PHP – 100+ Lab Exercises (Basic, Intermediate, Advanced)

☐ Basic Level (30+ Exercises)

Objective: Build a solid foundation in PHP syntax, data types, control structures, and basic web integration.

☐ PHP Fundamentals

- Install PHP and set up a development environment (XAMPP, MAMP, LAMP).
- Write and execute your first PHP script.
- Variables, constants, and data types (strings, integers, floats, arrays).
- Use operators (arithmetic, comparison, logical).
- Control structures: `if`, `else`, `switch`.
- Looping structures: `for`, `while`, `foreach`.
- Functions and built-in PHP functions.
- Handling form data via `GET` and `POST` methods.
- Basic string and array manipulations.
- Error handling with `try-catch` and `error_reporting`.

☐ Practice Exercises

- Create a simple calculator.
- Validate form inputs (email, phone number).
- Write a script to manipulate arrays (sorting, searching).
- Build a simple contact form that emails data.
- Implement a session-based visitor counter.

☐ Intermediate Level (40+ Exercises)

Objective: Develop skills in working with databases, object-oriented PHP, and building dynamic web applications.

☐ Database Integration & OOP

- Connect to MySQL/MariaDB using MySQLi and PDO.
- Perform CRUD operations on a database.
- Create and use classes and objects (OOP fundamentals).
- Implement inheritance, interfaces, and traits.
- Use namespaces and autoloading (PSR-4).
- Manage sessions and cookies securely.
- Handle file uploads and processing.
- Work with PHP date/time functions.
- Send emails using PHPMailer.
- Introduction to PHP frameworks (Laravel, Symfony).

☐ Practice Exercises

- Build a user registration and login system.
- Create a CRUD application for managing products.
- Develop a blog with posts, comments, and categories.
- Implement file upload and image resizing.
- Design a session-based shopping cart.

☐ Advanced Level (40+ Exercises)

Objective: Master advanced PHP concepts, REST APIs, security best practices, and scalable architecture.

☐ Advanced Concepts & API Development

- Create RESTful APIs with PHP (using Slim or Laravel Lumen).
- Implement JWT authentication.
- Work with Composer and manage dependencies.
- Use design patterns (Singleton, Factory, MVC).
- Implement caching strategies (Redis, Memcached).
- Optimize performance (opcode caching, profiling).
- Apply security best practices (SQL injection prevention, XSS, CSRF).
- Use PHPUnit for automated testing.
- Integrate third-party APIs and payment gateways.
- Deploy PHP applications with Docker and CI/CD pipelines.

⬜ Practice Exercises

- Build a fully functional REST API for a task management system.
- Secure an API with JWT and OAuth2.
- Implement role-based access control (RBAC).
- Create unit and integration tests for your PHP code.
- Deploy a PHP application using Docker and GitHub Actions.

⬜ Capstone Projects

- Develop a full-featured e-commerce website with payment integration.
- Build a social media platform with real-time notifications.
- Create a content management system (CMS) with user roles and permissions.
- Design a REST API for a mobile application backend.
- Implement a microservices architecture using PHP and Docker.

⬜ Tools & Technologies

- PHP 7.x/8.x
- MySQL/MariaDB
- Composer (Dependency Manager)
- PHPUnit (Testing Framework)
- Redis/Memcached (Caching)
- Docker (Containerization)
- Xdebug (Debugging)
- Popular frameworks: Laravel, Symfony, CodeIgniter