🎓 BCA 4th Semester - PHP Programming

📚 UNIT – 4: File Handling and PHP with MySQL

🎯 Learning Objectives

By the end of this unit, students will be able to:

- ✅ Understand file operations in PHP
- ✅ Implement file upload functionality
- ✅ Connect PHP applications with MySQL database
- ✅ Execute database queries effectively
- ✅ Display dynamic data in HTML format

📁 File Operations in PHP

🔓 Opening Files

File opening is the first step in file manipulation. PHP provides the `fopen()` function to open files in different modes.

File Opening Modes:

Mode	Description	Purpose
`r`	Read only	Opens file for reading, pointer starts at beginning
`w`	Write only	Opens file for writing, truncates to zero length
`a`	Append only	Opens file for writing, pointer starts at end
`r+`	Read/Write	Opens file for reading and writing
`w+`	Read/Write	Creates new file or truncates existing file
`a+`	Read/Append	Opens file for reading and writing, pointer at end

Key Points:

- 🔹 Always check if file exists before opening
- 🔹 Use appropriate mode based on your requirements
- 🔹 Handle errors gracefully when file operations fail
- 🔹 Close files after operations to free system resources

📖 Reading Files

Reading files allows you to extract content from existing files and process the data.

Reading Methods:

- `fread()` - Reads specified number of bytes
- `fgets()` - Reads one line at a time
- `fgetc()` - Reads one character at a time
- `file_get_contents()` - Reads entire file into string
- `file()` - Reads entire file into array

Important Considerations:

- 🔸 Large files should be read in chunks to avoid memory issues
- 🔸 Always validate file content before processing
- 🔸 Use appropriate reading method based on file size and structure
- 🔸 Handle encoding issues for special characters

✏️ Writing Files

Writing files enables you to save data, create logs, and store application output.

Writing Methods:

- `fwrite()` - Writes string to file
- `fputs()` - Alias of fwrite()
- `file_put_contents()` - Writes data to file (simpler approach)

Best Practices:

- 🔹 Always validate data before writing
- 🔹 Use file locking for concurrent access
- 🔹 Implement backup mechanisms for critical data
- 🔹 Check available disk space before writing large files

📝 Appending to Files

Appending allows you to add new content to existing files without overwriting previous data.

Common Use Cases:

- 📋 Log files
- 📋 User activity tracking
- 📋 Data collection
- 📋 Error reporting

Advantages:

- 🌟 Preserves existing data
- 🌟 Efficient for continuous data addition
- 🌟 Ideal for logging applications
- 🌟 Reduces file management complexity

📤 File Upload Handling

🎯 Understanding File Uploads

File upload functionality allows users to submit files through web forms to the server.

Upload Process Flow:

1. 🔄 User selects file through HTML form
2. 🔄 Browser sends file data to server
3. 🔄 PHP receives and validates file
4. 🔄 File is moved to designated directory
5. 🔄 Success/error response sent to user

🛡️ Security Considerations

File uploads pose significant security risks that must be addressed:

Security Measures:

- 🔒 Validate file types and extensions
- 🔒 Limit file sizes
- 🔒 Scan for malicious content
- 🔒 Use secure upload directories
- 🔒 Rename uploaded files
- 🔒 Implement user authentication

⚙️ Configuration Requirements

PHP settings that affect file uploads:

Setting	Description	Recommended Value
`file_uploads`	Enable/disable file uploads	On
`upload_max_filesize`	Maximum file size	10M
`post_max_size`	Maximum POST data size	12M
`max_file_uploads`	Maximum number of files	20

🗄️ Connecting PHP with MySQL Database

🔗 Database Connection Methods

PHP offers multiple ways to connect with MySQL databases:

Connection Options:

- **MySQLi Extension** - Improved MySQL extension
- **PDO (PHP Data Objects)** - Database abstraction layer
- **MySQL Extension** - Legacy (deprecated)

🌟 MySQLi vs PDO Comparison

Feature	MySQLi	PDO
Database Support	MySQL only	Multiple databases
Object-Oriented	Yes	Yes
Prepared Statements	Yes	Yes
Performance	Slightly faster	Good performance
Learning Curve	Easier	Moderate

🔧 Connection Best Practices

- 🎯 Use environment variables for credentials
- 🎯 Implement connection pooling
- 🎯 Handle connection errors gracefully
- 🎯 Use SSL for secure connections
- 🎯 Close connections when not needed

🔍 Executing Database Queries

➕ INSERT Operations

INSERT queries add new records to database tables.

