CIT6224: Web Application Development

Term: 2510

Subject: CIT6224 Web Application Development

Assignment Title: MMU Talent Showcase Portal

Group : R

Section : TC1L

Student Name	Student ID	Tutorial Section
Maryam binti Norazman	1211111809	TT4L
Nur Dania Iman Binti Desman Desa	1211112284	TT4L
Fadhilah Binti Mohamad Asri	1211312193	TT3L
Azryl Shamin bin Azrizal	1211103145	TT3L

Table Of Content

1.0 Introduction / Project Overview

1.1 Project Objectives

The primary objective of the MMU Talent Showcase Portal is to empower MMU students by providing a centralized, secure, and user-friendly web platform to showcase their academic, technical, and creative talents. The portal enables students to:

- Create personalized profiles.
- Upload diverse portfolios (e.g., code, artwork, videos, writing).
- Engage with peers through a public feedback wall and talent request board.
- Connect with other students and potential collaborators.
- Download or share CVs and seek freelance opportunities.

The system also ensures quality and governance through admin controls like content moderation, announcement postings, and activity analytics.

1.2 Scope of the Project

The project scope includes the development and deployment of a fully functional web application featuring:

- **Frontend**: HTML, CSS, and JavaScript for layout and interactivity.
- **Backend**: PHP for server-side scripting and form handling.
- **Database**: MySQL for storing user data, portfolios, and interactions. **Authentication**: User registration, login/logout, and admin-level access control.
- **User Roles**:
    - **Students** – Can register, upload work, post on the public wall, and submit talent collaboration requests.
    - **Admins** – Manage user content, announcements, feedback, and monitor system activity.

1.3 Target Users

MMU Students – The primary users who will:

- Register and maintain profiles.
- Upload their portfolios.
- Search and discover peer talents.
- Interact through feedback and collaboration boards.

Administrators – Responsible for:

- the maintenance, moderation, and overall management of the portal.
- Managing user accounts
- Moderating content (wall posts, portfolios, feedback)
- Posting news/announcements.
- Monitoring usage through dashboards.

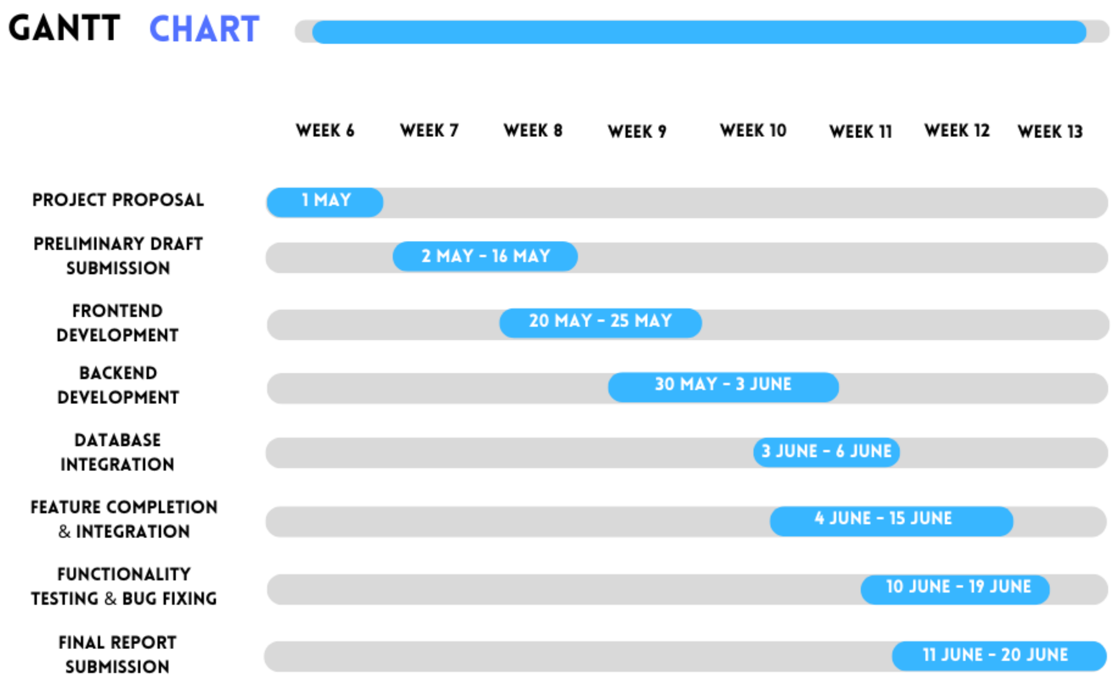
1.4 Major Features and Functionalities

- **User Registration & Login** (with role-based access)
- **Catalogue** with talent category filters
- **User Profile Management**
- **Portfolio Upload & Management** (files + descriptions + categories)
- **CV Upload and Download Zone**
- **Feedback Form** and **Public Feedback Wall**
- **Talent Request Board** for collaboration
- **Admin Dashboard**:
    - Overview portal activity
    - User management (ban, reset, approve)
    - Content moderation (posts, portfolios)
    - News & Announcement publishing
- **FAQ Section** with user-submitted questions and admin-managed content

- **Client-Side & Server-Side Validation** for all user input
- **Secure File Upload System**
- **Public Wall**
  - Post Creation: Users can post messages and upload photos.
  - Reply Functionality: Users can reply to existing posts.
  - Moderation: Content can be deleted by the user or an administrator.

2.0 Project Plan

2.1 Development Timeline (Gantt Chart)



2.2 Tools and Technologies Used

Category	Technology / Tools
Frontend	HTML, CSS, JavaScript
Backend	PHP
Database	MySQL
Server	Xampp (Apache and MySQL)
Development Tools	Vs Code

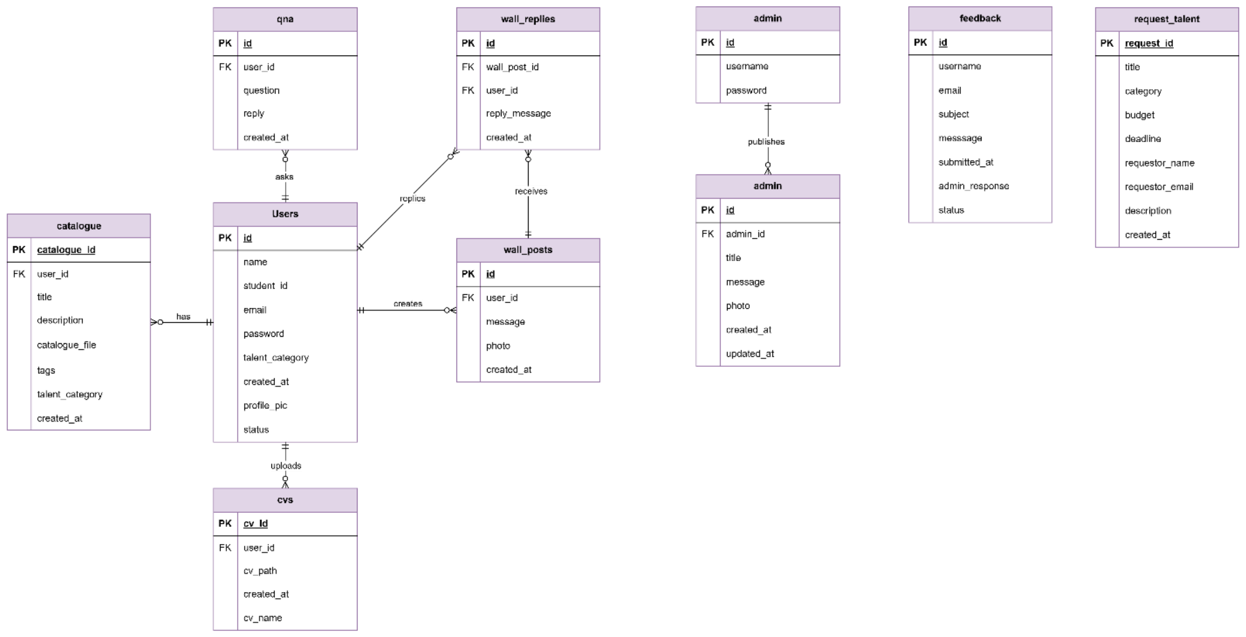
2.3 Project Milestones

Phase	Milestone	Description	Deliverables	Due Date	Week
Planning	Project Proposal	Define the project's goals, scope, users, and key features.	Proposal document (title, objectives, scope, user specs)	1 May 2025	Week 6
System Design	Preliminary Draft Submission	Design the structure of the system.	ERD, Data Dictionary, Wireframes, Website Navigation Structure	16 May 2025	Week 8
Implemen-tation	Frontend Development	Build the user interface using HTML, CSS, JavaScript.	Web pages (e.g., homepage, catalogue, profile), form validations	20 May 2025 - 25 May 2025	Week 9

	Backend Development	Develop server-side functionalities with PHP and MySQL.	Registration/login, profile updates, file uploads, shopping cart, etc.	30 May 2025 - 3 June 2025	Week 10
	Database Integration	Connect backend logic with MySQL database.	Database tables, queries, connection scripts	6 June 2025	Week 11
	Feature Completion & Integration	Integrate all components and ensure system-wide functionality.	Full-feature web app (with portfolio, feedback, CV zone, admin dashboard, etc.)	15 June 2025	Week 12
Testing	Functionality Testing & Bug Fixing	Validate all inputs and test system interactions; fix any detected issues.	Debug log, feedback notes, refined features	19 June 2025	Week 13
Documen-tation	Final Report Submission	Prepare the technical report summarizing the system, features, and usage.	Final report, project structure overview, screenshots, usage instructions	20 June 2025	Week 13

3.0 System Design

3.1 Final Entity Relationship Diagram (ERD)



3.2 Data Dictionary

Users

Attribute	PK or FK	Data Type	Description
id	PK	INT	Primary key for the student
name		VARCHAR(100)	Name of the student
student_id		VARCHAR(50)	Student's identification number
email		VARCHAR(100)	Email address of the student
password		VARCHAR(255)	Student Password
talent_category		TEXT	User’s talent category
created_at		TIMESTAMP	Time and date account created

bio		TEXT	Short student biography
profile_pic		VARCHAR(50)	Optional path to user’s profile picture
status		VARCHAR(50)	Student account status (active, ban)

Admin

Attribute	**PK or FK**	**Data Type**	**Description**
id	PK	INT	Primary key for student
username		VARCHAR(100)	Username of the admin
password		VARCHAR(50)	Admin's password

Feedback

Attribute	**PK or FK**	**Data Type**	**Description**
id	PK	INT	Primary key for the submission
username		VARCHAR(100)	Name of the student submitting the form
email		VARCHAR(100)	Email address of the student
subject		VARCHAR(255)	Subject or title of the message

message		TEXT	Detailed message from the student
submitted_at		TIMESTAMP	Timestamp of when the form was submitted
admin_response		TEXT	Response or reply from the admin
status		VARCHAR(50)	Status of the feedback (pending, resolve)

Talent Request

Attribute	PK or FK	Data Type	Description
request_id	PK	INT	Primary key for the request
title		VARCHAR(255)	Title of the talent requested
description		TEXT	Detailed description on the talent requested
category		VARCHAR(100)	Category of the request

budget		DECIMAL(10,2)	Budget estimation for the request that talent can expect
deadline		DATE	The due date of the requested talent needed
requester_name		VARCHAR(100)	The name of the talent requested
requester_email		VARCHAR(255)	The email address of the talent requester
created_at		TIMESTAMP	The time that the request are made

Faqs

Attribute	PK or FK	Data Type	Description
id	PK	INT	Primary key of the FAQ post
question		TEXT	Question of the faq
answer		TEXT	Answer to the faq question

Catalogue

Attribute	PK or FK	Data Type	Description

catalogue_id	PK	INT	Primary key as a unique identifier for each portfolio
user_id	FK	INT	Referencing users(id) of the user who uploaded the portfolio
title		VARCHAR(100)	Title of the talent
description		TEXT	Description of the portfolio
catalogue_file		VARCHAR(255)	Path to the uploaded portfolio file
tags		TEXT	Tags to categorize work (web development, art, etc.)
talent_category		VARCHAR(100)	The category of talent
created_at		TIMESTAMP	The timestamp of when the portfolio was uploaded

CVs

Attribute	PK or FK	Data Type	Description
cv_id	PK	INT	Primary key as a unique identifier for each CV to specific user
user_id	FK	INT	Referencing users(id) of the user who uploaded the CV file
cv_path		VARCHAR(255)	Path to the uploaded CVfile

cv_name		VARCHAR(255)	Name of the CV file
created_at		TIMESTAMP	The timestamp of when the CV was uploaded

Wall_posts

Attribute	PK or FK	Data Type	Description
id	PK	INT	Primary key as a unique identifier for each wall post
user_id	FK	INT	Referencing users(id) of the user who post in public wall
message		TEXT	Message user write in public wall
photo		VARCHAR(255)	Path to the uploaded file
created_at		TIMESTAMP	The timestamp of when the wall post was posted

Wall_replies

Attribute	PK or FK	Data Type	Description
id	PK	INT	Primary key as a unique identifier for each wall repky
wall_post_id	FK	INT	Referencing wall_posts(id) of the user who post in public wall

user_id	FK	INT	Referencing users(id) of the user who reply in public wall
Reply_message		TEXT	Reply message user write in public wall
created_at		TIMESTAMP	The timestamp of when the wall post was posted

Announce

Attribute	PK or FK	Data Type	Description
id	PK	INT	Unique identifier for each announcement.
admin_id	FK	INT	Referencing admin(id), identifier for the admin who created the announcement.
title		VARCHAR(255)	Title of the announcement.
message		TEXT	Detailed message of the announcement.
photo		VARCHAR(255)	URL or path to an optional photo related to the announcement.
created_at		TIMESTAMP	The timestamp of when the announcement was created

updated_at		TIMESTAMP	The timestamp of when the announcement was updated

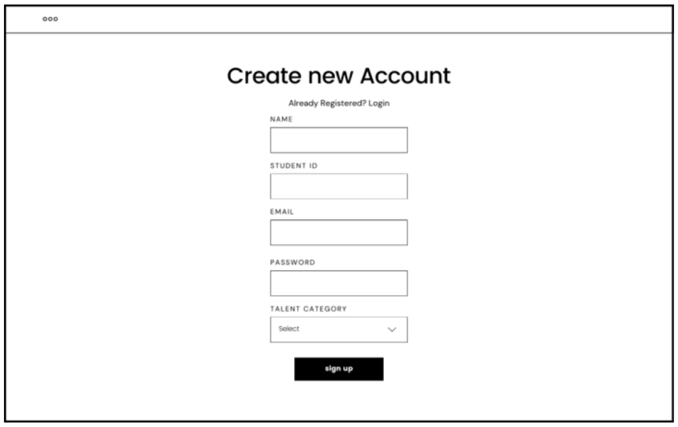
QnA

Attribute	PK or FK	Data Type	Description
id	PK	INT	Unique identifier for each Q&A entry.
user_id	FK	INT	Referencing users(id), identifier for the user who asked the question.
question		TEXT	The question asked by the user.
reply		TEXT	The reply to the user's question.
created_at		TIMESTAMP	Timestamp when the Q&A entry was created.

3.3 Wireframes for the primary layouts of the web pages

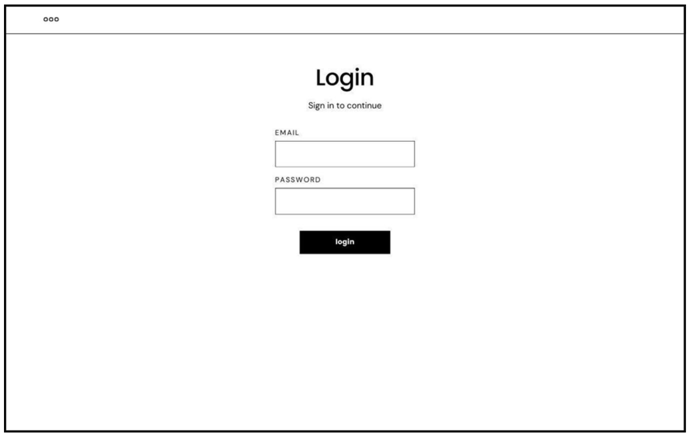
3.3.1 signup.php

A page where new students create an account by providing their name, student ID, email address, password, and choosing a skill category from a drop-down menu. The student can click the link to log in if they have already registered an account.



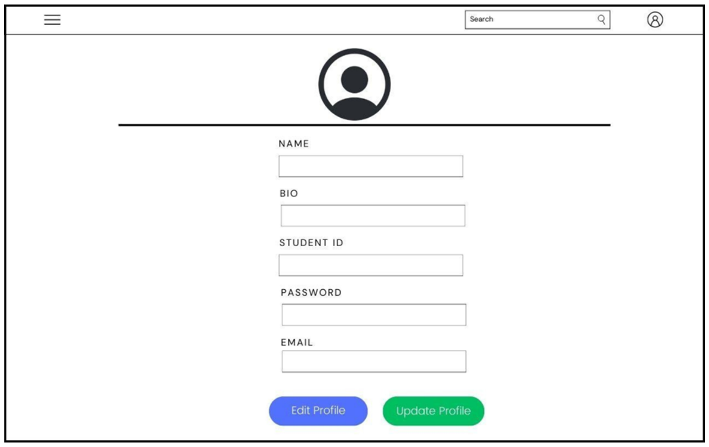
3.3.2 login.php

Login - If students already have an account, they just need to log in. To access the MMU Showcase Talent Portal, students must enter their email address and password.



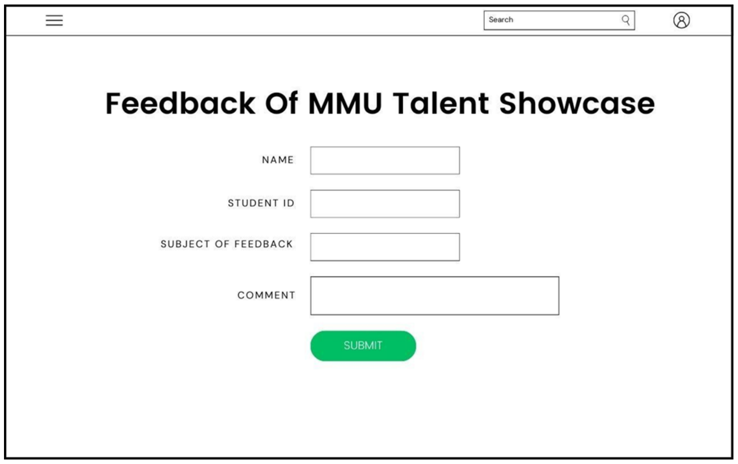
3.3.3 userdashboard.php

A form titled "Feedback of MMU Talent Showcase" where students can submit feedback by entering their name, student ID, subject of feedback and comments, about the MMU Talent Showcase, and clicking the "SUBMIT" button.



