, true);
$response = curl_exec($curl);
curl_close($curl);
```

این روش، امنیت لاگین را حتی در صورت نشت رمز عبور، به‌طور چشمگیری افزایش می‌دهد.

گام ۵: جلوگیری از تقلب در سیستم‌های آزمون — تشخیص تغییر تب و کپی‌پیست

این ویژگی‌ها را در صفحه‌ی آزمون با JavaScript پیاده‌سازی کنید:

```
<script>
let tabSwitchCount = 0;
let copyAttempt = 0;

// تشخیص تغییر تب
document.addEventListener('visibilitychange', function() {
```

```
  if (document.hidden) {
    tabSwitchCount++;
    if (tabSwitchCount >= 2) {
      alert('تغییر تب مجاز نیست. آزمون شما لغو خواهد شد.');
      // ارسال درخواست به سرور برای ثبت تقلب
      fetch('index.php?option=com_exam&task=logViolation', {
        method: 'POST',
        headers: { 'Content-Type': 'application/json' },
        body: JSON.stringify({ violation: 'tab_switch' })
      }).then(() => window.location.href = '/exam/terminated');
    }
  }
});

// جلوگیری از کپی‌پیست
document.addEventListener('copy', function(e) {
  copyAttempt++;
  if (copyAttempt >= 1) {
    e.preventDefault();
    alert('کپی محتوا مجاز نیست.');
  }
});

document.addEventListener('contextmenu', function(e) {
  e.preventDefault(); // غیرفعال کردن راست‌کلیک
});
</script>
```

⚠️ نکته: این کدها را با یک اسکریپت مینیفای‌شده و رندوم‌سازی‌شده در هر بار بارگذاری ارائه دهید تا دور زدن آن سخت‌تر شود.

گام ۶: فعال‌سازی Content Security Policy (CSP) حتی در هاست‌های اشتراکی

CSP می‌تواند اجرای اسکریپت‌های مخرب را جلوگیری کند. در فایل `.htaccess` :

```
<IfModule mod_headers.c>
  Header set Content-Security-Policy "default-src 'self'; script-src 'self' 'unsafe-inline' https://api.sms
</IfModule>
```

اگر هاست شما از `mod_headers` پشتیبانی نمی‌کند، این هدر را در `index.php` اصلی جوملا اضافه کنید:

```
<?php
header("Content-Security-Policy: default-src 'self'; script-src 'self' 'unsafe-inline'; style-src 'self' 'u
// سپس include پوشه‌ی libraries را ادامه دهید
require_once 'libraries/bootstrap.php';
...
```

گام ۷: محدودیت دسترسی بر اساس IP یا نقش کاربری

برای پنل‌های داخلی (مثل سیستم استخدام)، دسترسی را محدود کنید:

```
// در کنترلر کامپونت
$user = \Joomla\CMS\Factory::getUser();
$allowedGroups = [8, 9]; // ID گروه‌های مجاز

if (!in_array($user->id, $allowedGroups)) {
    \Joomla\CMS\Factory::getApplication()->enqueueMessage('دسترسی غیرمجاز', 'error');
    \Joomla\CMS\Factory::getApplication()->redirect('index.php');
}
```

تجربه‌ی شخصی: سیستم آزمون امن برای یک مرکز آموزشی در سیرجان

در یکی از پروژه‌های اخیر، سیستم آزمونی طراحی کردم که:

- لاگین با OTP از طریق sms.ir
- تشخیص تغییر تب و محدودیت کپی
- ذخیره‌سازی فعالیت‌های کاربر به‌صورت زنده
- گزارش تخلفات به ادمین با ایمیل و پیامک

این سیستم در یک هاست اشتراکی ایرانی راه‌اندازی شد و در طول یک سال، هیچ مورد تقلبی گزارش نشد — حتی با وجود هزاران شرکت‌کننده.

جمع‌بندی

امنیت در جوملا ۶ یک ویژگی نیست؛ یک فرآیند است. با رعایت اصول ساده‌ای که در این مقاله گفته شد — از تنظیمات پایه گرفته تا جلوگیری از تقلب در آزمون‌ها — می‌توانید سایتی بسازید که نه‌تنها زیبا و کاربردی است، بلکه **سپری در برابر تهدیدات دنیای دیجیتال** نیز هست.

یادتان باشد: امنیت، احترام به کاربر است. چون داده‌هایش را گران می‌دانید.

فصل ۱۰: ساخت چت‌بات هوشمند برای سایت جوملا

چت‌بات‌های عمومی — مانند آن‌هایی که فقط جواب «سلام» و «ممنون» می‌دهند — دیگر کافی نیستند. کاربران امروز انتظار دارند که دستیار مجازی شما **از محتوای سایت شما بداند:** از قیمت محصولات گرفته تا مراحل ثبت‌نام در آزمون‌ها، از آدرس دفتر گرفته تا زمان‌بندی دوره‌های آموزشی. در این مقاله، گام‌به‌گام نحوه‌ی ساخت یک **چت‌بات هوشمند و شخصی‌سازی‌شده** را آموزش می‌دهم که:

- به‌صورت خودکار از تمام محتوای سایت جوملا یاد می‌گیرد
- با هوش مصنوعی فارسی‌زبان (مثل Talkbot.ir) ادغام می‌شود
- تمام داده‌ها در همان سرور سایت ذخیره می‌شوند (امنیت بالا)
- رابط کاربری آن با Tailwind CSS و AJAX ساخته می‌شود — بدون رفرش صفحه

گام ۱: طراحی معماری سیستم — سه لایه‌ی هوشمند

چت‌بات ما بر سه لایه استوار است:

1. **لایه‌ی جمع‌آوری دانش:** اسکن خودکار محتوای سایت (مقالات، صفحات، محصولات)
2. **لایه‌ی پردازش هوشمند:** ادغام با API هوش مصنوعی برای درک سؤالات
3. **لایه‌ی رابط کاربری:** چت‌بات واکنش‌گرا در گوشه‌ی سایت

گام ۲: ایجاد جدول دانش (`#__site_knowledge`)

تمام دانش سایت را در یک جدول ساده ذخیره کنید:

```
CREATE TABLE `#__site_knowledge` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `question` varchar(255) NOT NULL,
  `answer` text NOT NULL,
  `source_url` varchar(255) DEFAULT NULL,
  `category` varchar(50) DEFAULT 'general',
  PRIMARY KEY (`id`),
  FULLTEXT KEY `ft_question` (`question`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_persian_ci;
```

گام ۳: جمع‌آوری خودکار دانش از محتوای جوملا

یک کامپوننت یا ماژول بنویسید که هنگام نصب یا با دستور دستی، محتوا را اسکن کند:

```
<?php
$db = \Joomla\CMS\Factory::getDbo();

// اسکن مقالات
$articles = $db->setQuery(
    $db->getQuery(true)
        ->select('id, title, introtext, alias')
        ->from('#__content')
        ->where('state = 1')
)->loadObjectList();

foreach ($articles as $article) {
```

```
    $questions = [
        "مقاله {$article->title} چیست؟",
        "درباره {$article->title} بگو",
        "{$article->title}"
    ];

    foreach ($questions as $q) {
        $db->setQuery(
            $db->getQuery(true)
                ->insert('#__site_knowledge')
                ->columns('question, answer, source_url, category')
                ->values(
                    $db->quote($q) . ', ' .
                    $db->quote($article->introtext) . ', ' .
                    $db->quote('/article/' . $article->alias) . ', ' .
                    $db->quote('article')
                )
        )->execute();
    }
}
```

این فرآیند را برای صفحات سایت، محصولات (در صورت استفاده از کامپوننت فروشگاهی) و سؤالات متداول نیز تکرار کنید.

گام ۴: ادغام با هوش مصنوعی فارسی (Talkbot.ir)

در سرور، یک سرویس هوشمند برای پردازش سؤالات ایجاد کنید:

```
<?php
class ChatbotService
{
    public function getResponse($userQuestion)
    {
        // ابتدا در دانش داخلی جستجو کن
        $internal = $this->searchInternal($userQuestion);
        if ($internal) return $internal;

        // اگر پیدا نشد، از هوش مصنوعی کمک بگیر
        return $this->askAI($userQuestion);
    }

    private function searchInternal($question)
    {
        $db = \Joomla\CMS\Factory::getDbo();
        $query = $db->getQuery(true)
            ->select('answer, source_url')
            ->from('#__site_knowledge')
            ->where("MATCH(question) AGAINST (" . $db->quote($question) . " IN NATURAL LANGUAGE MODE)")
            ->setLimit(1);
        $result = $db->setQuery($query)->loadObject();

        if ($result) {
```

```
            return $result->answer . ' [منبع]';
        }
        return false;
    }

    private function askAI($question)
    {
        $apiKey = 'YOUR_TALKBOT_API_KEY';
        $url = 'https://api.talkbot.ir/v1/chat';
        $data = json_encode(['message' => $question, 'context' => 'پاسخ بده فقط درباره سایت balvardi.ir']);

        $curl = curl_init();
        curl_setopt_array($curl, [
            CURLOPT_URL => $url,
            CURLOPT_RETURNTRANSFER => true,
            CURLOPT_HTTPHEADER => [
                'Content-Type: application/json',
                'Authorization: Bearer ' . $apiKey
            ],
            CURLOPT_POST => true,
            CURLOPT_POSTFIELDS => $data
        ]);
        $response = curl_exec($curl);
        curl_close($curl);

        $result = json_decode($response, true);
        return $result['response'] ?? 'متأسفم، نمی‌توانم به این سؤال پاسخ دهم.';
    }
}
```

گام ۵: رابط کاربری چت‌بات با Tailwind CSS و AJAX

یک ماژول ساده برای نمایش چت‌بات در همه‌ی صفحات ایجاد کنید:

```
<div id="chatbot-container" class="fixed bottom-6 left-6 z-50">
  <button id="chatbot-toggle" class="bg-blue-600 text-white rounded-full w-14 h-14 shadow-lg hover:bg-blue-
    💬
  </button>