Key Concepts:

- 📌 Data validation before insertion
- 📌 Handling auto-increment fields
- 📌 Managing foreign key constraints
- 📌 Bulk insert operations for efficiency

Important Considerations:

- 🔸 Validate all input data
- 🔸 Use prepared statements to prevent SQL injection
- 🔸 Handle duplicate key errors
- 🔸 Implement transaction support for complex operations

🔍 SELECT Operations

SELECT queries retrieve data from database tables.

Query Types:

- **Simple SELECT** - Basic data retrieval
- **Conditional SELECT** - With WHERE clause
- **JOIN Operations** - Combining multiple tables
- **Aggregate Functions** - COUNT, SUM, AVG, etc.

Optimization Tips:

- 🚀 Use appropriate indexes
- 🚀 Limit result sets with LIMIT clause

- 🚀 Avoid SELECT * for large tables
- 🚀 Use efficient WHERE conditions

🔄 UPDATE Operations

UPDATE queries modify existing records in database tables.

Update Strategies:

- **Single Record Update** - Modify one specific record
- **Bulk Update** - Modify multiple records
- **Conditional Update** - Update based on conditions
- **Partial Update** - Update only specific fields

Safety Measures:

- ⚠️ Always use WHERE clause to avoid updating all records
- ⚠️ Backup data before major updates
- ⚠️ Test updates on development environment first
- ⚠️ Use transactions for complex update operations

🗑️ DELETE Operations

DELETE queries remove records from database tables.

Deletion Types:

- **Soft Delete** - Mark records as deleted
- **Hard Delete** - Permanently remove records
- **Cascading Delete** - Delete related records

- **Conditional Delete** - Delete based on criteria

Precautions:

- 🛑 Always backup before deletion
- 🛑 Use WHERE clause to avoid deleting all records
- 🛑 Consider referential integrity
- 🛑 Implement confirmation mechanisms

🎨 Fetching Data and Displaying in HTML

📊 Data Retrieval Methods

Different approaches to fetch data from database results:

Fetch Methods:

- `fetch_assoc()` - Returns associative array
- `fetch_array()` - Returns both numeric and associative array
- `fetch_row()` - Returns numeric array
- `fetch_object()` - Returns object

🎭 HTML Display Techniques

Presenting database data in user-friendly HTML format:

Display Options:

- 📋 **Tables** - Structured data presentation
- 📋 **Lists** - Simple data enumeration

- 📋 **Cards** - Modern UI components
- 📋 **Forms** - Interactive data editing

🎨 Styling and Formatting

Enhancing data presentation with CSS and JavaScript:

Enhancement Techniques:

- 🌈 CSS styling for better appearance
- 🌈 JavaScript for interactive features
- 🌈 Responsive design for mobile compatibility
- 🌈 Data pagination for large datasets

🔄 Dynamic Content Generation

Creating dynamic web pages that respond to user input:

Dynamic Features:

- 🎯 Search functionality
- 🎯 Sorting capabilities
- 🎯 Filtering options
- 🎯 Real-time updates

💡 Best Practices and Tips

🛡️ Security Guidelines

- Always validate and sanitize user input

- Use prepared statements for database queries
- Implement proper error handling
- Regular security audits and updates

⚡ Performance Optimization

- Optimize database queries
- Use appropriate data types
- Implement caching mechanisms
- Monitor application performance

📝 Code Organization

- Follow consistent coding standards
- Use meaningful variable names
- Implement proper documentation
- Separate business logic from presentation

🔧 Error Handling

- Implement comprehensive error handling
- Log errors for debugging
- Provide user-friendly error messages
- Test error scenarios thoroughly

🎯 Summary

This unit covers essential concepts for file handling and database operations in PHP:

✨ **Key Takeaways:**

- File operations enable data persistence and manipulation
- Proper file upload handling ensures application security
- Database connectivity provides dynamic data management
- Effective query execution enables robust applications
- HTML integration creates user-friendly interfaces

✨ **Skills Developed:**

- File manipulation and management
- Secure file upload implementation
- Database connection and query execution
- Dynamic web page creation
- Security-aware programming practices

📚 Additional Resources

For further learning and reference:

- 📖 PHP Official Documentation
- 📖 MySQL Reference Manual
- 📖 Web Security Best Practices
- 📖 Database Design Principles
- 📖 Modern PHP Development Practices

💻 *Happy Coding! Keep practicing and building amazing PHP applications!* 🚀