

DIGITAL STUDIO

قسم البرمجة وتطوير الويب

Programming & Web Development Portfolio

إعداد وتطوير: Hebah Abdelwahab

الموقع: Hebah Portfolio

مستودع المشروع: <https://github.com/hebah-abdelwhab26/hebah-portfolio>

القسم الأول: التوثيق باللغة العربية

نبذة عن المشروع .

هو القسم البرمجي من الموقع الشخصي ويهدف إلى عرض المشاريع البرمجية وتطبيقات الويب بطريقة منظمة واحترافية يوفر هذا القسم مساحة لاستعراض الأعمال التقنية، والتعريف بفكرة كل مشروع، والتقنيات المستخدمة في تطويره، والوظائف التي يقدمها، مع إمكانية إضافة روابط المعاينة والمستودعات البرمجية والصور التوضيحية مع قاعدة بيانات لتخزين معلومات المشاريع وتصنيفاتها والتقنيات المرتبطة بها PHP و Laravel تم تطوير الموقع باستخدام لا يقتصر الهدف على عرض صور المشاريع فقط، بل يشمل بناء نظام منظم لإدارة بياناتها وتحديثها من خلال لوحة الإدارة أهداف المشروع .

Digital Studio يهدف الى :

- إنشاء واجهة احترافية لعرض المشاريع البرمجية
 - تنظيم المشاريع ضمن تصنيفات واضحة
 - تقديم وصف تفصيلي لكل مشروع
 - توضيح التقنيات والأدوات المستخدمة في التطوير
 - إتاحة روابط المستودعات البرمجية والمعاينات المباشرة عند توفرها
 - توفير لوحة إدارة لإضافة المشاريع وتعديلها وحذفها
 - إدارة صور المشاريع والملفات المرفقة
 - التحكم في حالة نشر المشاريع وترتيب ظهورها
 - إنشاء بنية قابلة للتوسع مع إضافة مشاريع جديدة
 - عرض المهارات العملية في تطوير تطبيقات الويب وإدارة قواعد البيانات
- فكرة النظام وآلية عمله .

يعتمد النظام على الفصل بين واجهة عرض المشاريع ولوحة الإدارة

واجهة عرض المشاريع

تتيح للزوار استكشاف المشاريع المنشورة والتعرف على تفاصيلها وتقنياتها، والانتقال إلى الروابط المرتبطة بها

لوحة الإدارة

تتيح للمسؤول إنشاء سجلات المشاريع وتعديل معلوماتها وإدارة الصور والتصنيفات والتقنيات وحالة النشر

قاعدة البيانات

تخزن بيانات المشاريع والعلاقات المرتبطة بها، مما يسمح باسترجاع المعلومات وعرضها بصورة منظمة

تعمل هذه الأجزاء معًا لتوفير نظام ديناميكي لإدارة محتوى الأعمال البرمجية بدلاً من كتابة معلومات كل مشروع يدويًا داخل صفحات الموقع

الواجهة العامة للمشاريع .

تمثل الواجهة العامة نقطة الوصول الأساسية للزوار الراغبين في استكشاف الأعمال البرمجية

ويمكن أن تعرض بيانات المشاريع المنشورة، مثل :

اسم المشروع

صورة المشروع أو صورته المصغرة

وصف مختصر

التصنيف

التقنيات المستخدمة

حالة المشروع أو نوعه بحسب طريقة العرض

رابط المعاينة المباشرة عند توفره

عند توفره GitHub رابط

يساعد هذا التنظيم على تقديم نظرة سريعة على الأعمال، مع إمكانية الانتقال إلى صفحة التفاصيل للحصول على معلومات أوسع

صفحة تفاصيل المشروع

تخصص صفحة تفاصيل مستقلة لكل مشروع، بهدف شرح فكرته ووظائفه بطريقة أوضح من بطاقة العرض المختصرة

يمكن أن تتضمن الصفحة العناصر التالية

اسم المشروع

يعرض اسم المشروع بصورة واضحة ويحدد هويته

الوصف المختصر والتفصيلي

يشرح الوصف المختصر فكرة المشروع، بينما يقدم الوصف التفصيلي معلومات أوسع عن المشكلة التي يعالجها المشروع، وطريقة عمله، والوظائف التي يوفرها.