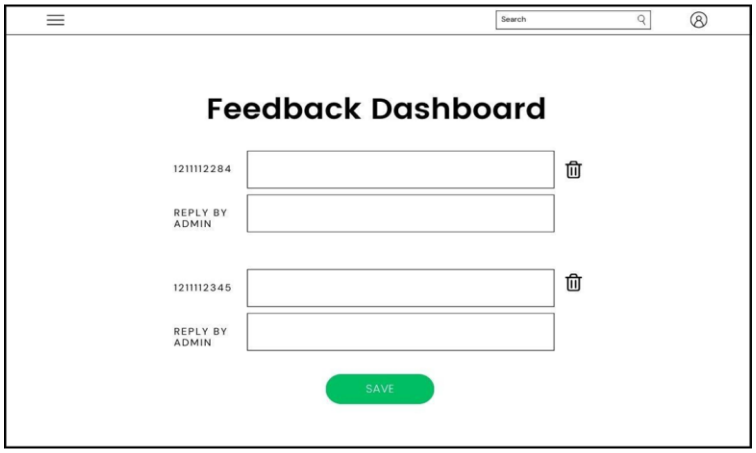
3.3.4 feedback.php

The form, titled "Feedback of MMU Talent Showcase," allows students to submit feedback by entering their name, student ID, subject of feedback, and comments before clicking the "SUBMIT" button.



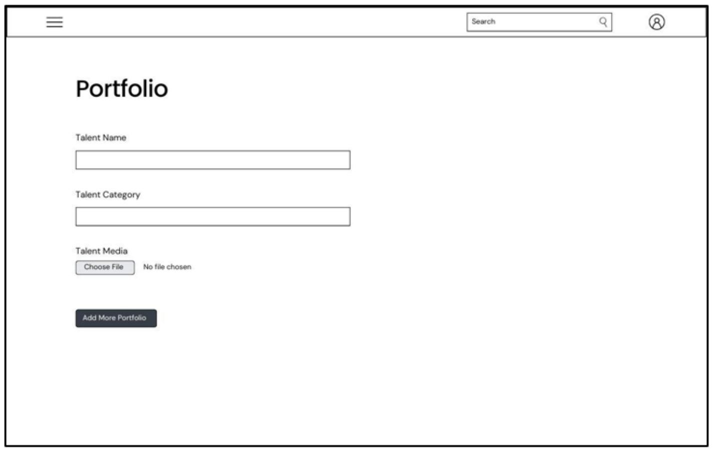
3.3.5 admin_feedback.php

Admin View - Similar to the student view but with buttons for deleting feedback entries and a green "send response" button at the bottom, allowing administrators to manage feedback and save their replies.



3.3.6 portfolio.php

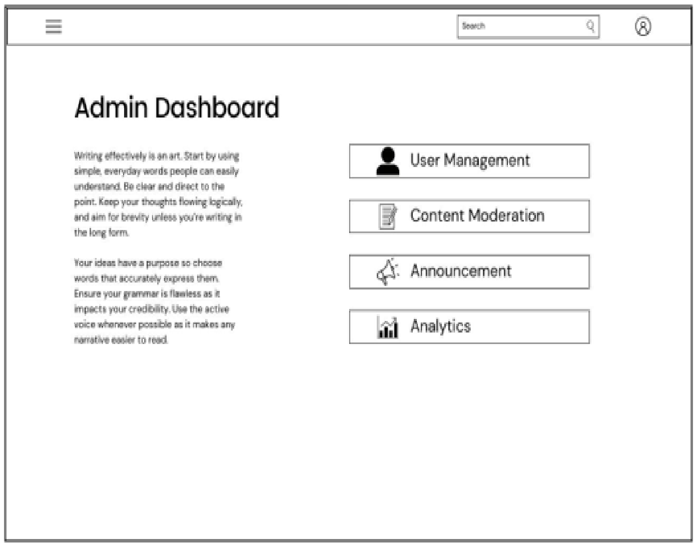
The portfolio page is an interactive platform designed for MMU students to upload their work, such as coding projects, artwork, videos, or written pieces. Users can submit files (PDF, images, videos) along with descriptions, tags, and categories for easy discovery. PHP handles file uploads, validating types and sizes while storing metadata in a MySQL database. Each portfolio entry is linked to a student ID, talent category, and portfolio visual, ensuring proper classification.



3.3.6 admin.php

The Admin Dashboard is a centralized control panel designed for administrators to efficiently manage the MMU Talent Showcase Portal. It provides key features such as User Management, allowing admins to approve or ban users and reset passwords, and

Content Moderation to review or remove portfolios, wall posts, and feedback. Through the Announcements feature, administrators can share important updates like events or policy changes, while the Analytics section offers insights into user activity, including uploads and logins. Powered by PHP, the dashboard securely authenticates admin roles, retrieves data from MySQL, and enables seamless CRUD (Create, Read, Update, Delete) operations via an intuitive interface.

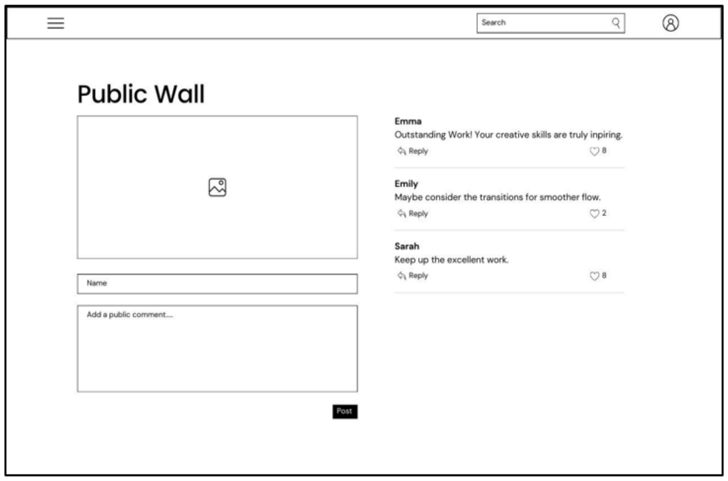


Writing effectively is an art. Start by using simple, everyday words people can easily understand. Be clear and direct to the point. Keep your thoughts flowing logically, and aim for brevity unless you're writing in the long form.

Your ideas have a purpose so choose words that accurately express them. Ensure your grammar is flawless as it impacts your credibility. Use the active voice whenever possible as it makes any narrative easier to read.

3.3.7 wall.php

The Public Wall is an interactive webpage where users can share comments, compliments, and feedback on displayed talents. It utilizes PHP form handling to validate input, ensuring appropriate and structured submissions, while its real-time display logic dynamically fetches and updates posts to maintain engagement. Users can like and reply to posts, fostering discussions and community interaction. To maintain a respectful environment, admins have moderation capabilities to filter out inappropriate content. This feature transforms the page into a live digital bulletin board, making it a dynamic and engaging platform for talent appreciation.



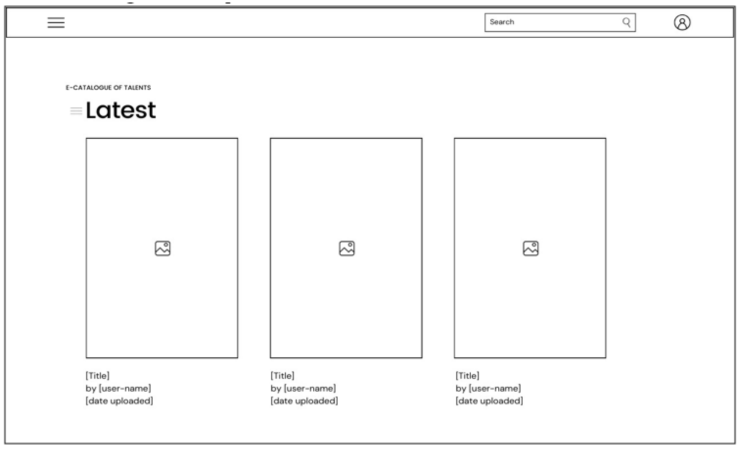
Emma
Outstanding Work! Your creative skills are truly inpiring.
Reply 8

Emily
Maybe consider the transitions for smoother flow.
Reply 2

Sarah
Keep up the excellent work.
Reply 8

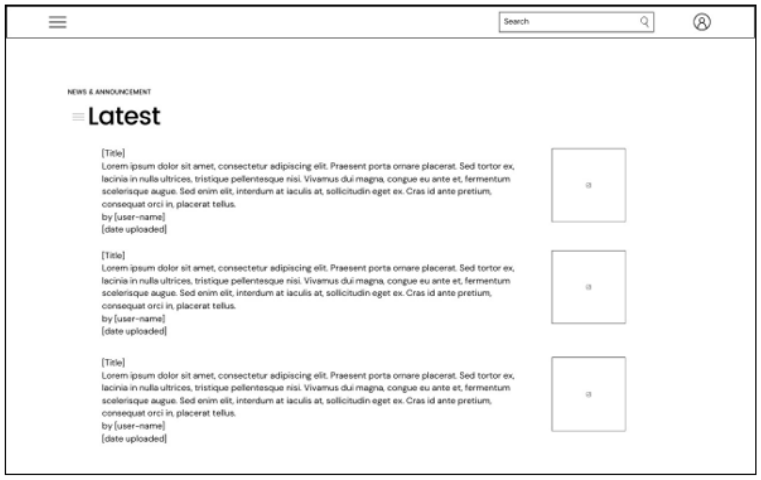
3.3.8 catalogue.php

This page displays a searchable and filterable catalogue of portfolios. Users can search by category, name, or tags. Each portfolio displayed includes a name, description, tags, and contact number. Layout will include a search bar, category filter, and a grid of user portfolios.



3.3.9 announcement.php

This public-facing page lists news and announcements posted by the admin. Each announcement includes a title, date, and content preview. Clicking a post expands to show full details. The layout includes a list of announcement cards in reverse chronological order.


Search

NEWS & ANNOUNCEMENT

Latest

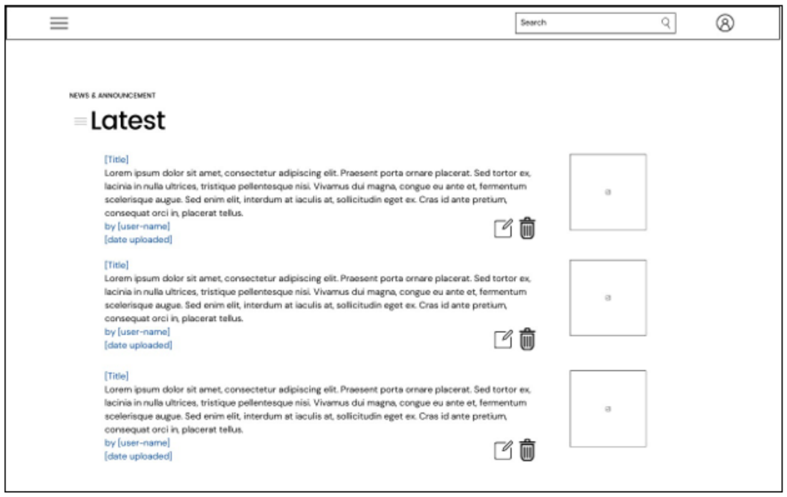
[Title]
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Praesent porta ornare placerat. Sed tortor ex, lacinia in nulla ultrices, tristique pellentesque nisi. Vivamus dui magna, congue eu ante et, fermentum scelerisque augue. Sed enim elit, interdum at iaculis at, sollicitudin eget ex. Cras id ante pretium, consequat orci in, placerat tellus.
by [user-name]
[date uploaded]

[Title]
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Praesent porta ornare placerat. Sed tortor ex, lacinia in nulla ultrices, tristique pellentesque nisi. Vivamus dui magna, congue eu ante et, fermentum scelerisque augue. Sed enim elit, interdum at iaculis at, sollicitudin eget ex. Cras id ante pretium, consequat orci in, placerat tellus.
by [user-name]
[date uploaded]

[Title]
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Praesent porta ornare placerat. Sed tortor ex, lacinia in nulla ultrices, tristique pellentesque nisi. Vivamus dui magna, congue eu ante et, fermentum scelerisque augue. Sed enim elit, interdum at iaculis at, sollicitudin eget ex. Cras id ante pretium, consequat orci in, placerat tellus.
by [user-name]
[date uploaded]

3.3.10 announcement_admin.php

This admin-only page allows the admin to add, edit, or delete announcements. It includes a form for posting new announcements and a table or list of existing announcements with edit/delete buttons.


Search

NEWS & ANNOUNCEMENT

Latest

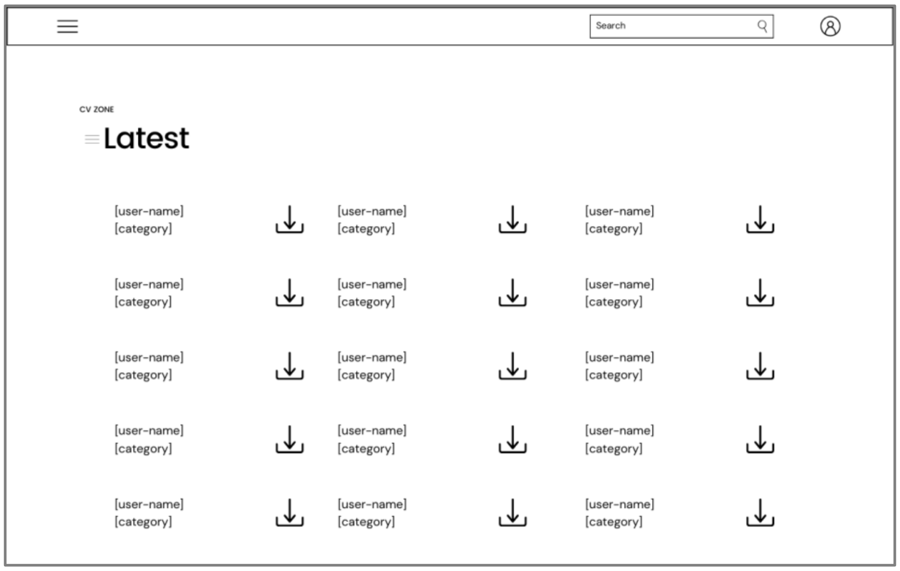
[Title]
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Praesent porta ornare placerat. Sed tortor ex, lacinia in nulla ultrices, tristique pellentesque nisi. Vivamus dui magna, congue eu ante et, fermentum scelerisque augue. Sed enim elit, interdum at iaculis at, sollicitudin eget ex. Cras id ante pretium, consequat orci in, placerat tellus.
by [user-name]
[date uploaded]

[Title]
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Praesent porta ornare placerat. Sed tortor ex, lacinia in nulla ultrices, tristique pellentesque nisi. Vivamus dui magna, congue eu ante et, fermentum scelerisque augue. Sed enim elit, interdum at iaculis at, sollicitudin eget ex. Cras id ante pretium, consequat orci in, placerat tellus.
by [user-name]
[date uploaded]

[Title]
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Praesent porta ornare placerat. Sed tortor ex, lacinia in nulla ultrices, tristique pellentesque nisi. Vivamus dui magna, congue eu ante et, fermentum scelerisque augue. Sed enim elit, interdum at iaculis at, sollicitudin eget ex. Cras id ante pretium, consequat orci in, placerat tellus.
by [user-name]
[date uploaded]

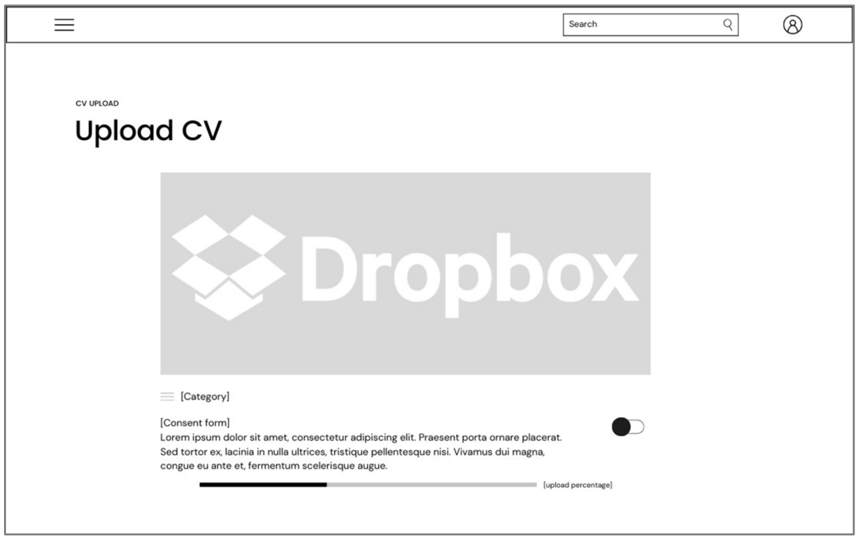
3.3.11 cv_zone.php

This page acts as a directory listing available CVs uploaded by users, with user consent. Users can view/download CVs. Each entry displays the user's name, category, and a download button.



3.3.12 upload_cv.php

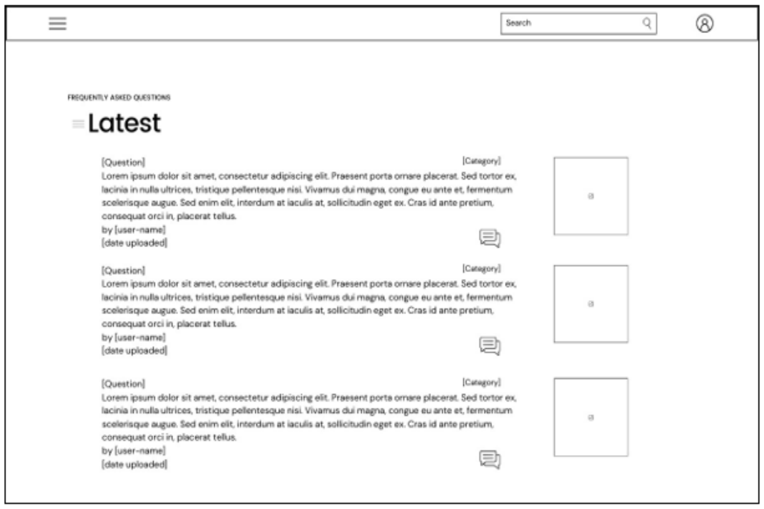
This page allows users to upload their CV for public access. It includes a form with file upload input and a consent checkbox. On submission, the file is stored and listed in the CV Zone.


Search
CV UPLOAD
Upload CV
Dropbox
[Category]
[Consent form]
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Praesent porta ornare placerat. Sed tortor ex, lacinia in nulla ultrices, tristique pellentesque nisi. Vivamus dui magna, congue eu ante et, fermentum scelerisque augue.
[upload percentage]

3.3.13 faq.php

This page allows users to submit their frequently asked questions and view existing ones. Admins will have the ability to edit or delete these questions and answers. The layout includes:

- A FAQ list section with expandable Q&A format.
- A user input form (question title, category, and content).
- Conditional rendering of edit/delete options if logged in as admin.