  <div id="chatbot-window" class="hidden mt-2 w-80 bg-white rounded-xl shadow-xl overflow-hidden">
    <div class="bg-blue-600 text-white p-3 font-bold">دستیار هوشمند</div>
    <div id="chat-messages" class="h-64 overflow-y-auto p-3 bg-gray-50"></div>
    <div class="p-3 border-t">
      <input type="text" id="chat-input" class="w-full px-3 py-2 border rounded-lg" placeholder="خود را بپرسید
      <button id="chat-send" class="mt-2 w-full bg-blue-600 text-white py-2 rounded-lg hover:bg-blue-700">ا
    </div>
  </div>
</div>

<script>
document.getElementById('chatbot-toggle').addEventListener('click', function() {
```

```
  const win = document.getElementById('chatbot-window');
  win.classList.toggle('hidden');
});

document.getElementById('chat-send').addEventListener('click', sendMessage);
document.getElementById('chat-input').addEventListener('keypress', function(e) {
  if (e.key === 'Enter') sendMessage();
});

function sendMessage() {
  const input = document.getElementById('chat-input');
  const question = input.value.trim();
  if (!question) return;

  // نمایش سؤال کاربر
  addMessage(question, 'user');
  input.value = '';

  // ارسال به سرور
  fetch('index.php?option=com_chatbot&task=ask&format=json', {
    method: 'POST',
    headers: { 'Content-Type': 'application/json' },
    body: JSON.stringify({ question: question })
  })
  .then(res => res.json())
  .then(data => {
    addMessage(data.response, 'bot');
  });
}

function addMessage(text, sender) {
  const box = document.getElementById('chat-messages');
  const div = document.createElement('div');
  div.className = sender === 'user' ? 'text-right mb-2' : 'text-left mb-2';
  div.innerHTML = `
    <div class="${sender === 'user' ? 'bg-blue-100 text-gray-800' : 'bg-gray-200'} inline-block px-3 py-1 r
      ${text}
    </div>
  `;
  box.appendChild(div);
  box.scrollTop = box.scrollHeight;
}
</script>
```

گام ۶: امنیت و محدودیت‌های لازم

برای جلوگیری از سوءاستفاده:

- در سرور، تعداد درخواست‌ها را محدود کنید (مثلاً ۱۰ پیام در دقیقه)
- خروجی‌های هوش مصنوعی را با `htmlspecialchars` فیلتر کنید
- در صورت نیاز، گزینه‌ی «ارتباط با ادمین» را اضافه کنید:

```
<button onclick="contactAdmin()">با مدیر سایت صحبت کنید</button>
```

تجربه‌ی شخصی: چت‌بات برای سایت آموزش خوشنویسی

در سایت شخصی‌ام (balvardi.ir)، چت‌باتی راه‌اندازی کردم که به سؤالاتی مثل «چگونه خط ثلث بنویسم؟» یا «دوره‌های آموزشی چه زمانی شروع می‌شوند؟» پاسخ می‌دهد. این چت‌بات نه‌تنها ترافیک تماس‌های تلفنی را ۶۰٪ کاهش داد، بلکه زمان پاسخ‌دهی به کاربران را به‌صورت نامحدود افزایش داد — حتی در نیمه‌های شب!

جمع‌بندی

یک چت‌بات هوشمند، فقط یک جعبه‌ی چت نیست؛ **نماینده‌ی دیجیتالی هوش و دانش سایت شماست**. با این روش، شما دیگر نیازی به افزونه‌های گران‌قیمت یا سرویس‌های خارجی ندارید. بلکه چت‌باتی می‌سازید که:

- کاملاً متعلق به شماست
- فقط از سایت شما می‌داند
- با زبان فارسی و فرهنگ ایرانی هماهنگ است
- در هاست‌های ایرانی به‌راحتی اجرا می‌شود

چرا که بهترین دستیار، کسی است که از کارش سردرآورد — و کار شما، همین سایت است.

منابع و تشکر

این کتاب بر اساس تجربه‌های چندین ساله‌ی نویسنده در زمینه‌ی طراحی وب، خوشنویسی و توسعه‌ی سیستم‌های هوشمند تألیف شده است. برخی از منابع فنی و الهام‌بخش عبارتند از:

- مستندات رسمی جوملا (docs.joomla.org)
- پروژه‌ی Vazirmatn — فونت فارسی متن‌باز (github.com/rastikerdar/vazirmatn)
- سرویس هوش مصنوعی Talkbot.ir (talkbot.ir)
- مستندات Tailwind CSS (tailwindcss.com)
- تجربه‌های شخصی در پروژه‌های وب ایرانی، از جمله سیستم‌های آزمون، سایت‌های فرهنگی و پلتفرم‌های آموزشی

همچنین از همه‌ی افرادی که در طول سال‌ها به من الهام داده‌اند — از استادان خوشنویسی گرفته تا همکاران توسعه‌دهنده — صمیمانه سپاسگزارم.

این کتاب به‌صورت **رایگان و متن‌باز** منتشر شده است و هرگونه استفاده غیرتجاری با ذکر منبع (balvardi.ir) آزاد است.

© ۱۴۰۴ — روح‌الله بلوردی

balvardi.ir