صور المشروع

.تستخدم الصور لعرض واجهات المشروع أو لقطات الشاشة التي توضح تصميمه ووظائفه

التقنيات المستخدمة

بحسب طبيعة MySQL أو React أو Laravel يعرض القسم التقنيات البرمجية المستخدمة في تطوير المشروع، مثل المشروع.

الروابط الخارجية

يمكن ربط المشروع بروابط ذات صلة، مثل :

.المعاينة المباشرة

.GitHub مستودع

.Figma تصميم

.ملفات أو مستندات المشروع، إذا كانت متاحة

.تختلف المعلومات الظاهرة في صفحة التفاصيل وفق البيانات التي تمت إضافتها للمشروع

لوحة إدارة المشاريع

تُعد لوحة الإدارة أحد المكونات الرئيسية للنظام، إذ توفر واجهة لإدارة محتوى المشاريع دون الحاجة إلى تعديل ملفات العرض يدويًا في كل مرة

:تتضمن وظائف إدارة المشاريع ما يلي

إضافة مشروع جديد

توفر صفحة إنشاء المشروع نموذجًا لإدخال البيانات الأساسية، مثل العنوان والوصف والتصنيف والتقنيات والروابط والصور

تعديل بيانات المشروع

.تسمح وظيفة التعديل بتحديث معلومات المشروع الموجود، بما يساعد على إبقاء محتوى الموقع محدثًا

حذف المشروع

.توفر وظيفة الحذف إمكانية إزالة المشاريع التي لم تعد مطلوبة، وفق صلاحيات الإدارة وآلية الحذف المستخدمة في النظام

إدارة حالة النشر

.يمكن تحديد ما إذا كان المشروع مسودة أو منشورًا، مع وجود حقل لتفعيل المشروع أو تعطيله

يساعد ذلك على تجهيز المشروع ومراجعته قبل عرضه للزوار.

ترتيب المشاريع

يتضمن النظام حقلاً لترتيب عرض المشاريع، مما يساعد على تنظيم ظهورها وفق الأولوية المطلوبة.

إدارة التصنيفات

ترتبط المشاريع بتصنيفات تساعد على تجميع الأعمال المتشابهة، مثل مشاريع الويب أو التطبيقات أو أنواع أخرى بحسب التصنيفات المعتمدة في الموقع.

إدارة التقنيات

يمكن ربط المشروع بالتقنيات المستخدمة في تطويره، مما يساعد على توضيح الأدوات والخبرات التي يتضمنها كل عمل.

إدارة الصور والملفات

يدعم نموذج المشروع مجموعة من حقول الصور والملفات، بما يساعد على تقديم المشروع بصرياً وتوفير مستندات إضافية عند الحاجة.

الصورة المصغرة

تستخدم لتمثيل المشروع في بطاقات العرض والقوائم التي تعرض عدة مشاريع.

صورة الغلاف

تستخدم لتقديم صورة رئيسية للمشروع في المواضيع التي تدعم عرضها.

معرض الصور

يتيح تنظيم عدة صور مرتبطة بالمشروع، مثل لقطات الشاشة أو واجهات التطبيق.

PDF ملف

اختياري، يمكن استخدامه لتوفير مستند توضيحي للمشروع عند الحاجة PDF يدعم النموذج إرفاق ملف.

تنظيم الملفات

تتطلب إدارة الملفات التحقق من نوع الملف وحجمه، والتأكد من حفظ مسارات الملفات بطريقة صحيحة واستخدامها عند عرض المشروع.