Search

FREQUENTLY ASKED QUESTIONS

Latest

[Question] [Category]
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Praesent porta ornare placerat. Sed tortor ex, lacinia in nulla ultrices, tristique pellentesque nisi. Vivamus dui magna, congue eu ante et, fermentum scelerisque augue. Sed enim elit, interdum at iaculis at, sollicitudin eget ex. Cras id ante pretium, consequat orci in, placerat tellus.
by [user-name]
[date uploaded]

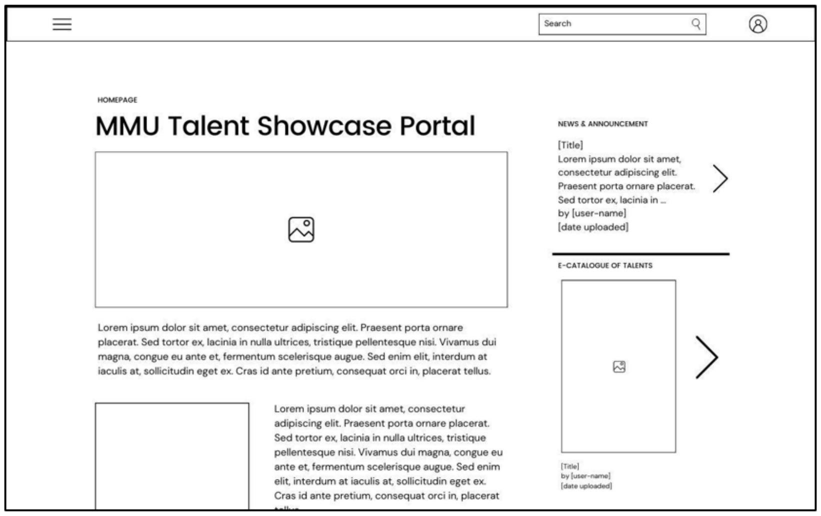
[Question] [Category]
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Praesent porta ornare placerat. Sed tortor ex, lacinia in nulla ultrices, tristique pellentesque nisi. Vivamus dui magna, congue eu ante et, fermentum scelerisque augue. Sed enim elit, interdum at iaculis at, sollicitudin eget ex. Cras id ante pretium, consequat orci in, placerat tellus.
by [user-name]
[date uploaded]

[Question] [Category]
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Praesent porta ornare placerat. Sed tortor ex, lacinia in nulla ultrices, tristique pellentesque nisi. Vivamus dui magna, congue eu ante et, fermentum scelerisque augue. Sed enim elit, interdum at iaculis at, sollicitudin eget ex. Cras id ante pretium, consequat orci in, placerat tellus.
by [user-name]
[date uploaded]

3.3.14 index.php

This is the homepage of the talent portal. It gives a quick overview of what the platform is about and navigational access to key features. The layout includes:

- A hero banner with a welcome message.
- Navigation bar
- Introduction or mission statement.
- Featured talents or latest announcements section.
- Footer with contact information and social links.



Lorem ipsum dolor sit amet, consectetur adipiscing elit. Praesent porta ornare placerat. Sed tortor ex, lacinia in nulla ultrices, tristique pellentesque nisi. Vivamus dui magna, congue eu ante et, fermentum scelerisque augue. Sed enim elit, interdum at iaculis at, sollicitudin eget ex. Cras id ante pretium, consequat orci in, placerat tellus.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Praesent porta ornare placerat. Sed tortor ex, lacinia in nulla ultrices, tristique pellentesque nisi. Vivamus dui magna, congue eu ante et, fermentum scelerisque augue. Sed enim elit, interdum at iaculis at, sollicitudin eget ex. Cras id ante pretium, consequat orci in, placerat

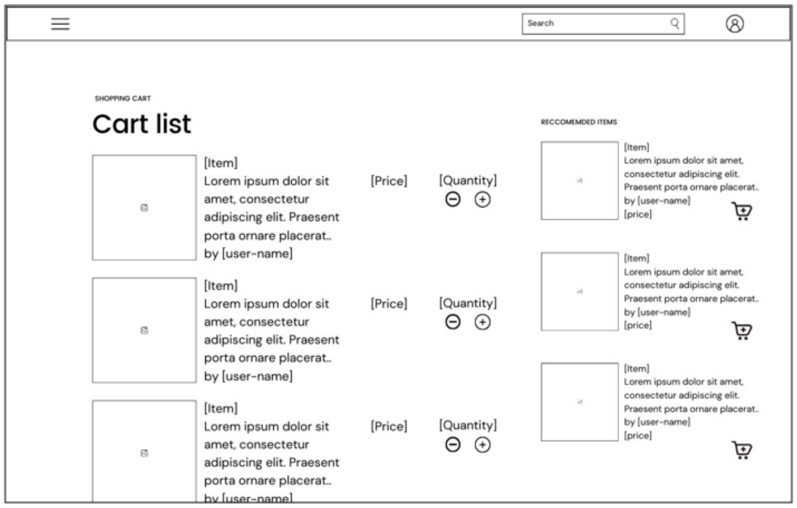
[Title]
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Praesent porta ornare placerat. Sed tortor ex, lacinia in ...
by [user-name]
[date uploaded]

[Title]
by [user-name]
[date uploaded]

3.3.15 cart.php

This page displays a list of selected gig offerings by users (e.g. digital art, music commissions). Users can review items before checking out. The layout includes:

- Cart item list
- Total cost calculation.
- Checkout button



[Item]
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Praesent porta ornare placerat..
by [user-name]

[Item]
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Praesent porta ornare placerat..
by [user-name]

[Item]
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Praesent porta ornare placerat..
by [user-name]

[Item]
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Praesent porta ornare placerat..
by [user-name]
[price]

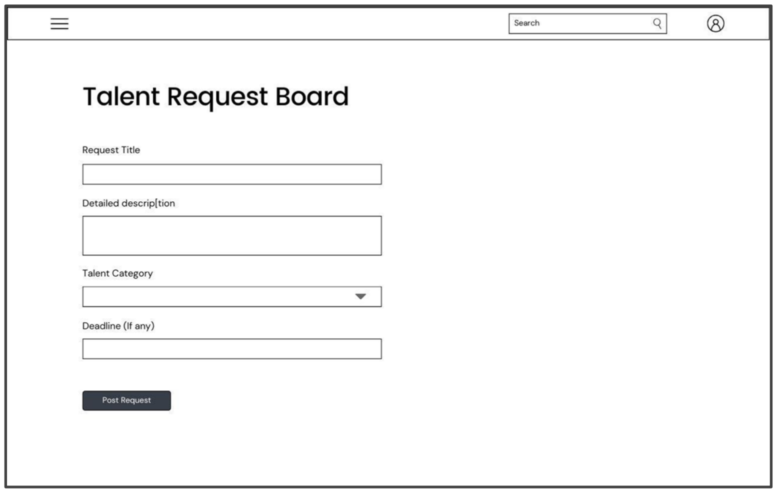
[Item]
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Praesent porta ornare placerat..
by [user-name]
[price]

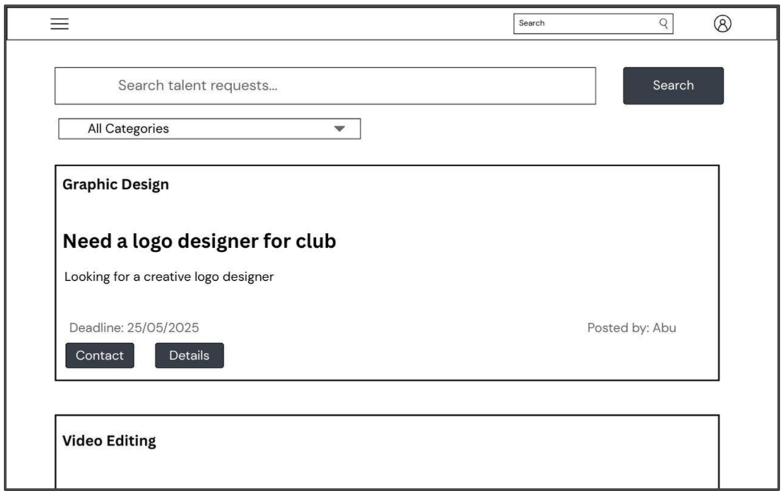
[Item]
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Praesent porta ornare placerat..
by [user-name]
[price]

3.3.16 request_board.php

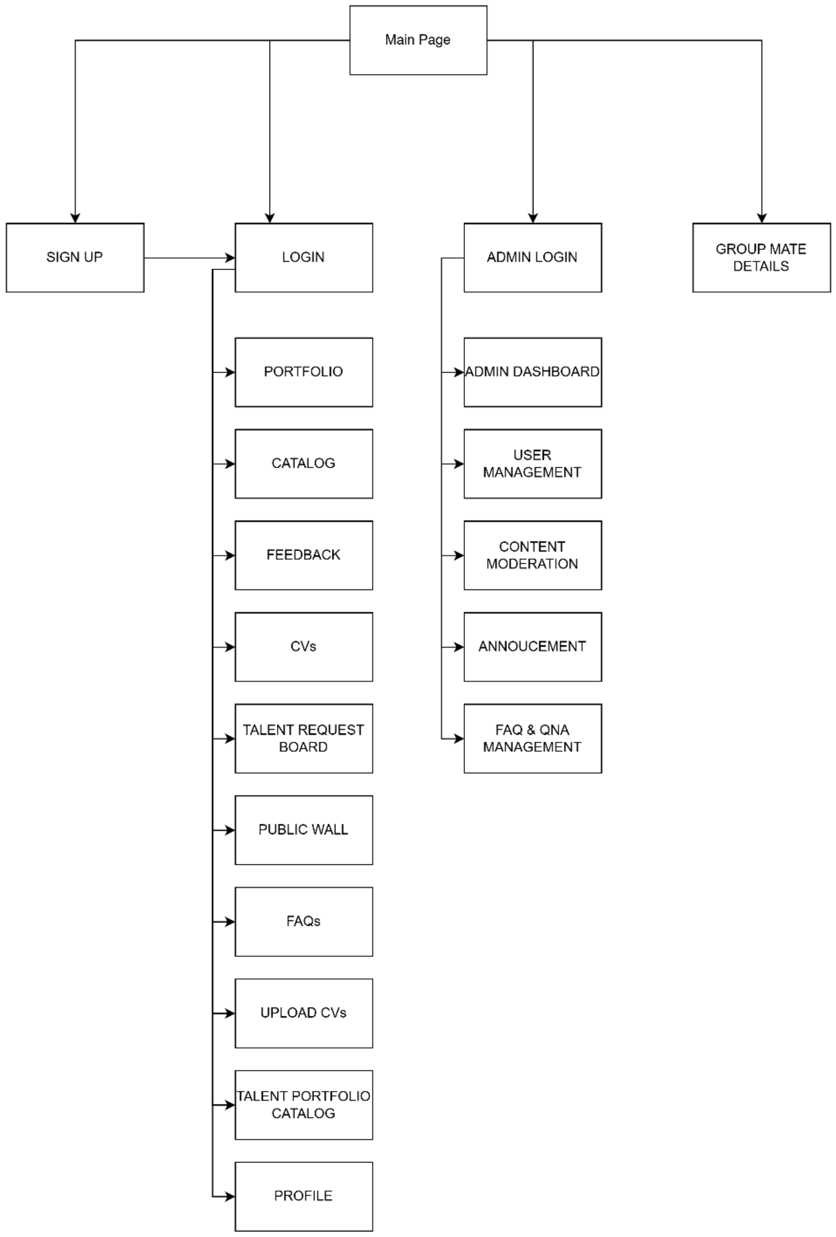
This page functions as a public board where users can post and respond to talent requests. The layout includes:

- Form to create a new request (title, description, category, contact method).
- List of existing requests with pagination.
- Optional reply section (basic comment-style interaction).



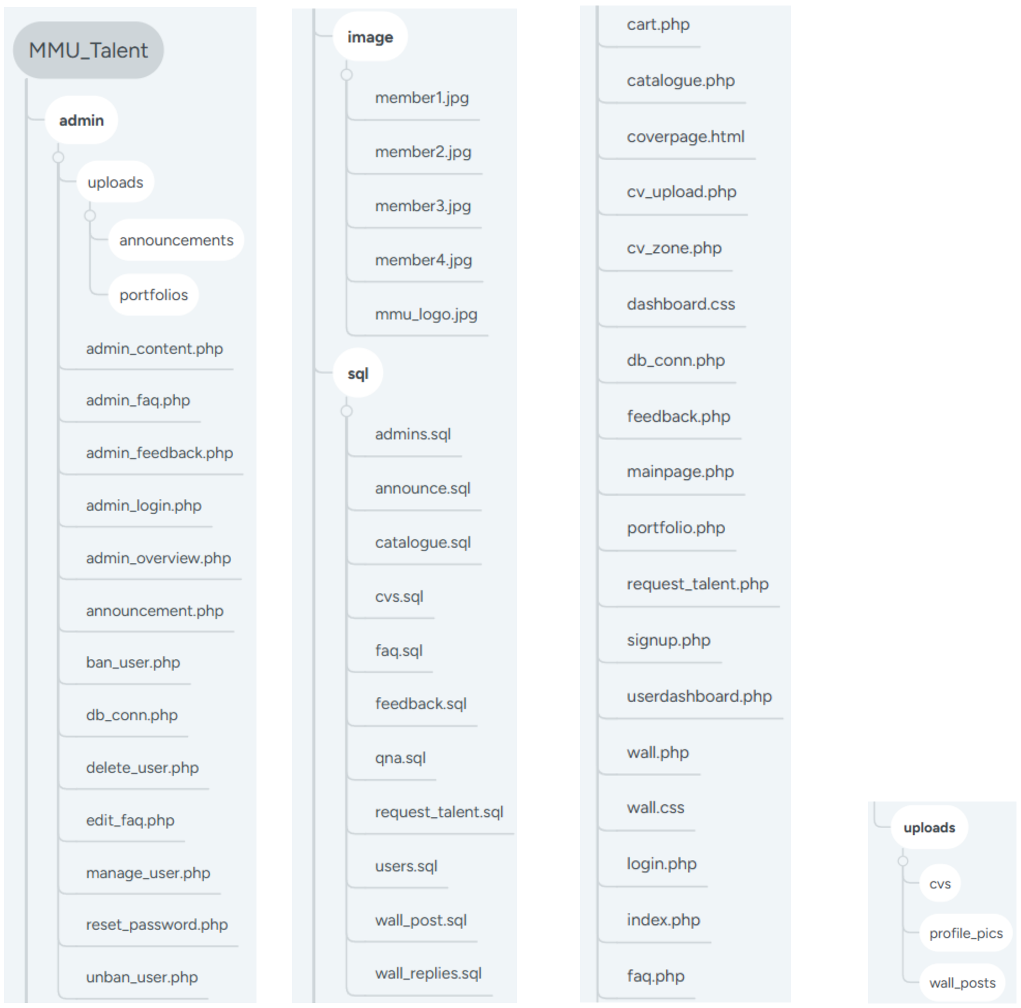


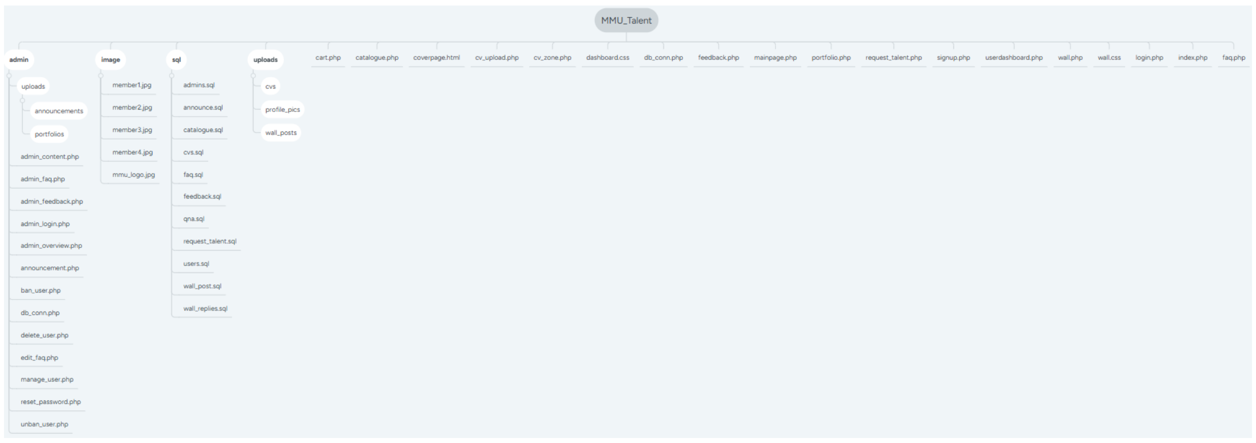
3.4 Website Navigational Structure



4.0 System Implementation

4.1 Overview of Project Structure (Files/Folders)





Explanation of the Structure

- **MMU_Talent/**: This is **root directory** for the entire project.
- **admin/**: This directory contains all the backend scripts and interfaces specifically for administrators of the MMU Talent platform. This is where admin-specific tasks like managing content, users, and announcements are handled.
- **image/**: This directory is dedicated to storing static image assets used across website, such as member photos and the MMU logo.
- **sql/**: This directory is for database-related files, primarily SQL scripts for creating tables or for initial data population.
- **uploads/**: This is a crucial directory for user-generated content. It's further organized into subdirectories:
  - **cvs/**: For uploaded CVs.
  - **profile_pics/**: For user profile pictures.
  - **wall_posts/**: For images or media associated with wall posts.
- **Root PHP Files (.php)**: These are the main front-facing PHP scripts that handle various functionalities for regular users, such as:
  - cart.php: Handles shopping cart functionalities.
  - catalogue.php: Displays a catalogue of talents.
  - cv_upload.php: Allows users to upload their CVs.
  - cv_zone.php: Likely a section for users to manage their CVs.
  - feedback.php: For user feedback submission.
  - mainpage.php: The primary entry point or homepage for general users.
  - portfolio.php: Displays talent portfolios.
  - request_talent.php: Allows users to request specific talents.
  - signup.php: Handles user registration.
  - userdashboard.php: The dashboard for regular users.
  - wall.php: Likely a social wall or activity feed.
- **CSS Files (.css)**: These files contain the styling rules for your web pages.
  - dashboard.css: Styling specific to the dashboard.

    - wall.css: Styling for the wall feature.
- **HTML File (.html)**:
    - coverpage.html: Potentially a static cover page or landing page.
- **Common Database Connection File (db_conn.php)**: This file, present in both the root and admin directories, used to establish a connection to database from various parts of the application. It's common to have a single, centralized db_conn.php included where needed.

This structure clearly separating administrative functions, static assets, database schemas, and user-uploaded content.

4.2 Implementation of Major Features

4.2.1 e-Catalogue

4.2.1.1 catalogue.php

```
START

// Step 1: Initialize the page and retrieve user input (search query and category filter)
If a GET request is made with search and/or category filters:
  Set searchQuery to value of search parameter (default to empty if not provided)
  Set categoryFilter to value of category parameter (default to empty if not provided)

// Step 2: Query the database to get the list of talents (catalogue items)
Prepare SQL Query to select * from catalogue where talent_category matches the searchQuery
If categoryFilter is provided:
  Add condition to query to filter by the categoryFilter
Execute query to retrieve matching catalogues

// Step 3: Display the retrieved catalogue items
If there are catalogue items:
  For each item in the catalogue list:
    Display the talent information (image, title, category, description, tags, etc.)
    If the file is an image:
      Display the image
    If the file is a video:
      Display the video
    If the file is an audio:
      Display the audio controls
    If the file is a PDF:
      Provide a link to view the PDF

Else:
  Display a message indicating no matching talents were found

END
```

4.2.2 User Profile Maintenance

4.2.2.1 login.php

```
START

// Step 1: Check if user is already logged in
IF user is not logged in
  Redirect to the login page
  EXIT

// Step 2: Retrieve user data from the database
TRY
  Prepare SQL query to fetch user data based on user ID
  Execute query
  Fetch user data into user variable
```

```
CATCH error
  Set error message as "Failed to fetch user data"

// Step 3: Handle POST request for user login
IF request method is POST
  Capture user input (email and password)

  TRY
    Prepare SQL query to check if user exists based on email
    Execute query
    Fetch user data from database

    // Step 4: Verify user credentials
    IF user exists
      IF entered password matches stored password
        IF user is not banned
          Set session variables for user ID and name
          Redirect to the main page (index.php)
        ELSE
          Set error message as "Your account has been banned"
      ELSE
        Set error message as "Invalid email or password"
    ELSE
      Set error message as "Invalid email or password"
  CATCH error
    Set error message as "Error during login: [error message]"

// Step 5: Display HTML page (login form)
Display login form
```

```
// Step 6: Show or hide password functionality
IF password field is clicked
  TOGGLE password visibility
  UPDATE show/hide text in the button

END
```

4.2.2.2 mainpage.php

```
START

// Step 1: Display login page if user is not logged in
IF session variable for user_id is not set
  Redirect user to the login page
  EXIT

// Step 2: Handle POST request for user login
IF request method is POST and login details are submitted
  Capture user input for email and password

  // Step 3: Validate the user email and password against the database
  TRY
    Prepare SQL query to select user based on email
    Execute query
    Fetch user data from the database
  CATCH error
    Display error message "Error fetching user data"
  // Step 4: Check if user exists
  IF user exists
    IF entered password matches stored password
      IF user is not banned
        Set session variables for user_id and user_name
        Redirect to index page
      ELSE
        Display error message "Your account has been banned"
    ELSE
      Display error message "Invalid email or password"
  ELSE
    Display error message "Invalid email or password"

// Step 5: Display HTML page (login form)
Display the login form with options for login, signup, and admin login

// Step 6: Allow user to toggle password visibility
IF showHidePass button is clicked
  Toggle password field visibility
  Update button text to 'Show' or 'Hide' based on password visibility

END
```

4.2.2.3 signup.php

```
START

// Step 1: Check if the form is submitted via POST method
IF form is submitted via POST
```

```
  // Step 2: Capture form data (name, student ID, email, password, talent category)
  Capture name, student_id, email, password, talent_category

  // Step 3: Check if the email is already registered
  Prepare SQL query to check if email exists in the database
  Execute query
  IF email exists
    Display error: "Email is already registered! Please use a different email."
  ELSE
    // Step 4: Insert user registration data into the database
    Prepare SQL query to insert data (name, student_id, email, password, talent_category)
    Execute query
    Display success message: "Registration successful! Please verify your account by logging in."
    Redirect to login page

// Step 5: Handle errors in database queries
CATCH error
  Display error message with error details

// Step 6: Display registration form
Display registration form with fields for name, student_id, email, password, and talent_category

// Step 7: Validate email format
IF email format is invalid
  Display alert: "Please enter a valid email address."
  RETURN false
```

```
END
```

4.2.2.4 userdashboard.php

```
START

// Step 1: Check if the user is logged in
IF user is not logged in (i.e., $_SESSION['user_id'] is not set)
  Redirect to login page

// Step 2: Retrieve user information from the database
Prepare SQL query to fetch user information (name, email, etc.)
Execute the query
IF query fails
  Display error message

// Step 3: Handle form submission for updating user profile
IF form is submitted via POST
  Capture form inputs (bio, email, talent category, and profile picture)

  // Step 4: Validate profile picture upload
  IF profile picture is uploaded
    Validate the file type (only image files are allowed)
    IF file is valid
      Save the uploaded file to the server
      Update the database with the new profile picture path
    ELSE
      Set error message for invalid file type

  // Step 5: Validate email format
  IF email is not in a valid format
    Set error message for invalid email

  // Step 6: Update user details in the database
  IF no errors
    Prepare SQL query to update the user profile (bio, email, talent category, profile picture)
    Execute the query
    IF query is successful
      Set success message: "Profile updated successfully!"
    ELSE
      Set error message for failed update

// Step 7: Display success or error messages
IF there is a success message
  Display success message

IF there is an error message
  Display error message

// Step 8: Display the user profile page with form for updating details
Display the user's current information (bio, talent category, profile picture) in the form

END
```

4.2.2.5 ban_user.php

```
START

// Step 1: Check if admin is logged in
IF admin is not logged in
```

```
  Redirect to admin login page

// Step 2: Check if 'id' parameter exists in the URL
IF 'id' parameter exists
  Get the user ID from the 'id' parameter
  // Step 3: Ban the user by updating their status in the database
  Attempt to execute a database query:
    Update the 'status' of the user with the given ID to 'banned'
    IF the user is successfully banned
      Set success message: "User ID [user_id] has been banned"
    ELSE
      Set error message: "User ID [user_id] not found or already banned"
  Catch database error
    Set error message: "Error banning user: [error_message]"

// Step 4: If no 'id' parameter exists
IF 'id' parameter does not exist
  Set error message: "No user ID specified for banning"

// Step 5: Redirect to the user management page
Redirect to 'manage_users.php' page

END
```

4.2.2.6 delete_user.php

```
START

// Step 1: Check if admin is logged in
IF admin is not logged in
```

```
  Redirect to admin login page

// Step 2: Check if 'id' parameter exists in the URL
IF 'id' parameter exists
  Get the user ID from the 'id' parameter

  // Step 3: Attempt to delete the user from the database
  TRY:
    Execute a database query to delete the user with the given ID
    IF user is successfully deleted
      Set success message: "User ID [user_id] and associated data have been permanently
deleted."
    ELSE
      Set error message: "User ID [user_id] not found or could not be deleted."
  CATCH database error:
    Set error message: "Error deleting user: [error_message]"

ELSE
  // Step 4: Handle case where no 'id' parameter is passed
  Set error message: "No user ID specified for deletion."

// Step 5: Redirect to the user management page
Redirect to 'manage_users.php' page

END
```

4.2.2.7 reset_password.php

```
START
```

```
// Step 1: Check if the admin is logged in
IF admin is not logged in
  Redirect to admin login page
  Exit

// Step 2: Retrieve user ID from the request
SET user_id to the value passed in the URL (GET parameter)

// Step 3: Validate if user ID is greater than zero
IF user_id is greater than zero
  // Step 4: Generate a random temporary password for the user
  SET characters to
'0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ!@#$%^&*(
)_+'
  SET random_password to an empty string
  FOR i from 0 to 7
    Append a random character from the 'characters' string to random_password
  END FOR

  // Step 5: Update the user's password in the database with the generated random
password
  TRY
    Prepare SQL query to update the user's password in the 'users' table using the generated
random password
    Execute the query with user_id and random_password
    IF the password is successfully updated
      // Step 6: Provide a success message to the admin
      Set success message "Password for user ID [user_id] has been successfully reset"
      Set new temporary password message with the random password generated
    ELSE
      // Step 7: If user ID not found or password not updated, set error message
      Set error message "Could not find user ID [user_id] or password was not updated"
  CATCH any database error
    // Step 8: If there is a database error, set error message
    Set error message "Database error: [error_message]"

ELSE
  // Step 9: If user ID is invalid, set error message
  Set error message "Invalid user ID provided"

// Step 10: Redirect to user management page
Redirect to manage_users.php
END
```

4.2.2.8 unban_user.php

```
START

// Step 1: Check if the admin is logged in
IF admin is not logged in
  Redirect to admin login page
  Exit

// Step 2: Retrieve user ID from the URL (GET parameter)
SET user_id to the value passed in the URL (GET parameter)

// Step 3: Validate if the user ID is provided
```

```
IF user_id is greater than zero
  // Step 4: Try to unban the user by updating their status to 'active'
  TRY
    Prepare SQL query to update the user's status to 'active' in the 'users' table
    Execute the query with user_id
    IF the status update is successful
      // Step 5: Set success message
      Set success message "User ID [user_id] has been unbanned."
    ELSE
      // Step 6: If user ID not found or already active, set error message
      Set error message "User ID [user_id] not found or already active."
  CATCH any database error
    // Step 7: If there is a database error, set error message
    Set error message "Error unbanning user: [error_message]"
ELSE
  // Step 8: If no user ID is provided, set error message
  Set error message "No user ID specified for unbanning."

// Step 9: Redirect to the user management page
Redirect to manage_users.php
END
```

4.2.3 Resource Sharing

4.2.3.1 cv_upload.php

```
START

// Step 1: Check if the user is logged in
IF user is NOT logged in:
  Redirect to the login page
```

```
// Step 2: Get the form inputs if the request method is POST
IF POST request is made:
    Get the CV name (category)
    Get the file upload (cv_file)
    Check if the file is valid (JPG, PNG, PDF, DOCX)

// Step 3: Validate the file type and size
IF file type is invalid OR file size exceeds limit:
    Show error message: "Invalid file type or file too large"
    STOP process

// Step 4: Create the target directory for storing CV
IF target directory does not exist:
    Create the directory

// Step 5: Move the uploaded file to the target directory
IF file is successfully moved:
    Store the file path in the database
    Store user information (user_id, CV name, file path)
    Show success message: "CV uploaded successfully!"
ELSE:
    Show error message: "Failed to upload file"

// Step 6: Provide file download link and show uploaded CV information
IF CV is successfully uploaded:
    Display the CV name and download link

END
```

4.2.3.2 cv_zone.php

```
START

// Step 1: Check if the user is logged in
IF user is NOT logged in:
  Redirect user to login page

// Step 2: Get search and category filter input from GET request
IF search query is provided:
  Set search term as query parameter

IF category filter is provided:
  Set category filter as query parameter

// Step 3: Execute database query to retrieve matching CVs
Construct SQL query:
  - SELECT CV ID, CV Name, CV Path, User Name FROM CVs
  - Filter by search term and category if applicable
  - ORDER by creation date (DESC)

// Step 4: Fetch and display CVs
IF matching CVs found:
  Display CV details including:
    - CV Name
    - Uploaded By (User Name)
    - CV Category
    - CV Path (Download link)
  Allow user to download the CV
```

```
ELSE:
  Display message: "No CVs found."

// Step 5: Handle user action (e.g., Show Details button)
IF user clicks "Show Details":
  Toggle visibility of CV details (description and download link)

// Step 6: Allow user to download CV
IF user clicks "Download CV":
  Initiate download of the CV file

END
```

4.2.4 Communication Features

4.2.4.1 wall.php

```
START

// Step 1: Check if user is logged in
IF user is not logged in (i.e., $_SESSION['user_id'] is not set)
  Redirect to login page

// Step 2: Handle new wall post submission
IF form is submitted for a new post (POST request with 'submit_post')
  IF user is logged in
    Capture the post message from the form
    IF post message is empty
```

```
      Set error message: "Wall post cannot be empty!"
    ELSE
      IF post contains a file (e.g., image)
        Validate file type (JPG, JPEG, PNG, GIF)
        IF file type is valid
          Save the file to the server directory
          Store file path in the database
        ELSE
          Set error message: "Invalid file type"
      END IF

      IF no errors
        Insert the post into the database (message and file path)
        Set success message: "Your post has been added to the Public Wall!"
      END IF
    END IF
  ELSE
    Set error message: "You must be logged in to post on the Public Wall."
  END IF
END IF

// Step 3: Handle new reply submission
IF form is submitted for a reply (POST request with 'submit_reply')
  IF user is logged in
    Capture the reply message from the form
    Capture the post ID being replied to
    IF reply message is empty
      Set error message: "Reply message cannot be empty!"
    ELSE
```

```
      Insert the reply into the database (post ID, user ID, and reply message)
      Set success message: "Your reply has been added!"
    END IF
  ELSE
    Set error message: "You must be logged in to reply to a post."
  END IF
END IF

// Step 4: Handle post deletion
IF request to delete a post (GET request with 'delete_post' action)
  IF user is logged in
    IF user is admin or user is the post owner
      Delete the post from the database
      IF post contains a photo
        Delete the photo file from the server
      END IF
      Set success message: "Post deleted successfully!"
    ELSE
      Set error message: "You do not have permission to delete this post."
    END IF
  ELSE
    Set error message: "You must be logged in to delete a post."
  END IF
END IF

// Step 5: Fetch all wall posts and their replies
Retrieve all wall posts from the database (order by creation date)
For each post
  Fetch replies associated with the post (order by creation date)
```

```
END FOR

// Step 6: Display messages (success or error)
IF there is a success message
  Display the success message
IF there is an error message
  Display the error message

// Step 7: Render the wall posts and replies
For each wall post
  Display the post message and associated data (user, timestamp)
  IF the post contains a photo
    Display the photo
  END IF

  Display replies for each post
  Display reply form for logged-in users
END FOR

END
```

4.2.5 Shopping Cart

4.2.5.1 cart.php

```
START

// Step 1: Initialize variables for success message and error handling.
Set successMessage to empty string
Set errorMessage to empty string
```

```
// Step 2: Handle buy button press.
IF POST request is made and 'buy_id' exists:
   Retrieve the buy_id from POST data
   Display successMessage with the contact number for payment

// Step 3: Fetch catalogue items from database.
Try to execute SQL query to fetch catalogue items (title, description, catalogue_file, tags,
created_at).
Store the result in $catalogues array.
IF an error occurs:
   Set $catalogues to an empty array.

// Step 4: Display the catalogue items.
Display the title "Available Services"
IF there are catalogue items:
   FOR each catalogue item in $catalogues:
      Display image, video, or PDF file based on file type
      Show title, description, tags for each item
      Display the "Buy" button (with confirmation)
ELSE:
   Display "No catalogue items found."

// Step 5: Handle confirmation for buying an item.
When the form is submitted:
   Confirm with the user if they are sure they want to buy the item.
   IF the user confirms:
      Proceed with the action (display message with payment contact).
   ELSE:
      Prevent form submission.
```

```
END
```

4.2.6 Feedback Form

4.2.6.1 feedback.php

```
START

// Step 1: Check if the user is logged in
IF the session is not set (user_id is not present)
   REDIRECT to login page
   STOP

// Step 2: Fetch user information from the database
TRY
   PREPARE SQL query to fetch user details (name, email) from the database using user_id
   EXECUTE query
   FETCH user data into $user
CATCH database error
   DISPLAY error message "Failed to fetch user data"

// Step 3: Handle form submission (Feedback submission)
IF form is submitted via POST
   GET the subject, message from POST request
   GET the user_id from session

   // Step 4: Validate the feedback message
   IF the feedback message is empty
      SET error to "Wall post cannot be empty"
   ELSE
```

```
    // Step 5: Insert the feedback into the database
    TRY
      PREPARE SQL query to insert feedback into the feedback table (subject, message,
user_id)
      EXECUTE the query to insert data
      SET success message to "Thank you for your feedback!"
    CATCH database error
      SET error message "Failed to submit feedback: " + error message
  END IF

END IF

// Step 6: Fetch existing feedback for display
TRY
  QUERY to fetch all feedback from the database
  FETCH all feedback entries into $feedbackList
CATCH database error
  SET error message "Failed to load feedback"

// Step 7: Display feedback to the user
IF there is feedback in $feedbackList
  FOR each feedback entry in $feedbackList
    DISPLAY feedback (subject, message, admin response if available)
  END FOR
ELSE
  DISPLAY message "No feedback submitted yet."

END IF
```

```
END
```

4.2.7 FAQ Section

4.2.7.1 faq.php

```
START


// Step 1: Check if user is logged in
IF user is NOT logged in:
  Redirect user to login page


// Step 2: Handle form submission for a user question
IF POST request is made and user has submitted a question:
  Get the question submitted by user
  IF user is logged in:
    Set user_id from session
    Prepare SQL query to insert question into the database:
      INSERT INTO qna (user_id, question)
      Execute the query with user_id and user question
    IF insertion is successful:
      Set success message to "Question submitted, our admin will reply soon."
    ELSE:
      Capture any database errors and set error message


  ELSE:
    Set error message to "Please log in to submit a question."


// Step 3: Fetch FAQ data from the database
TRY:
  Prepare and execute query to select question and answer from the FAQ table
```

```
   Fetch all FAQ records and display them
CATCH any errors:
   Display error message

// Step 4: Fetch user questions (Q&A) from the database
TRY:
   Prepare and execute query to select user questions and replies
   Fetch all user-submitted questions and display them
CATCH any errors:
   Display error message

// Step 5: Display the FAQ data and user-submitted questions in HTML
Display FAQs with "Show Answer" button
For each question in the FAQ:
   - Display the question and hide the answer by default
   - When user clicks "Show Answer", toggle the visibility of the answer

Display the form for submitting new user questions
Display success message if the user question is submitted successfully

END
```