قاعدة البيانات والعلاقات

لتخزين معلومات المشاريع والعناصر المرتبطة بها MySQL يعتمد النظام على قاعدة بيانات

ويتضمن سجل المشروع حقولاً مثل:

- عنوان المشروع: title
- الرابط المختصر للمشروع: slug

- subtitle: العنوان الفرعي.
- short_description: الوصف المختصر.
- description: الوصف التفصيلي.
- project_category_id: معرف تصنيف المشروع.
- thumbnail: مسار الصورة المصغرة.
- cover_image: مسار صورة الغلاف.
- pdf_file: الاختياري مسار ملف PDF.
- live_demo: رابط المعاينة المباشرة.
- github: رابط المستودع البرمجي.
- figma: رابط التصميم.
- client: اسم العميل عند استخدام هذا الحقل.
- project_date: تاريخ المشروع.
- duration: مدة تنفيذ المشروع.
- featured: تحديد ما إذا كان المشروع مميزًا.
- is_active: حالة تفعيل المشروع.
- sort_order: ترتيب العرض.
- status: حالة النشر.

يساعد تنظيم هذه الحقول على تخزين بيانات المشروع بطريقة واضحة وقابلة للإدارة.

العلاقات بين المشاريع والتصنيفات والتقنيات 9.

يستخدم النظام علاقات بين البيانات لتجنب تكرار المعلومات وتسهيل تنظيم المحتوى.

علاقة المشروع بالتصنيف

يرتبط المشروع بتصنيف محدد، مما يسمح بتجميع المشاريع ضمن مجموعات مناسبة.

علاقة المشروع بالتقنيات

يمكن ربط المشروع بعدة تقنيات، لأن المشروع الواحد قد يستخدم أكثر من لغة برمجة أو إطار عمل أو أداة.

علاقة المشروع بالصور

يمكن ربط المشروع بصور إضافية لعرضها ضمن معرض خاص به.

يساعد هذا التصميم على فصل البيانات الأساسية عن المعلومات المرتبطة بها، مع المحافظة على بنية منظمة

التحقق من البيانات ومعالجة الأخطاء

يتضمن النظام قواعد للتحقق من المدخلات قبل حفظ المشاريع

وتشمل هذه القواعد التحقق من الحقول المطلوبة، وصحة الروابط، وصيغ الملفات المسموح بها، والأحجام المقبولة للملفات المرفوعة، ووجود التصنيف المطلوب

في cover_image خلال تطوير النظام، ظهرت مشكلة في حفظ مشروع عندما لم تُرفع صورة غلاف؛ إذ كان عمود قاعدة البيانات يتطلب قيمة، على الرغم من أن الصورة اختيارية في نموذج الإنشاء

عند عدم رفع صورة NULL تمت معالجة المشكلة بتعديل تعريف العمود ليُسمح بالقيمة

يوضح هذا المثال أهمية توافق قواعد التحقق في التطبيق مع القيود الفعلية في قاعدة البيانات

كما يوضح أهمية عرض رسائل الأخطاء بوضوح للمسؤول حتى يتمكن من تحديد سبب فشل الحفظ ومعالجته

الأمان والصلاحيات

تتطلب لوحة الإدارة حماية العمليات التي تسمح بتعديل محتوى الموقع

ومن الجوانب المهمة في هذا النوع من الأنظمة :

حماية صفحات الإدارة من الوصول غير المصرح به

التحقق من صلاحيات المستخدم قبل تنفيذ العمليات الإدارية

التحقق من البيانات المستلمة من النماذج

في النماذج CSRF استخدام حماية

التحقق من أنواع الملفات المرفوعة وأحجامها

عدم الاعتماد على واجهة المستخدم وحدها لضمان صحة البيانات

يجب أن تتوافق تفاصيل التوثيق الأمني مع آليات الحماية المنفذة فعلاً في النسخة المنشورة

التقنيات المستخدمة

Laravel

إطار العمل الأساسي المستخدم في بناء التطبيق، وتنظيم المسارات والمتحكمات وعمليات إدارة البيانات

PHP

لغة البرمجة المستخدمة لتنفيذ منطق التطبيق على الخادم

MySQL

قاعدة البيانات المستخدمة لتخزين المشاريع والتصنيفات والعلاقات المرتبطة بها

Blade

محرك القوالب المستخدم لعرض الصفحات وبيانات المشاريع.