4.2.7.2 admin_faq.php

```
START

// Step 1: Check if the user is logged in as an admin
IF admin is not logged in (i.e., $_SESSION['admin_id'] is not set)
   Redirect to admin login page
```

```
    EXIT

// Step 2: Initialize variables for messages and errors
SET message = ""
SET error = ""

// Step 3: Check if a new FAQ is being submitted
IF $_SERVER["REQUEST_METHOD"] is POST AND submit_faq is set
    GET faq_question and faq_answer from POST data
    TRY to insert the new FAQ into the database
        IF successful
            Set success message: "New FAQ added successfully!"
            Redirect to admin FAQ page
        ELSE
            Set error message with the database error

// Step 4: Check if a delete action is requested
IF $_GET['action'] is 'delete_faq' AND $_GET['id'] is set
    GET the FAQ ID and type from $_GET
    SET the table and ID column based on the type (faq or qna)

    TRY to delete the FAQ or Q&A entry from the database
        IF successful
            Set success message: "FAQ or Q&A deleted successfully!"
        ELSE
            Set error message with the database error
    END TRY

// Step 5: Fetch all FAQs from the database
```

```
EXECUTE SQL query to fetch all FAQs from the 'faq' table
STORE results in $faqs

// Step 6: Fetch all Q&A entries from the database
EXECUTE SQL query to fetch all Q&A entries from the 'qna' table
STORE results in $qna_list

// Step 7: Display the FAQ management section
IF $faqs is not empty
  FOR each faq in $faqs
    DISPLAY FAQ question and answer (with a preview for long answers)
    PROVIDE options to edit or delete the FAQ
  END FOR
ELSE
  Display message: "No FAQs available."

// Step 8: Display the User Q&A section
IF $qna_list is not empty
  FOR each qna in $qna_list
    DISPLAY question, user details, and response (if available)
    Display status of the question (answered or pending)
    PROVIDE options to reply or delete the Q&A
  END FOR
ELSE
  Display message: "No user questions available."

// Step 9: End
END
```

4.2.7.3 edit_faq.php

```
START

// Step 1: Check if the admin is logged in
IF admin is not logged in
  Redirect to login page

// Step 2: Get 'id' and 'type' from the GET request
IF 'id' is provided and 'type' is either 'faq' or 'qna'
  SET item_id to the value of 'id'
  SET item_type to the value of 'type'

  // Step 3: Fetch the item data based on the 'type'
  IF item_type is 'faq'
    FETCH faq data from database using item_id
  ELSE IF item_type is 'qna'
    FETCH qna data from database using item_id

  // Step 4: Check if the item data exists
  IF no data is found
    SET error message: "Item not found"

  // Step 5: Handle form submission for updating or deleting the item
  IF form is submitted with 'update_item'
    VALIDATE the input fields for question and answer
    IF validation fails
      SET error message: "Invalid input data"
    ELSE
```

```
      UPDATE the corresponding item (FAQ or Q&A) in the database with the new answer or reply

      IF update is successful
        SET success message: "Item updated successfully"
      ELSE
        SET error message: "Failed to update item"

  IF form is submitted with 'delete_item'
    DELETE the corresponding item (FAQ or Q&A) from the database

    IF delete is successful
      SET success message: "Item deleted successfully"
    ELSE
      SET error message: "Failed to delete item"

// Step 6: Display the item data on the page for editing
IF item_data is available
  Display the item question and current answer/reply in editable fields

// Step 7: Handle success or error messages
Display success or error messages

END
```

4.2.8 News & Announcement

4.2.8.1 index.php

```
START
```

```
// Step 1: Check if the user is logged in
IF user is not logged in
  Redirect to login page
  EXIT

// Step 2: Fetch recent announcements from the database
TRY
  Connect to the database
  Execute query to fetch announcements ordered by creation date
  Store fetched data in announcements array
CATCH error
  Set announcements array to empty

// Step 3: Handle user feedback submission
IF POST request is made and user is logged in
  Capture feedback details (subject, message)
  Validate fields (make sure they're not empty)
  Try to insert feedback into the database
    IF insert successful
      Set success message to "Thank you for your feedback!"
    ELSE
      Set error message to "Failed to submit feedback"

// Step 4: Render HTML page
Render page with the following:
  Display user profile icon and site title
  Display navigation bar with various links
  Display announcements section:
    IF there are announcements:
      Loop through announcements
      Display each announcement (title, message, timestamp, image)
    ELSE
      Display "No announcements available"

// Step 5: Handle image display for announcements
IF announcement contains an image
  Check if image exists in the directory
  Display image with announcement

// Step 6: Display feedback form
IF feedback submission is successful
  Display success message
IF feedback submission has error
  Display error message

// Step 7: End
END
```

4.2.8.2 announcement.php

```
START

// Step 1: Admin login validation
IF admin is not logged in
  Redirect to admin login page

// Step 2: Check if an action (edit/delete) is requested
IF action is 'delete' AND announcement ID is provided
```

```
  // Step 3: Delete the announcement
  Get announcement details by ID
  IF announcement exists
    // Step 4: Delete associated photo file if exists
    IF photo exists and file is found
      Delete photo file from server
    END
    // Step 5: Delete announcement from the database
    DELETE the announcement from the database
    Display success message
  ELSE
    Display error message (announcement not found)
  END
END

IF action is 'edit' AND announcement ID is provided
  // Step 6: Fetch announcement details for editing
  Get the announcement by ID
  IF announcement exists
    // Step 7: Display form to edit the announcement
    Show the current title, message, and photo for editing
  ELSE
    Display error message (announcement not found)
  END
END

// Step 8: Handle form submission for adding or updating an announcement
IF POST request is submitted
  Get title, message, current photo, and uploaded photo (if any)
  // Step 9: Remove photo if requested
  IF remove photo is checked
    IF current photo exists
      Delete the photo file from the server
      Set photo path to null
    END
  END
  // Step 10: Validate photo upload
  IF new photo is uploaded
    Validate file type (allow JPG, PNG, GIF, etc.)
    IF file type is valid
      Move the uploaded file to the server
      Update photo path
    ELSE
      Display error message (invalid file type)
    END
  END
  // Step 11: Insert or update the announcement in the database
  IF announcement ID exists (update case)
    Update the announcement with new details (title, message, photo)
  ELSE
    Insert the new announcement into the database
  END
  Display success message (announcement updated or posted)

// Step 12: Fetch all announcements
SELECT all announcements from the database
Display each announcement with options to edit or delete
```

```
END
```

4.2.9 Photo/File Uploads

4.2.9.1 portfolio.php

```
START


// Step 1: Check if the user is logged in
IF user is NOT logged in
    Redirect to login page
    EXIT


// Step 2: Retrieve user information from the database
TRY
    Prepare SQL query to fetch user information based on user_id
    Execute the query
    Fetch user data from the database
CATCH error
    Display error message if user data fetching fails


// Step 3: Handle the form submission
IF form is submitted (POST request)
    Capture form data (title, category, description, tags, file)


    // Step 4: Check if a file is uploaded
    IF file is uploaded
        Check the file type (only allowed file types: images, audio, video, pdf)
        IF file type is valid and file size is acceptable (<= 10MB)
            Create target directory for file upload if not exists
            Move uploaded file to the target directory
```

```
      Store the file path for database insertion

      // Step 5: Insert portfolio data into the database
      TRY
        Prepare SQL query to insert portfolio data (user_id, title, category, file_path, description, tags)
        Execute the query with the captured data
        Display success message
      CATCH error
        Display error message if the database query fails
    ELSE
      Display error message if file type or size is invalid
  ELSE
    Display error message if no file is selected

// Step 6: Retrieve all portfolio catalogues from the database
TRY
  Prepare SQL query to fetch all catalogues ordered by creation date
  Execute the query
  Fetch the catalogue data
CATCH error
  Display error message if catalogue fetching fails

// Step 7: Display the uploaded portfolios in a grid layout
FOR each portfolio in the catalogues list
  Display portfolio details (title, category, description, tags)
  Display appropriate file content (image, video, audio, or PDF)
  Display a download link for the portfolio file
```

```
END
```

4.2.10 Admin Dashboards

4.2.10.1 admin_content.php

```
START

// Step 1: Check if user is logged in as admin
IF admin is not logged in (i.e., $_SESSION['admin_id'] is not set)
    Redirect to admin login page
    EXIT

// Step 2: Initialize variables for recent posts, catalogues, and feedback
SET recent_wall_posts = []
SET recent_catalogues = []
SET recent_feedback = []

// Step 3: Fetch recent wall posts from database
EXECUTE SQL query to retrieve wall posts ordered by created_at DESC, limit to 20
STORE result in recent_wall_posts

// Step 4: Fetch recent catalogues from database
EXECUTE SQL query to retrieve catalogue details ordered by created_at DESC, limit to 20
STORE result in recent_catalogues

// Step 5: Fetch recent feedback from database
EXECUTE SQL query to retrieve feedback details ordered by submitted_at DESC, limit to 20
STORE result in recent_feedback

// Step 6: Handle content deletion request (wall post, catalogue, feedback)
```

```
IF there is a delete action requested
  CHECK the content type (wall_post, catalogue, feedback)
  SET the correct table and ID column based on the content type

  // Step 6.1: Check if the admin has permission to delete
  IF admin has permission (either as admin or post owner)
    TRY deleting the content from the corresponding table
    IF deletion is successful
      SHOW success message "Content deleted successfully!"
    ELSE
      SHOW error message "Failed to delete content"
  ELSE
    SHOW error message "No permission to delete content"
  END IF

// Step 7: Display the content on the page
FOR each wall post in recent_wall_posts
  DISPLAY post content (username, message, created_at)
  IF post contains photo
    DISPLAY photo
  END IF

FOR each catalogue in recent_catalogues
  DISPLAY catalogue details (title, description, file, category)
  IF catalogue contains file
    PROVIDE download link for the file
  END IF

FOR each feedback in recent_feedback
```

```
  DISPLAY feedback details (username, subject, message, submitted_at)
  IF feedback has admin response
    DISPLAY admin response
  END IF

// Step 8: End
END
```

4.2.10.2 admin_feedback.php

```
START

// Step 1: Check if the user is logged in as admin
IF admin is not logged in (i.e., $_SESSION['admin_id'] is not set)
  Redirect to admin login page
  EXIT

// Step 2: Initialize message and error variables
SET message = ""
SET error = ""

// Step 3: Check if a feedback ID is provided in the URL
GET feedback ID from $_GET['id']
IF feedback ID is valid
  TRY to fetch feedback from database based on the ID
    IF feedback not found
      SET error = "Feedback item not found"
  ELSE
    SET error = "No feedback ID provided"
END IF
```

```
// Step 4: Handle form submission to respond to feedback
IF form is submitted (POST request with 'submit_response')
  GET admin response from POST data (admin_response_text)
  GET status (new_status) from POST data
  IF response and status are valid
    UPDATE feedback in database with admin response and new status
    SET success message = "Feedback updated successfully!"
    Redirect to the same feedback page (admin_feedback.php?id=feedback_id)
  ELSE
    SET error = "Invalid response or status"

// Step 5: Fetch all feedback items from the database
EXECUTE SQL query to fetch feedback details from 'feedback' table

// Step 6: Display feedback details on the page
IF feedback is found
  Display feedback message and details (subject, username, etc.)
  Display the admin response if available

// Step 7: Allow admin to reply to the feedback and change its status
IF feedback exists
  Display a form to submit a new admin response
  Display the current status and allow status to be changed (pending, resolved, answered)

// Step 8: Provide options to go back to the content moderation page
Display a "back to content moderation" button

END
```

4.2.10.3 admin_login.php

```
START

// Step 1: Check if the form has been submitted (POST request)
IF form is submitted (POST request)
  // Step 2: Retrieve username and password from the form
  GET username and password from the form

  // Step 3: Check if username and password are provided
  IF username and password are provided
    // Step 4: Prepare and execute the SQL query to check for admin credentials
    EXECUTE SQL query to find admin with matching username and password

    // Step 5: Check if the admin is found
    IF admin exists
      // Step 6: Set session variables for the logged-in admin
      SET session variables for admin_logged_in and admin_username

      // Step 7: Redirect to admin dashboard
      REDIRECT to admin_overview.php
      EXIT
    ELSE
      // Step 8: Set error message if admin is not found
      SET error message = "Invalid username or password."
    END IF
  ELSE
    // Step 9: Set error message if fields are empty
    SET error message = "Please fill in both fields."
```

```
   END IF
END IF

// Step 10: Display the login page with any errors (if any)
SHOW login form with error messages

END
```

4.2.10.4 admin_overview.php

```
START

// Step 1: Check if the admin is logged in
IF admin is not logged in
   Redirect to "admin_login.php"
   EXIT

// Step 2: Fetch the admin's username from the session
Get admin's username from session
IF username is not set
   Set default username as 'Admin'

// Step 3: Display the Admin Dashboard
SHOW the following:
   - Admin's name at the top of the page
   - List of available administrative tasks:
      1. Overview: Track user logins, uploads, and platform activity.
      2. User Management: Approve or ban users, reset passwords.
      3. Content Moderation: Respond and review feedback, wall posts, and portfolios.
      4. Announcements: Post public updates such as events or changes.
```

```
    5. FAQ Management: Edit, delete, or publish answers to user-submitted questions.

END
```

4.2.10.5 manage_users.php

```
START

// Step 1: Check if the admin is logged in
IF admin is not logged in
  Redirect to login page

// Step 2: Fetch user details from the database
SET user_count = fetch user data from the database

// Step 3: Display success or error message if available
IF success message exists
  Display success message

IF error message exists
  Display error message

// Step 4: If there is a user list, iterate through users
IF user list is not empty
  FOR each user in the list
    Display user's ID, name, email, and status
    IF the user is banned
      Display 'Unban' option
    ELSE
      Display 'Ban' option
    Display options to reset password, delete user, etc.

  END FOR
ELSE
  Display message: "No users found."

// Step 5: Ban user functionality
IF 'ban' option is selected for a user
  Set user status to 'banned' in the database
  Display confirmation message for banning user

// Step 6: Unban user functionality
IF 'unban' option is selected for a user
  Set user status to 'active' in the database
  Display confirmation message for unbanning user

// Step 7: Reset user password functionality
IF 'reset password' option is selected for a user
  Generate a temporary password
  Update user password with temporary password in the database
  Display success message for password reset

// Step 8: Delete user functionality
IF 'delete' option is selected for a user
  Delete user and associated data from the database
  Display success message for deleting user

END
```

4.2.10.6 announcement.php

```
// Step 1: Check if the admin is logged in
IF admin is not logged in
   Redirect to login page

// Step 2: Fetch announcements from the database
SET announcements = fetch announcement data from the database

// Step 3: Display success or error message if available
IF success message exists
   Display success message
IF error message exists
   Display error message

// Step 4: If there is an announcement list, iterate through announcements
IF announcement list is not empty
   FOR each announcement in the list
      Display announcement's ID, title, message, photo (if available), and timestamps
      Display options to edit or delete the announcement
   END FOR
ELSE
   Display message: "No announcements posted yet."

// Step 5: Post new announcement functionality
IF 'post announcement' form is submitted
   Validate title and message
   IF valid
```

```
    Upload photo (if provided)
    Insert new announcement into the database
    Display success message for posting announcement
  ELSE
    Display error message for invalid input

// Step 6: Edit announcement functionality
IF 'edit' option is selected for an announcement
  Fetch announcement details from the database
  Display announcement details in the form for editing
  IF 'update announcement' form is submitted
    Validate title and message
    IF valid
      Upload new photo (if provided) and delete old photo (if replaced)
      Update announcement in the database
      Display success message for updating announcement
    ELSE
      Display error message for invalid input

// Step 7: Delete announcement functionality
IF 'delete' option is selected for an announcement
  Delete announcement and associated photo from the database
  Display success message for deleting announcement
END
```

4.3 Implementation of Additional Features

4.3.1 Request Talent Board (request_talent.php)

```
START
```

```
// Step 1: Check if the user is logged in
IF user is NOT logged in
  Redirect to login page
  EXIT

// Step 2: Retrieve user information from the database
TRY
  Prepare SQL query to fetch user information based on user_id
  Execute query
  Fetch user data
CATCH error
  Display error message if fetching fails

// Step 3: Handle the form submission (Talent Request)
IF form is submitted (POST request)
  Capture form data: title, category, budget, deadline, requester name, email, description

  // Step 4: Validate if all fields are filled
  IF any field is empty
    Display error message: "Please fill in all fields."

  ELSE
    // Step 5: Insert talent request data into the database
    TRY
      Prepare SQL query to insert request data (title, category, budget, deadline, name, email, description)
      Execute query
      Display success message: "Request submitted successfully!"
    CATCH error
```

```
        Display error message if database insertion fails

// Step 6: Fetch the list of submitted requests
TRY
    Prepare SQL query to fetch all requests ordered by creation date
    Execute query
    Fetch all requests
CATCH error
    Display error message if fetching requests fails

// Step 7: Display the list of requests in the UI
FOR each request in the list
    Display request details (title, category, description, budget, deadline, name, email)

// Step 8: Validate the budget (must be at least RM 10)
IF budget is less than 10
    Display error: "Budget must be at least RM 10."
    STOP form submission

END
```

5.0 User Guidance

5.1 How to Use the System

To effectively use the system, follow these steps:

- Login: Access the system by entering your username (admin), email (user) and password on the login page.

- Navigation: Use the navigation menu to explore different sections of the system, such as the e-Catalogue, user profiles, and announcements.
- Search and Filter: Utilize the search bar and category filters to find specific talents or information within the e-Catalogue.
- View Details: Click on the Catalogue to view detailed information, including images, videos, and descriptions.
- Interact: Use features like the "Wall" to post updates, interact with other users, and stay informed about the latest announcements.
- Profile Management: Update your profile by adding new skills, projects, and other relevant information to showcase your talents.

5.2 Registration Process

To register for the system, follow these steps:

- Access Main Page: Navigate to the main page where you can choose to log in, sign up, or log in as an admin.
- Login: If you choose to log in, enter your email and password. If the credentials are correct, you will be directed to the user dashboard.
- Sign Up: If you haven't registered yet and want to sign up, fill in the required data on the registration form. Once completed, you can enter the portal.
- Admin Login: If you choose to log in as an admin, enter your username and password. If the credentials are correct, you will be directed to the admin overview portal.
- This process ensures that users and admins can access their respective dashboards efficiently and securely.

5.3 Adding and Editing Records

5.3.1 User Management (manage_users.php)

The User Management section allows administrators to oversee and modify user accounts.

- **Accessing User Management**: Click on "User Management" in the navigation bar.
- **Viewing Users**: A table displays all registered users with their ID, Name, Email, and current Status.
- **User Actions**: For each user, administrators can perform the following actions:
  - **Ban/Unban User**:
    - If a user's status is 'active', a "Ban" link is available to restrict their access. A confirmation prompt will appear before banning.
    - If a user's status is 'banned', an "Unban" link is available to restore their access. A confirmation prompt will appear before unbanning.
  - **Reset Password**: Click "Reset Password" to generate a temporary password for a user. A confirmation prompt will appear.
  - **Delete User**: Click "Delete" to permanently remove a user account and all associated data. A warning and confirmation prompt will appear as this action cannot be undone.
- **Messages**: Success or error messages related to user actions (e.g., "User banned successfully!", "Temporary password generated!") will be displayed at the top of the page.