HTML5

يستخدم لبناء البنية الأساسية لصفحات الموقع والنماذج.

CSS3

يستخدم لتنسيق الواجهات وإدارة الألوان والمسافات والتخطيطات.

JavaScript

يستخدم لإضافة التفاعلات المطلوبة داخل الواجهات، بحسب الوظائف المنفذة.

Git وGitHub

مستودع المشروع ويساعد على إدارة إصداراته GitHub لتتبع التغييرات في الكود، بينما يستضيف Git يستخدم

تنظيم بنية التطبيق

التي تفصل مسؤوليات التطبيق إلى أجزاء مختلفة Laravel يعتمد المشروع على بنية

(Routes) المسارات

تحدد المسارات عناوين الصفحات والعمليات المتاحة، وترتبط الطلبات بالمتحكمات المناسبة

(Controllers) المتحكمات

تتعامل المتحكمات مع الطلبات وتنفذ العمليات المطلوبة، مثل إنشاء المشروع وتعديله وحذفه

(Models) النماذج

Eloquent ORM. تمثل النماذج البيانات المخزنة في قاعدة البيانات، وتساعد على التعامل معها من خلال

(Form Requests) طلبات التحقق

تستخدم قواعد التحقق لتحديد البيانات المسموح بها قبل تنفيذ عمليات الحفظ

Blade قوالب

تعرض صفحات الموقع وبيانات المشاريع، وتوفر نماذج لإدخال البيانات من لوحة الإدارة

يساعد هذا الفصل بين المسؤوليات على تسهيل صيانة التطبيق وتطويره

التصميم وتجربة المستخدم

. على عرض الأعمال البرمجية بطريقة منظمة، مع إبراز هوية كل مشروع وتقديم معلوماته الأساسية دون إرباك الزائر يركز

وتشمل أهداف التصميم :

توحيد أسلوب عرض المشاريع

استخدام صور مناسبة لكل مشروع

تنظيم التفاصيل ضمن أقسام واضحة

تسهيل الوصول إلى الروابط الخارجية

إظهار التقنيات المستخدمة

دعم إضافة مشاريع جديدة دون إعادة تصميم الصفحات الأساسية

الحفاظ على هوية بصرية متناسقة مع الموقع الشخصي

التحديات التقنية

من أبرز الجوانب التقنية التي يتناولها المشروع

بناء نظام ديناميكي لإدارة المشاريع

تصميم قاعدة بيانات منظمة للمشاريع والتصنيفات والتقنيات

إدارة الصور والملفات المرفقة

التحقق من صحة البيانات قبل الحفظ

معالجة التعارض بين الحقول الاختيارية وقيود قاعدة البيانات

تنظيم عمليات الإضافة والتعديل والحذف

إدارة حالة النشر وتفعيل المشاريع

التعامل مع العلاقات بين الجداول

عرض رسائل الأخطاء والنجاح بطريقة واضحة

تنظيم الكود بحيث يسهل تطوير وظائف جديدة مستقبلاً

المهارات التي يبرزها المشروع

يعكس المشروع تطبيقاً عملياً لمجموعة من المهارات، من بينها

Laravel تطوير تطبيقات الويب باستخدام

PHP البرمجة الخلفية باستخدام

MySQL تصميم قواعد بيانات

Eloquent ORM استخدام

CRUD إنشاء عمليات

بناء نماذج إدخال البيانات والتحقق منها

رفع الصور والملفات وإدارة مساراتها

تصميم العلاقات بين المشاريع والتصنيفات والتقنيات

لبناء صفحات ديناميكية Blade استخدام

تنظيم المسارات والمتحكمات والنماذج

تشخيص أخطاء قاعدة البيانات ومعالجتها

لإدارة الكود GitHub و Git استخدام

النتائج والخلاصة

نظامًا لعرض المشاريع البرمجية وإدارة معلوماتها من خلال موقع شخصي ولوحة إدارة Digital Studio يمثل

ويساعد على تقديم الأعمال بصورة احترافية، مع توفير بنية منظمة لإدارة التصنيفات والتقنيات والصور والروابط وحالة النشر.

يتعامل مع قواعد البيانات والنماذج والملفات والعلاقات، ومعالجة Laravel كما يبرز المشروع القدرة على تطوير تطبيق المشكلات التقنية التي تظهر أثناء التطوير

DIGITAL STUDIO

Programming and Web Development Portfolio

1. Project Overview

Digital Studio is the programming and web development section of the Hebah Portfolio website. It is designed to present software projects and web applications in a structured and professional format.

The section provides a place to showcase technical work, explain each project's purpose, describe its technologies, and provide related images and external links.

The website is built with Laravel and PHP, with a database for storing project details, categories, and associated technologies.

Beyond displaying project cards, the system includes administrative functionality for maintaining project information without manually editing the presentation pages.

2. Project Objectives

Digital Studio aims to:

1. Provide a professional interface for showcasing software projects.
2. Organize projects into categories.
3. Present detailed descriptions of individual projects.
4. Identify the technologies used in development.
5. Provide GitHub and live-demo links where available.
6. Support project creation, editing, and deletion through an administration panel.
7. Manage project images and attached documents.
8. Control publication status and display order.
9. Provide a scalable structure for future projects.
10. Demonstrate practical web development and database-management skills.

3. System Concept and Workflow

The application separates the public project interface from the administration panel.

Public Project Interface: Allows visitors to explore published projects, review their details, and access related links.

Administration Panel: Allows authorized users to create and update project records and manage images, categories, technologies, and publication status.

Database: Stores project information and its related records so that content can be retrieved and displayed dynamically.

Together, these components provide a maintainable project-management workflow instead of requiring manual updates to individual presentation pages.

4. Public Project Interface

The public interface provides an overview of the available programming projects.

Project listings can include:

- Project title.
- Project thumbnail.
- Short description.
- Category.
- Technologies used.
- Publication or project status, depending on the interface.
- Live-demo link, when available.
- GitHub repository link, when available.

This structure allows visitors to quickly explore the portfolio and open individual project pages for more information.

5. Project Details Page

Each project can have a dedicated details page that explains its purpose and functionality.

5.1 Project Title

Identifies the project clearly.

5.2 Short and Detailed Descriptions

The short description introduces the project, while the full description provides additional information about its purpose, workflow, and functionality.

5.3 Project Images

Images illustrate the application's interface, design, and relevant screens.

5.4 Technologies Used

The technology section identifies the programming languages, frameworks, and tools used in each project.

5.5 External Links

Project records can include links to:

- Live demos.
- GitHub repositories.
- Figma designs.
- Supporting project documents, where available.

The information displayed depends on the data provided for each project.

6. Project Administration Panel

The administration panel provides a central interface for maintaining project content.

6.1 Create Projects

The creation form supports entering project information, including title, descriptions, category, technologies, links, and images.

6.2 Edit Projects

Existing project records can be updated to keep the portfolio content current.

6.3 Delete Projects

The administration interface provides a deletion workflow for projects that are no longer required, subject to the application's permissions and deletion behavior.

6.4 Publication Status

Projects can be managed using draft and published statuses, together with an activation field.

This supports preparing and reviewing content before making it available to visitors.

6.5 Display Order

A sort-order field supports organizing the order in which projects are displayed.

6.6 Category Management

Projects can be associated with categories to group related work.

6.7 Technology Associations

Each project can be linked to its associated technologies, helping visitors understand the tools used during development.

7. Image and File Management

The project form supports multiple image and document fields.

Thumbnail: Represents the project in listings and cards.

Cover Image: Provides a main visual for project pages or other supported display locations.

Image Gallery: Stores additional images associated with a project.

PDF Document: Supports an optional PDF attachment.

File Organization: Uploaded files require appropriate validation and reliable storage paths so they can be retrieved and displayed correctly.

8. Database Structure

The application uses MySQL to store project records and related information.