5.3.2 Announcement Management (announcement.php)

- The Announcement Panel enables administrators to create, edit, and delete system-wide announcements.
- **Accessing Announcements**: Click on "Announcements" in the navigation bar.
- **Posting a New Announcement**:
  - Fill in the "Announcement Title" and "Write your announcement..." fields.
  - Optionally, "Upload Photo" to include an image with the announcement. Supported formats are JPG, JPEG, PNG, and GIF.
  - Click "Post Announcement" to publish it. A confirmation prompt will appear.
- **Editing an Existing Announcement**:
  - Locate the announcement in the "All Announcements" section.

    - Click the "Edit" link next to the announcement you wish to modify.
    - The form at the top of the page will be pre-filled with the announcement's current details.
    - You can change the title, message, or upload a new photo.
    - If there's an existing photo, you can choose to "Remove Current Photo" using the checkbox.
    - Click "Update Announcement" to save your changes. A confirmation prompt will appear.
    - Click "Cancel Edit" to discard changes and return to the main announcement view.
- **Deleting an Announcement**:
    - Locate the announcement in the "All Announcements" section.
    - Click the "Delete" link next to the announcement. A confirmation prompt will appear, warning that the action cannot be undone.
- **Messages**: Success or error messages will be displayed at the top of the page (e.g., "Announcement posted successfully!", "Announcement deleted successfully!").

5.3.3 FAQ Management (admin_faq.php)

The FAQ Management section allows administrators to manage frequently asked questions and user-submitted questions.

- **Accessing FAQ Management**: Click on "FAQ Management" in the navigation bar.
- **Adding a New FAQ**:
    - Enter the question and answer in the provided form.
    - Submit the form to add the new FAQ to the system.
- **Editing an Existing FAQ**:
    - Locate the FAQ in the "All FAQs" section.
    - Click the "Edit" option to modify its content.
- **Deleting an FAQ**:

    - Locate the FAQ in the "All FAQs" section.
    - Click the "Delete" option to remove it permanently.
- **Managing User Q&A**:
    - Review user-submitted questions in the "User Q&A" section.
    - Provide responses to pending questions.
    - Delete inappropriate or resolved user questions.
- **Messages**: Success or error messages will be displayed regarding FAQ and Q&A operations.

5.3.4 Content Moderation (admin_content.php)

The Content Moderation section empowers administrators to review and manage user-generated content, including wall posts, portfolios (catalogues), and feedback.

- **Accessing Content Moderation**: Click on "Content Moderation" in the navigation bar.
- **Overview of Content**: This page is divided into three main sections, each displaying recent entries:
    - **Recent Wall Posts**: Shows the most recent activity on the public wall.
    - **Recent Portfolios (Catalogues)**: Displays recently uploaded user portfolios.
    - **Recent Feedback**: Lists feedback submitted by users.
- **Managing Wall Posts**:
    - **Viewing**: Each entry displays the post content, the user who posted it (name and email), and the creation timestamp.
    - **Deletion**: An "Delete" link is available next to each wall post. Clicking this will prompt a confirmation message. Upon confirmation, the wall post will be permanently removed from the system.
- **Managing Portfolios (Catalogues)**:

  - **Viewing**: Each entry shows the portfolio's title, a preview of its description, the talent category, the file name, the user who uploaded it (name and email), and the upload timestamp.
  - **Deletion**: An "Delete" link is provided for each portfolio. A confirmation message will appear to prevent accidental deletion. Deleting a portfolio permanently removes it and its associated files.
- **Managing Feedback**:
  - **Viewing**: Each entry displays the username, email, subject, a preview of the message, the submission timestamp, and the feedback status (e.g., "Pending", "Answered").
  - **Responding to Feedback**: A "Respond" link (which typically leads to admin_feedback.php) is available. Clicking this allows the administrator to view the full feedback message and potentially send a reply or mark it as resolved.
  - **Deletion**: An "Delete" link is available for each feedback entry. A confirmation message will appear before permanent deletion.
- **Purpose**: This module is crucial for maintaining the quality, safety, and relevance of content within the MMU Talent Portal, ensuring a positive environment for all users.

5.4 Searching and Filtering

5.4.1 CV Zone (cv_zone)

- Searching for CV

Users can find and filter a specific type of CV by entering keyword related to the CV's category (e.g. "Coding", "Writing") into the search field.

5.4.2 Catalogue (catalogue.php)

- Search bar

  The users can use the search bar to find portfolios by typing the category keyword.

- Filter Category

  Users can filter the portfolio list by using the dropdown menu next to the search bar and select the pre-defined categories such as “Coding”, “Design” or “Writing”. After selecting, press on the search button to filter the list.

5.5 Screenshots with Descriptions

5.5.1 Sign Up

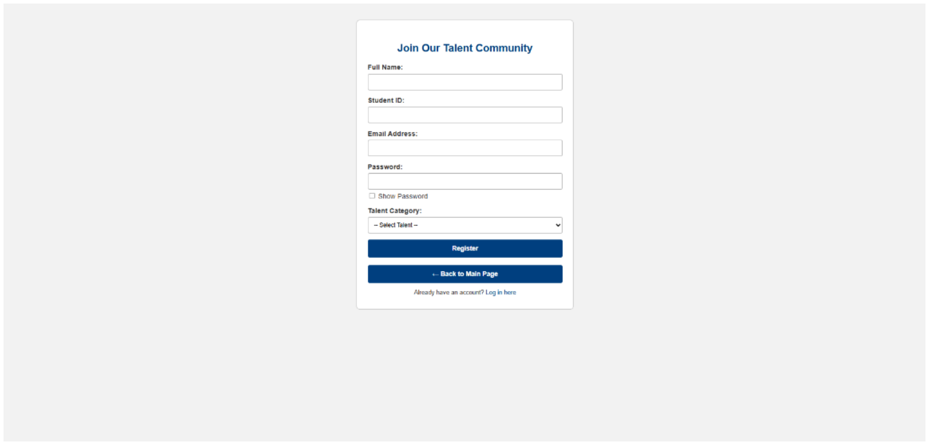


Figure 5.5.1

Figure 5.5.1 shows a sign-up page where new users are required to enter their name, student ID, email, password, and select a talent category to create an account.

5.5.2 Log In

User view :

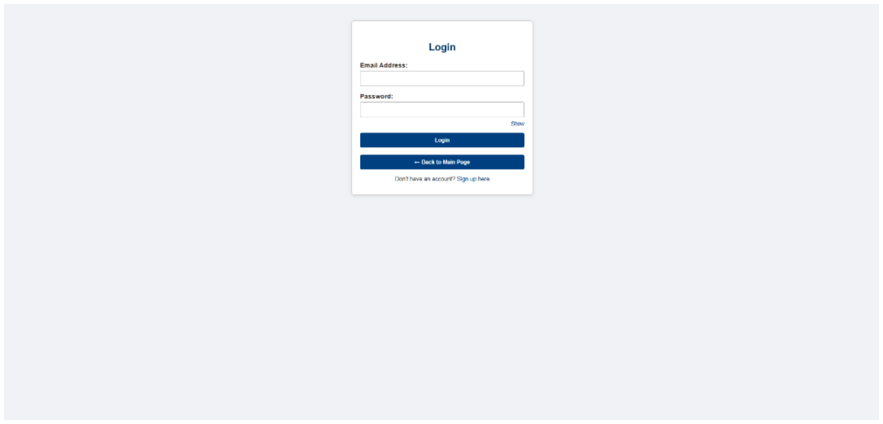


Figure 5.5.2

Figure 5.5.2 shows a login page where users are required to enter their email address and password to access the site.

Admin view :

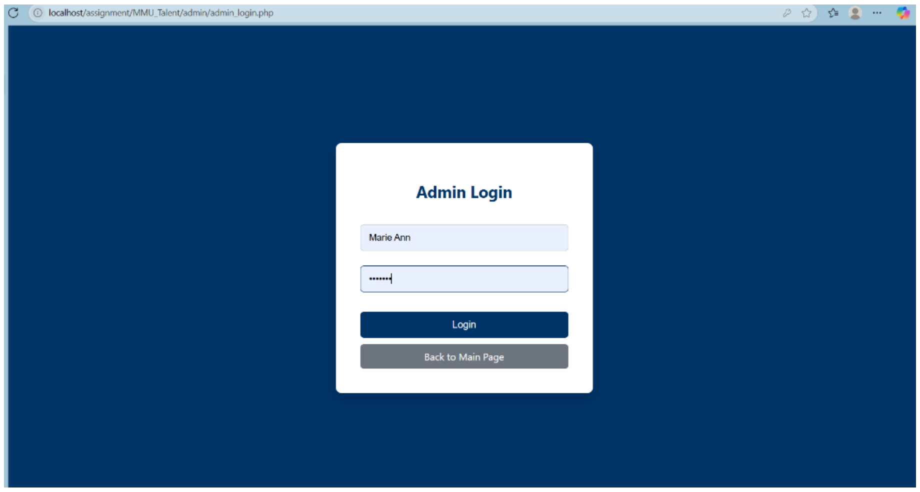
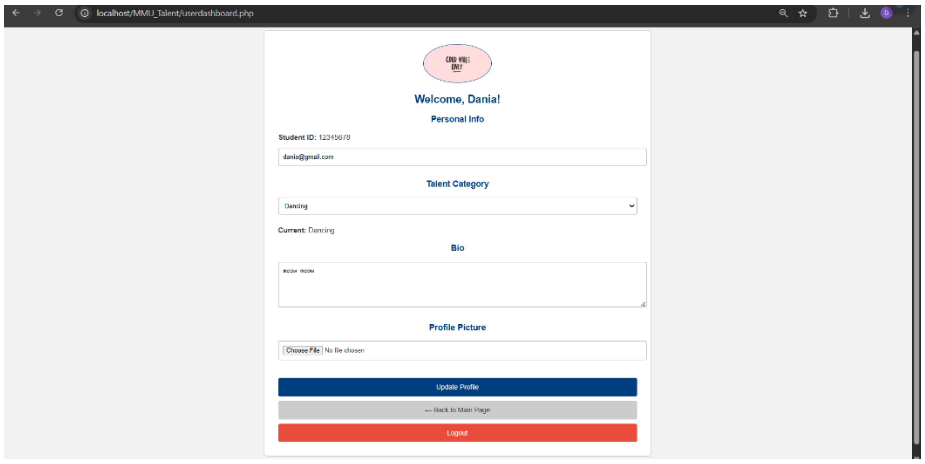

localhost/assignment/MMU_Talent/admin/admin_login.php

Figure 5.5.3

Figure 5.5.3 displays the secure administrative login interface. This page serves as the initial access point for authorized personnel, requiring valid credentials (username and password) to gain entry to the Admin Dashboard and its content management functionalities. Successful login establishes an authenticated session, redirecting the administrator to the main dashboard.

5.5.3 User dashboard


localhost/MMU_Talent/userdashboard.php

Welcome, Dania!

Personal Info

Student ID: 12345678

dania@gmail.com

Talent Category

Dancing

Current: Dancing

Bio

meow meow

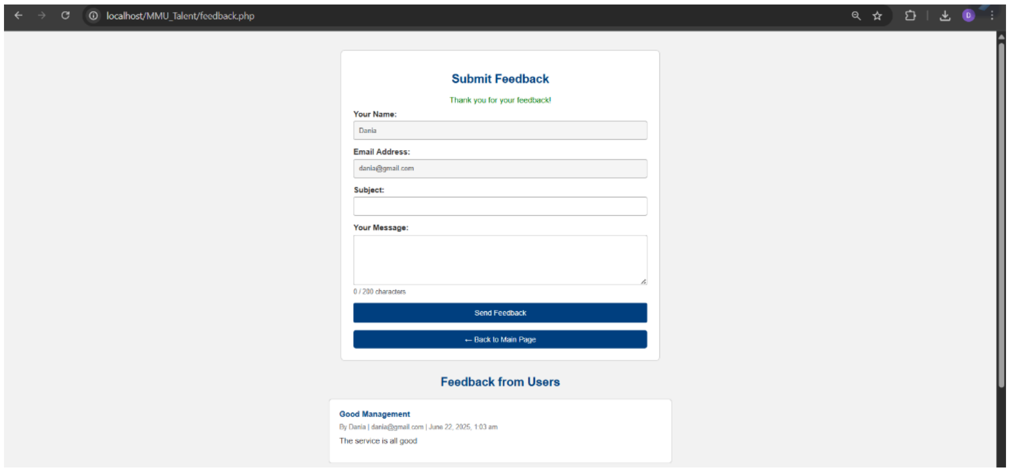
Profile Picture

Figure 5.5.4

Figure 5.5.3 shows a user dashboard with options to edit personal details, select talent category, write bio and add profile picture. Through the same page also the user can choose to logout from the user profile.

5.5.4 Feedback

User view :



localhost/MMU_Talent/feedback.php

Submit Feedback

Thank you for your feedback!

Your Name: Dania

Email Address: dania@gmail.com

Subject:

Your Message:

0 / 200 characters

Feedback from Users

Good Management

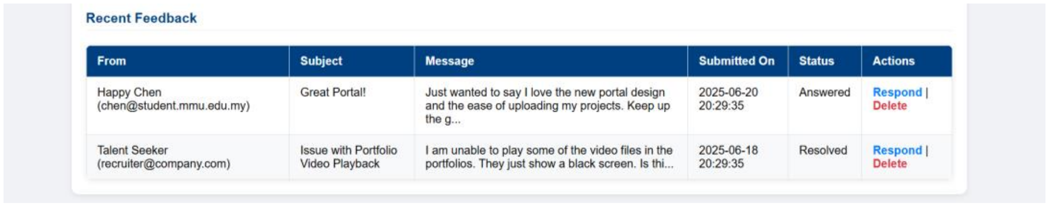
By Dania | dania@gmail.com | June 22, 2025, 1:03 am

The service is all good

Figure 5.5.5

Figure 5.5.4 shows a feedback page where users can write public feedback of the page. Users also able to see others feedback via the same page.

Admin View :



Recent Feedback

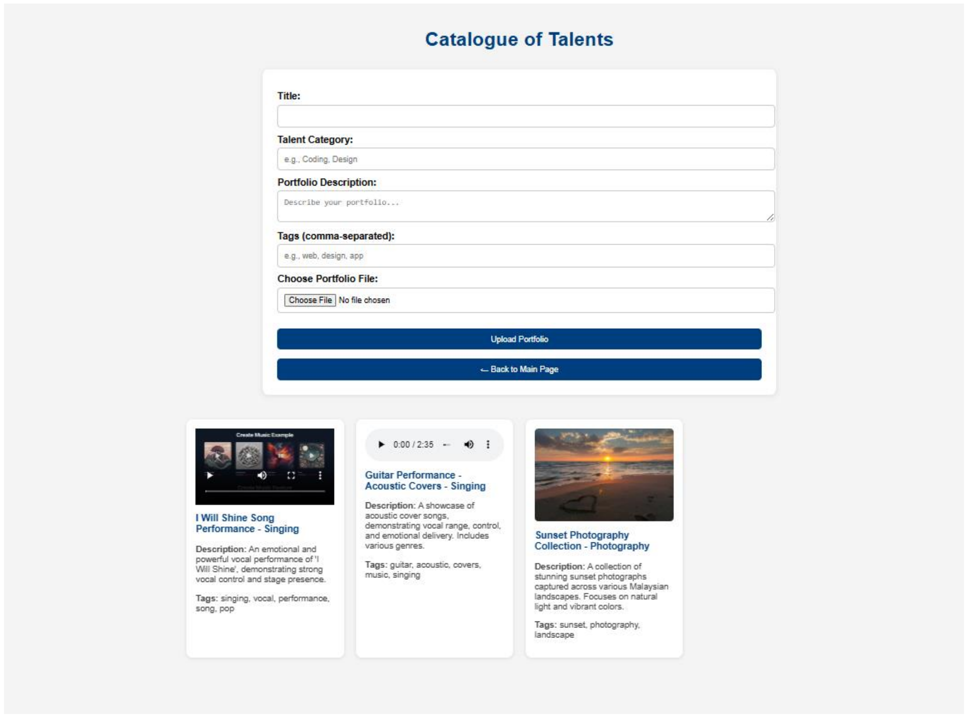
From	Subject	Message	Submitted On	Status	Actions
Happy Chen (chen@student.mmu.edu.my)	Great Portal!	Just wanted to say I love the new portal design and the ease of uploading my projects. Keep up the g...	2025-06-20 20:29:35	Answered	Respond | Delete
Talent Seeker (recruiter@company.com)	Issue with Portfolio Video Playback	I am unable to play some of the video files in the portfolios. They just show a black screen. Is thi...	2025-06-18 20:29:35	Resolved	Respond | Delete

Figure 5.5.6

Figure 5.5.6 illustrates the 'Feedback Management' section within the Admin Dashboard. This interface provides administrators with a comprehensive list of submitted user feedback, including details such as the sender's username and email, subject, message content, and submission date. It enables administrators to review feedback statuses, respond to inquiries, and take necessary actions, such as deleting irrelevant or resolved feedback entries.

5.5.5 portfolio

User View :



Catalogue of Talents

Title:

Talent Category: e.g., Coding, Design

Portfolio Description: Describe your portfolio...

Tags (comma-separated): e.g., web, design, app

Choose Portfolio File: Choose File No file chosen

I Will Shine Song Performance - Singing

Description: An emotional and powerful vocal performance of 'I Will Shine', demonstrating strong vocal control and stage presence.

Tags: singing, vocal, performance, song, pop

Guitar Performance - Acoustic Covers - Singing

Description: A showcase of acoustic cover songs, demonstrating vocal range, control, and emotional delivery. Includes various genres.

Tags: guitar, acoustic, covers, music, singing

Sunset Photography Collection - Photography

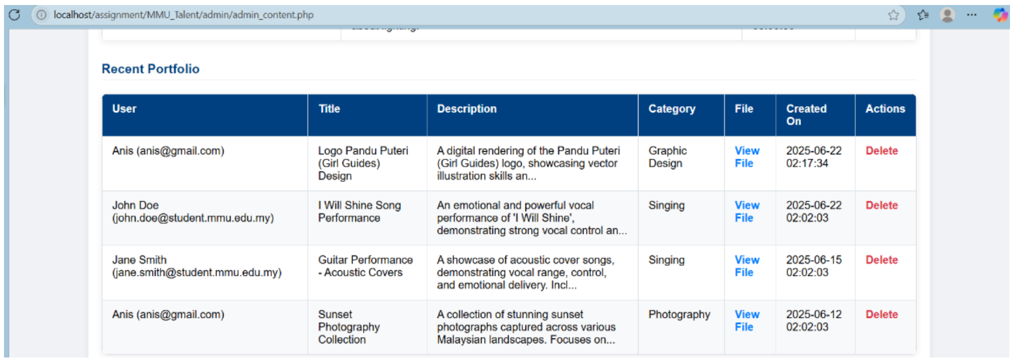
Description: A collection of stunning sunset photographs captured across various Malaysian landscapes. Focuses on natural light and vibrant colors.

Tags: sunset, photography, landscape

Figure 5.5.7

Figure 5.5.7 showcases the 'Portfolio' page, emphasizing the functionality for students to upload their creative and academic work. This screenshot captures the upload form with fields for title, category, description, and tags, alongside the ability to select various file types (e.g., images, PDFs, videos). It also illustrates how uploaded portfolios are displayed, including their title, category, description, and relevant tags.

Admin View :


localhost/assignment/MMU_Talent/admin/admin_content.php

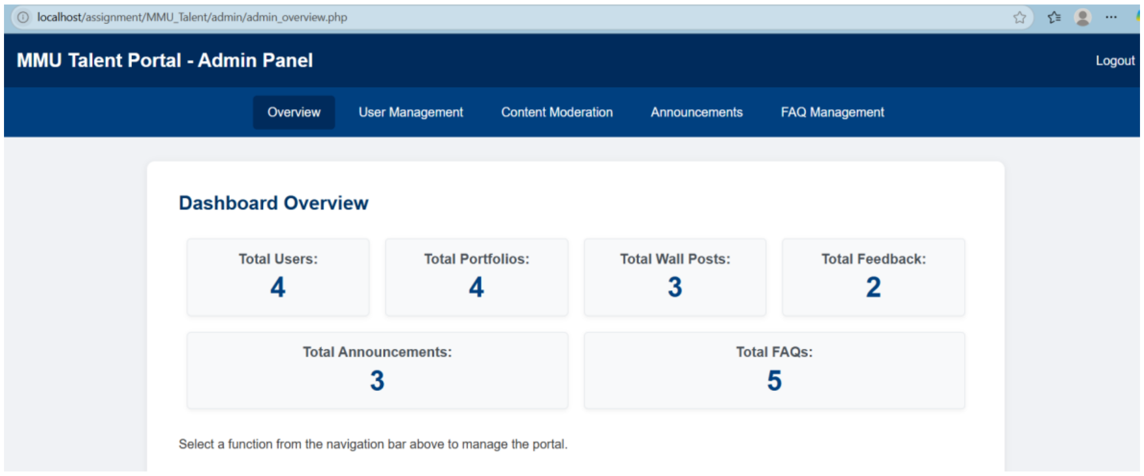
Recent Portfolio

User	Title	Description	Category	File	Created On	Actions
Anis (anis@gmail.com)	Logo Pandu Puteri (Girl Guides) Design	A digital rendering of the Pandu Puteri (Girl Guides) logo, showcasing vector illustration skills an...	Graphic Design	View File	2025-06-22 02:17:34	Delete
John Doe (john.doe@student.mmu.edu.my)	I Will Shine Song Performance	An emotional and powerful vocal performance of 'I Will Shine', demonstrating strong vocal control an...	Singing	View File	2025-06-22 02:02:03	Delete
Jane Smith (jane.smith@student.mmu.edu.my)	Guitar Performance - Acoustic Covers	A showcase of acoustic cover songs, demonstrating vocal range, control, and emotional delivery. Incl...	Singing	View File	2025-06-15 02:02:03	Delete
Anis (anis@gmail.com)	Sunset Photography Collection	A collection of stunning sunset photographs captured across various Malaysian landscapes. Focuses on...	Photography	View File	2025-06-12 02:02:03	Delete

Figure 5.5.8

Figure 5.5.8 depicts the administrative interface for 'Portfolio' content management. This view enables administrators to oversee all uploaded portfolios, including their titles, descriptions, categories, and associated files. The screenshot demonstrates the administrative tools for reviewing these submissions and, specifically, the option to delete portfolios that do not adhere to platform guidelines or policies, ensuring content quality and appropriateness.

5.5.6 admin dashboard


localhost/assignment/MMU_Talent/admin/admin_overview.php

MMU Talent Portal - Admin Panel

Logout

Overview | User Management | Content Moderation | Announcements | FAQ Management

Dashboard Overview

Total Users: 4

Total Portfolios: 4

Total Wall Posts: 3

Total Feedback: 2

Total Announcements: 3

Total FAQs: 5

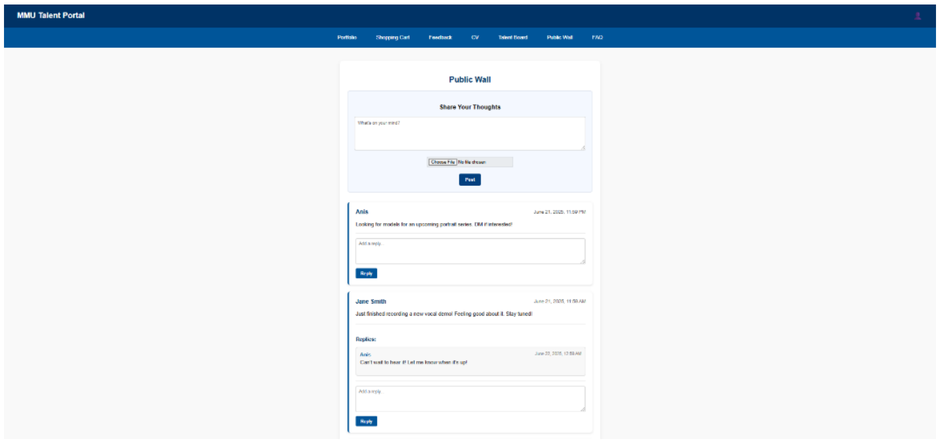
Select a function from the navigation bar above to manage the portal.

Figure 5.5.9

Figure 5.5.9 illustrates the administrative dashboard, providing an overview of recent wall posts, catalogues (portfolios), and user feedback. This interface enables administrators to efficiently monitor content, manage user activities, and perform moderation tasks, including direct links to delete inappropriate content.

5.5.7 public wall

User View :



Anis — June 21, 2025, 11:59 PM
Looking for models for an upcoming portrait series. DM if interested!

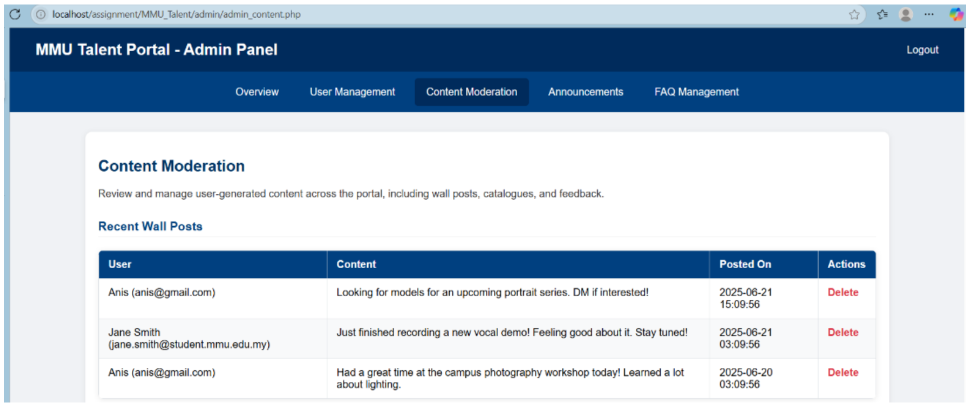
Jane Smith — June 21, 2025, 11:59 AM
Just finished recording a new vocal demo! Feeling good about it. Stay tuned!

Anis — June 22, 2025, 12:59 AM
Can't wait to hear it! Let me know when it's up!

Figure 5.5.10

Figure 5.5.10 depicts the 'Public Wall' interface, demonstrating user interaction capabilities. It shows how users can create new posts, including text messages and image uploads, and engage with existing content by adding replies. The display also highlights the integration of user profiles with posts and the option for users to delete their own content.

Admin View :


localhost/assignment/MMU_Talent/admin/admin_content.php
MMU Talent Portal - Admin Panel
Logout
Overview
User Management
Content Moderation
Announcements
FAQ Management

Content Moderation

Review and manage user-generated content across the portal, including wall posts, catalogues, and feedback.

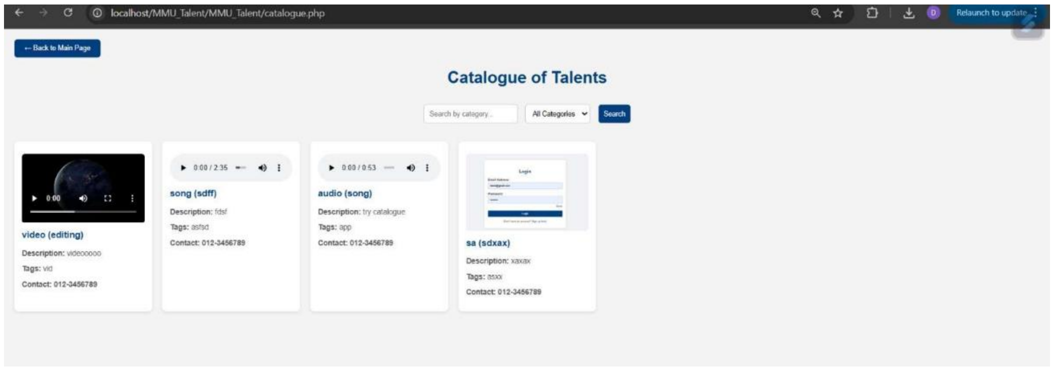
Recent Wall Posts

User	Content	Posted On	Actions
Anis (anis@gmail.com)	Looking for models for an upcoming portrait series. DM if interested!	2025-06-21 15:09:56	Delete
Jane Smith (jane.smith@student.mmu.edu.my)	Just finished recording a new vocal demo! Feeling good about it. Stay tuned!	2025-06-21 03:09:56	Delete
Anis (anis@gmail.com)	Had a great time at the campus photography workshop today! Learned a lot about lighting.	2025-06-20 03:09:56	Delete

Figure 5.5.11

Figure 5.5.11 displays the administrative view of the 'Public Wall,' highlighting content moderation capabilities. This interface allows administrators to review all user-generated posts and their associated replies. The screenshot specifically illustrates the ability to identify posts by user, view content, and directly initiate the deletion of any inappropriate or non-compliant wall posts from the system.

5.5.8 e-catalogue



Catalogue of Talents

video (editing)
Description: videooooo
Tags: vid
Contact: 012-3456789

song (sdff)
Description: fdsf
Tags: asfsd
Contact: 012-3456789

audio (song)
Description: try catalogue
Tags: app
Contact: 012-3456789

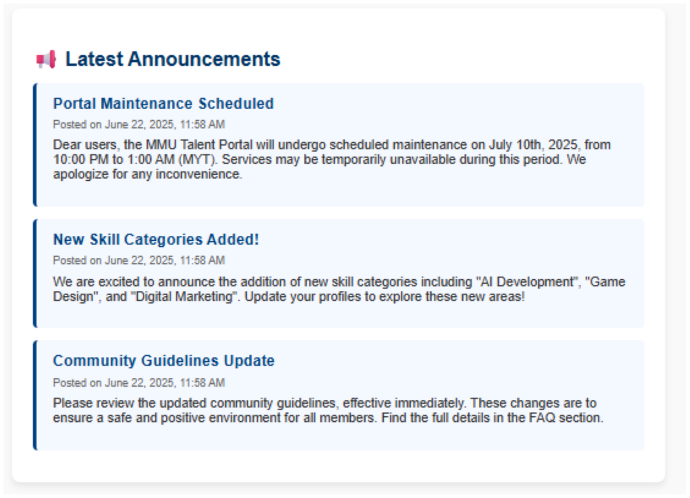
sa (sdxax)
Description: xaxax
Tags: asxx
Contact: 012-3456789

Figure 5.5.12

Figure 5.5.12 shows catalogue of talents page where users can see posted portfolio by other users. Each card in the catalogue contains the preview of uploaded portfolio file, title, description and the tags.

5.5.9 announcements

User View :



📢 Latest Announcements

Portal Maintenance Scheduled
Posted on June 22, 2025, 11:58 AM
Dear users, the MMU Talent Portal will undergo scheduled maintenance on July 10th, 2025, from 10:00 PM to 1:00 AM (MYT). Services may be temporarily unavailable during this period. We apologize for any inconvenience.

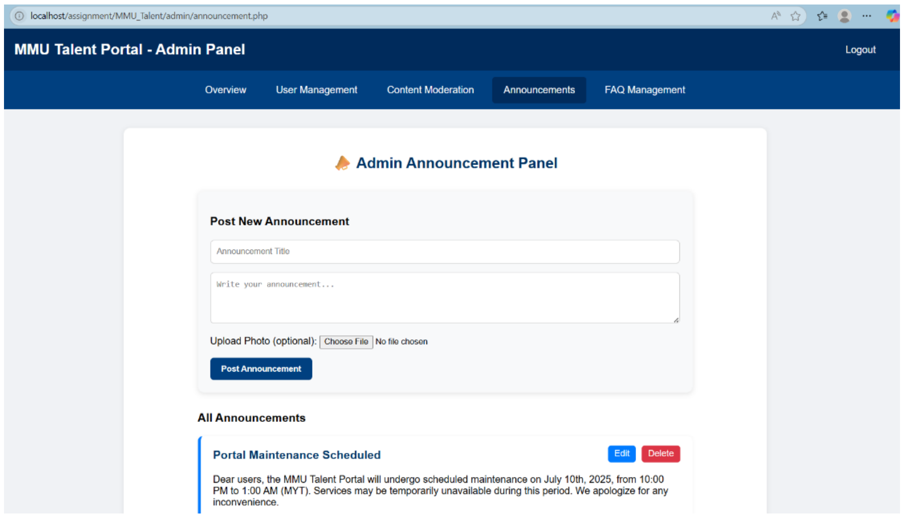
New Skill Categories Added!
Posted on June 22, 2025, 11:58 AM
We are excited to announce the addition of new skill categories including "AI Development", "Game Design", and "Digital Marketing". Update your profiles to explore these new areas!

Community Guidelines Update
Posted on June 22, 2025, 11:58 AM
Please review the updated community guidelines, effective immediately. These changes are to ensure a safe and positive environment for all members. Find the full details in the FAQ section.

Figure 5.5.13

Figure 5.5.13 shows an announcement section in the home page of the website. The admin posted announcement will appear here for all other users.

Admin View :

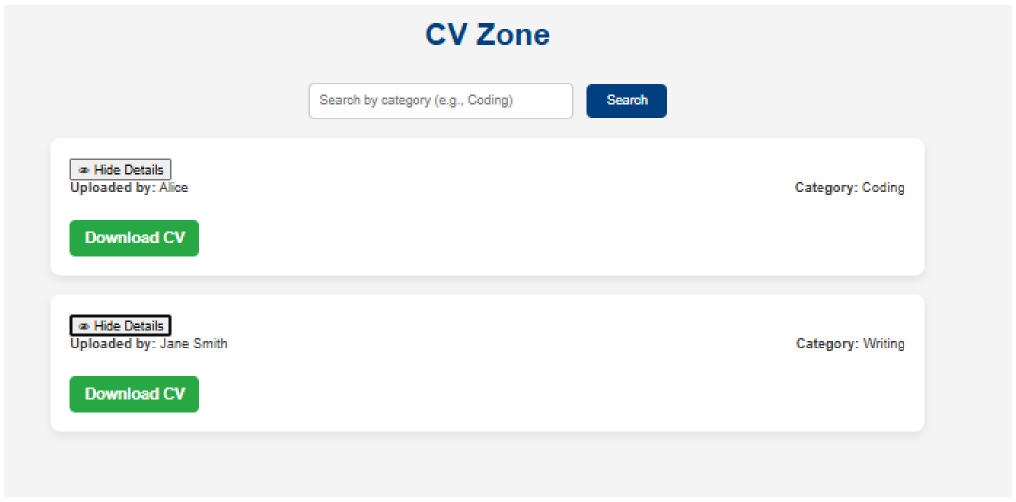

localhost/assignment/MMU_Talent/admin/announcement.php
MMU Talent Portal - Admin Panel
Logout
Overview
User Management
Content Moderation
Announcements
FAQ Management
Admin Announcement Panel
Post New Announcement
Announcement Title
Write your announcement...
Upload Photo (optional): Choose File No file chosen
Post Announcement
All Announcements
Portal Maintenance Scheduled
Edit
Delete

Dear users, the MMU Talent Portal will undergo scheduled maintenance on July 10th, 2025, from 10:00 PM to 1:00 AM (MYT). Services may be temporarily unavailable during this period. We apologize for any inconvenience.

Figure 5.5.14

Figure 5.5.14 depicts the 'Announcements Management' interface, a core administrative feature for broadcasting information to all portal users. This screenshot demonstrates how administrators can create new announcements, specify their content and visibility, and manage existing announcements (e.g., editing, deleting, or publishing/unpublishing). This ensures efficient communication of important updates, events, or policy changes across the platform.

5.5.10 cv zone



Uploaded by: Alice

Category: Coding

Uploaded by: Jane Smith

Category: Writing

Figure 5.5.15

Figure 5.5.15 shows the CV Zone page where users are able to see cvs uploaded by other users.

5.5.11upload cv

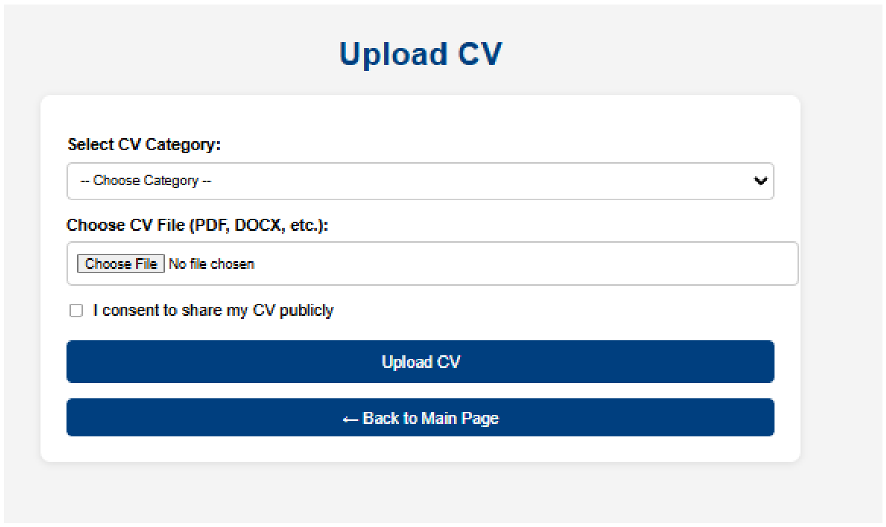
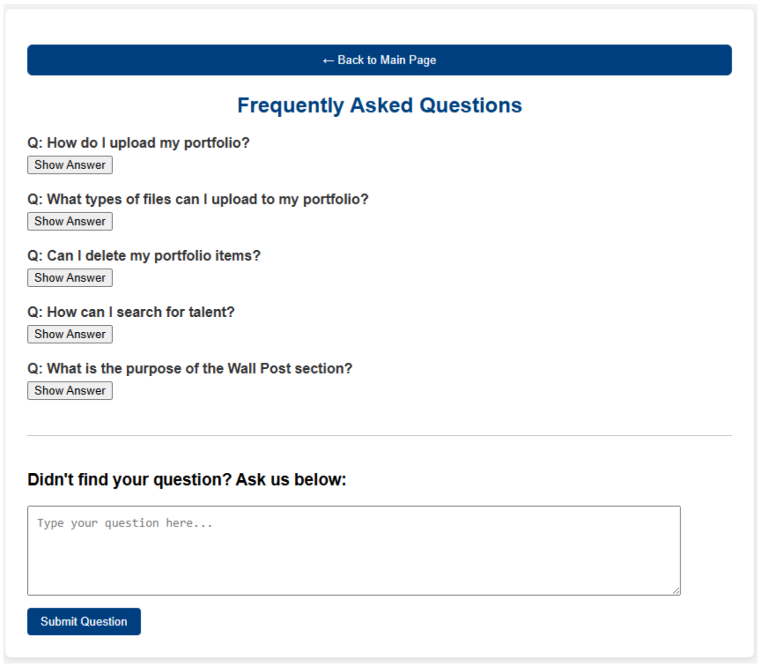


Figure 5.5.16

Figure 5.5.16 shows the upload CV page where users able to upload their CVs for public to see. Users need to select their talent category and choosing the CV file from their system.