A project record includes fields such as:

- title: Project title.
- slug: URL-friendly project identifier.
- subtitle: Project subtitle.
- short_description: Short project summary.
- description: Full project description.
- project_category_id: Associated category identifier.
- thumbnail: Thumbnail image path.
- cover_image: Cover image path.
- pdf_file: Optional PDF document path.
- live_demo: Live demonstration URL.
- github: Repository URL.
- figma: Design URL.
- client: Client name, where applicable.
- project_date: Project date.

- duration: Project duration.
- featured: Indicates whether the project is featured.
- is_active: Project activation status.
- sort_order: Display order.
- status: Publication status.

This structure supports organized storage and administration of project information.

9. Relationships Between Projects, Categories, Technologies, and Images

The system uses relationships to organize associated data and reduce unnecessary duplication.

Project–Category Relationship: Associates a project with a category.

Project–Technology Relationship: Allows one project to be associated with multiple technologies.

Project–Image Relationship: Supports additional images associated with a project.

These relationships provide a structured approach to storing and retrieving project data.

10. Validation and Error Handling

The application validates submitted information before saving project records.

Validation can include required fields, URL formats, accepted file types, file-size limits, and category existence.

During development, a database constraint prevented project creation when no cover image was uploaded. The form treated the image as optional, but the database column required a value.

The issue was resolved by adjusting the database column to allow NULL when a cover image is not provided.

This example demonstrates the importance of maintaining consistency between form validation, application logic, and database constraints.

Clear error messages are also important because they help administrators understand why a save operation failed.

11. Security and Access Control

Administrative functionality should be protected against unauthorized access.

Important security considerations include:

- Protecting administrative routes.
- Checking user permissions before administrative actions.
- Validating submitted data.
- Using CSRF protection for forms.
- Validating uploaded file types and sizes.
- Enforcing data integrity on the server side.

Security documentation should reflect the mechanisms implemented in the deployed application.

12. Technology Stack

Laravel: Main framework for application structure, routing, controllers, and data-management workflows.

PHP: Server-side programming language.

MySQL: Database for project records, categories, and related data.

Blade: Template engine for rendering project pages and administrative forms.

HTML5: Page structure and form markup.

CSS3: Interface styling, layouts, colors, and spacing.

JavaScript: Client-side interactions implemented in the interface.

Git and GitHub: Version control and source-code hosting.

13. Application Architecture

The project follows Laravel's separation of application responsibilities.

Routes: Define page URLs and connect incoming requests to the appropriate handlers.

Controllers: Process requests and execute operations such as creating, updating, and deleting projects.

Models: Represent database records and provide access through Eloquent ORM.

Form Requests: Define validation rules for incoming data.

Blade Templates: Render pages and provide forms for managing project content.

This structure supports maintainability and future development.

14. Design and User Experience

Digital Studio focuses on presenting programming work in a consistent and professional format.

Its design objectives include:

- Consistent project cards and layouts.
- Appropriate project imagery.
- Clear organization of project details.
- Easy access to external links.
- Visible technology information.
- Support for adding new projects without rebuilding the main interface.
- Visual consistency with the personal portfolio.

15. Technical Challenges

Important technical considerations include:

1. Building a dynamic project-management system.
2. Structuring project, category, and technology data.
3. Managing uploaded images and documents.
4. Validating submitted information.
5. Resolving conflicts between optional form fields and database constraints.
6. Implementing create, update, and delete operations.
7. Managing project publication and activation states.
8. Maintaining relationships between database tables.
9. Displaying clear success and error messages.
10. Organizing the application for future extensions.

16. Skills Demonstrated

The project demonstrates practical experience with:

- Laravel web development.

- PHP backend programming.
- MySQL database design.
- Eloquent ORM.
- CRUD operations.
- Form validation.
- Image and document uploads.
- Relationships between projects, categories, and technologies.
- Dynamic Blade templates.
- Route, controller, and model organization.
- Database error diagnosis and resolution.
- Git and GitHub version control.

17. Conclusion

Digital Studio provides a structured system for presenting programming projects and managing their information through a personal website and administration panel.

It supports professional project presentation while organizing categories, technologies, images, external links, and publication status.

The project demonstrates the ability to build a Laravel application that works with databases, forms, files, and related records while addressing technical issues encountered during development