5.5.12 faq

User View :


← Back to Main Page
Frequently Asked Questions
Show Answer
Didn't find your question? Ask us below:
Type your question here...
Submit Question

Q: How do I upload my portfolio?

Q: What types of files can I upload to my portfolio?

Q: Can I delete my portfolio items?

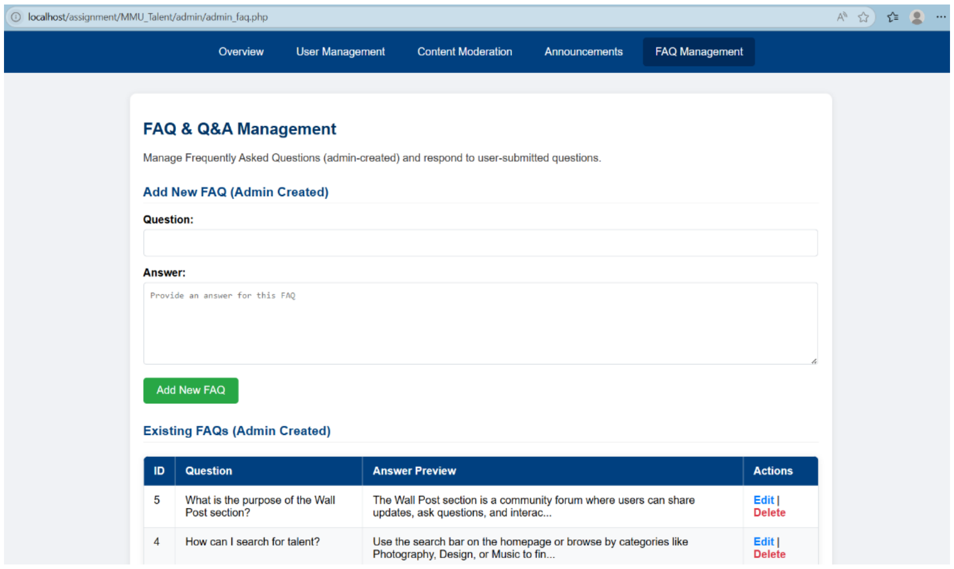
Q: How can I search for talent?

Q: What is the purpose of the Wall Post section?

Figure 5.5.17

Figure 5.5.17 shows FAQs that is answered by the admin on the FAQ page. If the user unable to find the answer to question they're looking for, users are also allowed to ask new question for the admin to answer.

Admin View :



localhost/assignment/MMU_Talent/admin/admin_faq.php

FAQ & Q&A Management

Manage Frequently Asked Questions (admin-created) and respond to user-submitted questions.

Add New FAQ (Admin Created)

Question:

Answer:

Provide an answer for this FAQ

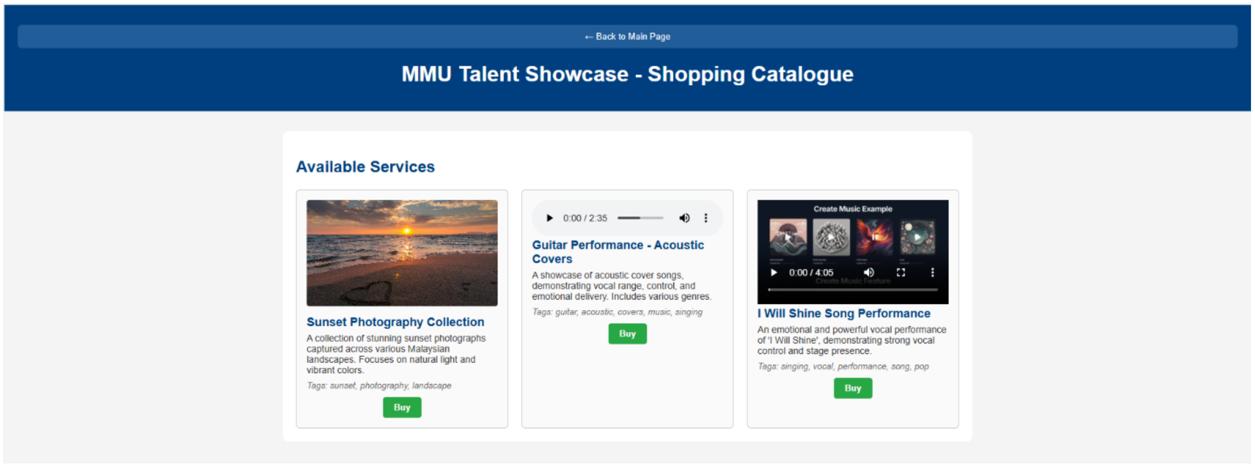
Existing FAQs (Admin Created)

ID	Question	Answer Preview	Actions
5	What is the purpose of the Wall Post section?	The Wall Post section is a community forum where users can share updates, ask questions, and interac...	Edit | Delete
4	How can I search for talent?	Use the search bar on the homepage or browse by categories like Photography, Design, or Music to fin...	Edit | Delete

Figure 5.5.18

Figure 5.5.18 showcases the administrative interface for 'Frequently Asked Questions (FAQ)' management. This section allows administrators to curate and maintain a knowledge base for users by adding new questions and answers, editing existing entries, and removing outdated information. The aim is to provide easily accessible solutions to common user queries, reducing the need for direct support.

5.5.13 cart



← Back to Main Page

MMU Talent Showcase - Shopping Catalogue

Available Services

Sunset Photography Collection

A collection of stunning sunset photographs captured across various Malaysian landscapes. Focuses on natural light and vibrant colors.

Tags: sunset, photography, landscape

Buy

0:00 / 2:35

Guitar Performance - Acoustic Covers

A showcase of acoustic cover songs, demonstrating vocal range, control, and emotional delivery. Includes various genres.

Tags: guitar, acoustic, covers, music, singing

Buy

Create Music Example

0:00 / 4:05

I Will Shine Song Performance

An emotional and powerful vocal performance of 'I Will Shine', demonstrating strong vocal control and stage presence.

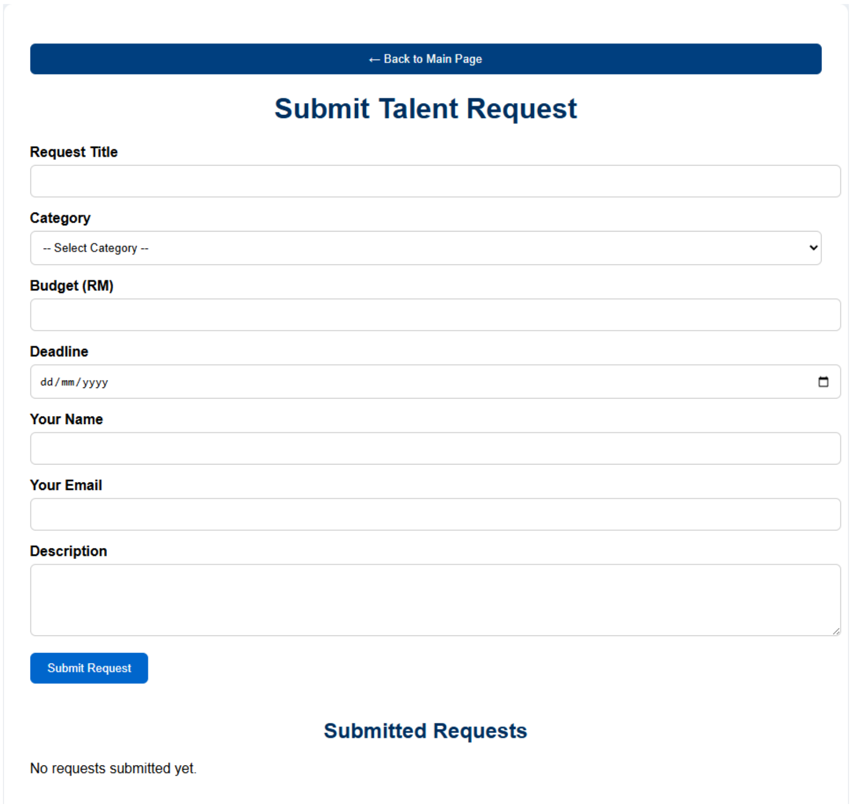
Tags: singing, vocal, performance, song, pop

Buy

Figure 5.5.19

Figure 5.5.19 shows shopping catalogue page where users can see available services by other students. If users are interested in any of the service available, they can buy and contact the one providing it.

5.5.14 request talent



← Back to Main Page

Submit Talent Request

Request Title

Category

-- Select Category --

Budget (RM)

Deadline

dd/mm/yyyy

Your Name

Your Email

Description

Submit Request

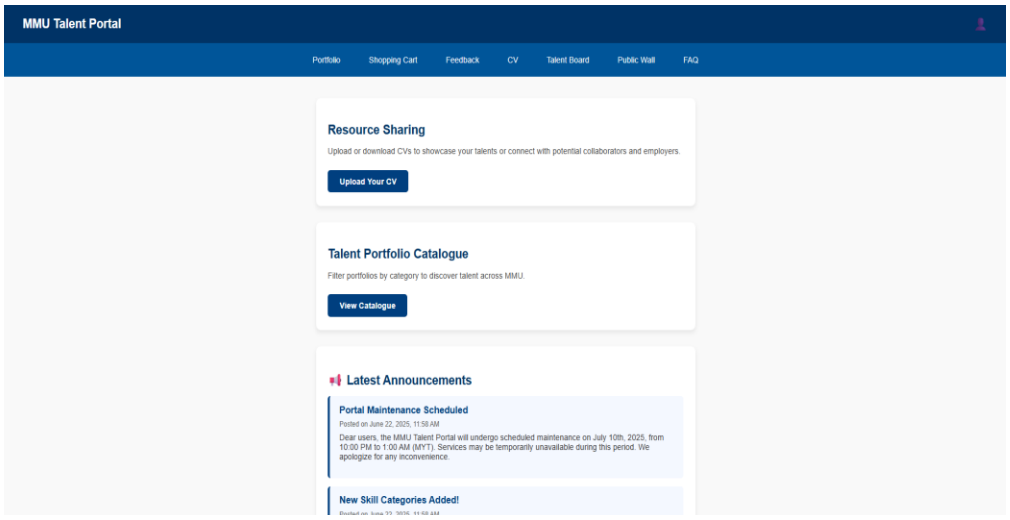
Submitted Requests

No requests submitted yet.

Figure 5.5.20

Figure 5.5.20 shows talent request page where users can request any specific service from other users. The user must fill in the title, detail description and category of the service, budget, deadline and details from the requestor before submitting their requests. On the same page, user can also see request submitted by other users.

5.5.15 index



Resource Sharing

Upload or download CVs to showcase your talents or connect with potential collaborators and employers.

Talent Portfolio Catalogue

Filter portfolios by category to discover talent across MMU.

Latest Announcements

Portal Maintenance Scheduled

Posted on June 22, 2025, 11:58 AM

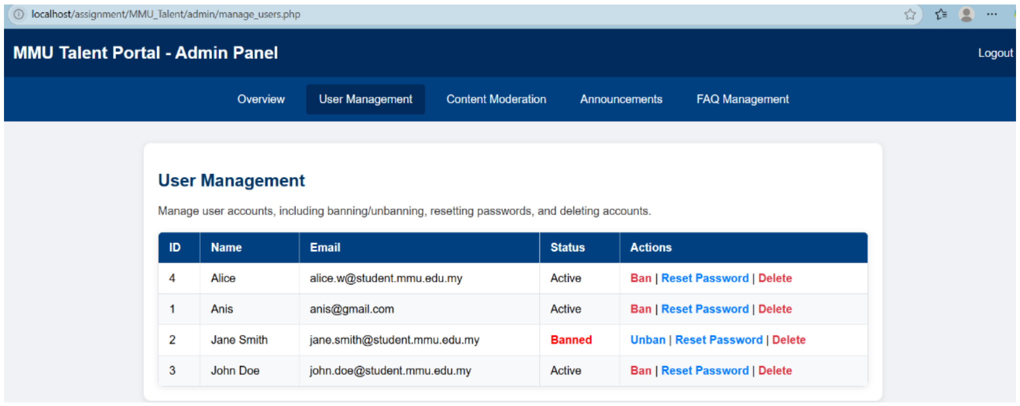
Dear users, the MMU Talent Portal will undergo scheduled maintenance on July 10th, 2025, from 10:00 PM to 1:00 AM (MYT). Services may be temporarily unavailable during this period. We apologize for any inconvenience.

New Skill Categories Added!

Figure 5.5.21

Figure 5.5.21 shows the home page for user. Users can navigate to other user functions via this page.

5.5.16 user management


localhost/assignment/MMU_Talent/admin/manage_users.php
MMU Talent Portal - Admin Panel
Logout
Overview
User Management
Content Moderation
Announcements
FAQ Management

User Management

Manage user accounts, including banning/unbanning, resetting passwords, and deleting accounts.

ID	Name	Email	Status	Actions
4	Alice	alice.w@student.mmu.edu.my	Active	Ban | Reset Password | Delete
1	Anis	anis@gmail.com	Active	Ban | Reset Password | Delete
2	Jane Smith	jane.smith@student.mmu.edu.my	Banned	Unban | Reset Password | Delete
3	John Doe	john.doe@student.mmu.edu.my	Active	Ban | Reset Password | Delete

Figure 5.5.22

Figure 5.5.22 presents the 'User Management' panel within the Admin Dashboard. This crucial administrative tool provides a consolidated view of registered users on the platform. It allows administrators to perform essential tasks such as approving new user registrations, banning problematic accounts, and initiating password resets, thereby maintaining the integrity and security of the user base.

6.0 Conclusion

6.1 Summary of Project Outcome

MMU Talent Showcase Portal was developed to improve the digital presence of the students, and this goal was achieved by provision of a thorough, secure, and user-friendly web application, which facilitates the displaying of talent. It allows creating personalized profiles, leaving portfolio edges, code, drawings, video artworks, and written pieces stored, accessed through peer-to-peer interactions via open-access feedback walls and an open talent request board. The platform also allows students to download the CVs or share them, fostering cooperation, as it allows students to freelance work.

Administratively, the system provides content moderation features, announcement posting and full monitoring of all activity thus making the system safe about quality and governance. The user-interface is built with the help of HTML, CSS, and JavaScript, the backend is created with the aid of PHP, and the storage is made with the use of MySQL(which saves the user profiles, portfolios, and communications). Both student and administrator authentication is done through roles with secure authentication, which guarantees stability and security of the platform.

To summarize, the portal covers both aspects of providing students with a convenient platform to exhibit their abilities and promoting their communication with each other but also ensures that the structure of the application offers the required level of control in the hands of the administrators to maintain the integrity of the resulting platform. Integration with strong authentication and file-management systems also ensures increased reliability and security by the system.

6.2 Challenges Faced and Lessons Learned

6.2.1 File Upload Validation

- Challenge: To handle various types and sizes of file uploads while ensuring file upload validations are appropriate and acceptable.
- Lesson Learned: Proper validation is essential to ensure good performance and prevent improper file formats from being uploaded. Limiting file types and sizes prevents system vulnerabilities and degraded performance.

6.2.2 Data Integrity

- Challenge: Maintaining the integrity of data by ensuring that user records are correctly linked to related data like portfolios, CVs, and files.
- Lesson Learned: The data model should be designed to preserve referential integrity, ensuring that accidental orphaning of records cannot occur. Cascade deletion or updates should be prevented.

6.2.3 Error Handling and Messaging

- Challenge: Properly handling errors and displaying descriptive error messages to users when their actions fail.
- Lesson Learned: Clear, concise error and success messages are crucial for improving user experience. This helps users understand the outcome of their actions and avoids system confusion.

6.2.4 Concurrency Management

- Challenge: Ensuring that multiple users can update files such as CVs, announcements, or portfolios, concurrently without data collisions.
- Lesson Learned: Proper orchestration of concurrency is critical to prevent data inconsistencies or race conditions. Ensuring synchronization of processes avoids conflicts and protects data integrity.

6.2.5 User Interface Consistency and Usability

- Challenge: Ensuring that the user interface is consistent and versatile across different types of media like text, images, videos, and other files.
- Lesson Learned: A uniform user interface improves usability and reduces errors. It enhances accessibility and makes the platform easier to use for a diverse range of users.

6.3 Suggested Future Enhancements

6.3.1 Profile Customization

Implement a profile customization feature where users can add additional fields or customize the layout of their profiles.

6.3.2 Advance Search and Filtering

Implement advanced search settings, such as multi-category filters, date ranges, and user-rated criteria, to narrow down the search for portfolios and CVs. This will improve the user experience by making searches more efficient and personalized.

6.3.3 File Preview Before Submission

Add file preview functionality for images, videos, and documents so users can preview their files before submission. This feature will help users ensure their files are correct and meet the system's requirements.

6.3.4 Public Wall

- Implement a "like" or "react" system for wall posts and replies.
- Add a reporting mechanism for inappropriate content directly on the wall for users.
- Implement real-time updates for new posts/replies without page refresh.

7.0 References

- **List of all referenced materials with APA formatting**

  - *PHP: Hypertext Preprocessor*. (n.d.). https://www.php.net/manual/en/features.file-upload.php
  - *W3Schools.com*. (n.d.). https://www.w3schools.com/php/php_file_upload.asp

- **Hyperlinked URLs immediately after each borrowed content**
  - MMU event picture
  - logo_pandu_puteri.jpg
  - john_sunset.jpg
  - jane_guitar.mp3
  - i_will_shine_song.mp4