# Project Documentation

## Directory Structure

The project is structured as follows:

.

├── add\_database.php # Page to add a new database

├── auto\_backup.php # Script to handle automatic backups

├── backups # Directory to store backup files

├── backup\_scheduler.php # Scheduler for managing backup intervals

├── Chart.js # JavaScript file for rendering charts

├── check\_and\_run\_backup.php # Script to check and run backup jobs

├── cron

│ └── crontabe # Cron job configuration for automated backups

├── db.php # Database connection and handling

├── delete\_backup.php # Script to delete a specific backup

├── delete\_database.php # Script to delete a specific database

├── delete\_schedule.php # Script to delete backup schedule

├── download\_backup.php # Script to download backup files

├── download.php # General download handler

├── footer.php # Footer section for HTML pages

├── header.php # Header section for HTML pages

├── images

│ └── logo.png # Logo image for the project

├── index.php # Main entry point (landing page)

├── list\_backups.php # Page to list all available backups

├── login.php # Login page for user authentication

├── logout.php # Logout functionality

├── manual\_backup.php # Page to trigger manual backups

├── modify\_database.php # Page to modify database details

├── README.md # Project README file (this file)

├── REDME.dotx # Alternate documentation file in .dotx format

├── restore\_backup.php # Page to restore a backup

├── restore\_backup.sh # Script to restore a backup from file

├── restore\_database.php # Script to restore a database

├── schedule.php # Page to manage backup schedules

├── scripts.js # JavaScript for various frontend interactions

├── select\_database.php # Page to select a database for backup/restore

├── sql

│ ├── backup\_app.sql # SQL dump for the backup application

├── styles

│ └── style.css # Stylesheet for the frontend pages

├── update\_database.php # Page to update database settings

├── update\_schedule.php # Page to update backup schedule settings

├── upload\_script.sh # Script to upload backup files

└── users.php # User management page (add, modify, delete users)

## Project Overview

This project provides a web interface for managing MySQL database backups. It includes functionality for scheduling backups, performing manual backups, restoring backups, and managing users. The application also supports storing backups and managing backup history.

### Features

* **Database Management**: Add, modify, and delete databases.
* **Backup Management**: Schedule, perform manual backups, and restore backups.
* **Backup Scheduler**: Set intervals for automatic backups.
* **Backup History**: View and download previous backups.
* **User Management**: Add, modify, and delete users who can access the system.

## Requirements

To run this project locally, you will need:

* PHP 7.4 or higher
* MySQL server
* Apache web server
* Docker (optional for containerization)

## Installation

1. **Clone the repository**:

git clone <repository\_url>

cd project\_directory

1. **Set up the MySQL database**: Import the database schema and application data:

mysql -u root -p < sql/backup\_app.sql

1. **Configure the project**:
   * **Database Connection**: Database configuration is handled in the db.php file.
   * The default credentials for logging into the application are:
     + **Username**: admin
     + **Password**: admin
2. **Start the web application**: Make sure Apache and PHP are set up to serve the application, or use Docker to containerize the app.

## Usage

* **Accessing the Web Interface**: After setting up, visit http://localhost/index.php to access the main page of the application.
* **Backup Operations**: Use the backup\_scheduler.php to set up backup schedules, or perform manual backups via manual\_backup.php.
* **User Management**: Admin users can manage system users via users.php.
* **Backup Restoration**: You can restore backups via restore\_backup.php.
* **Cron**: You can use cron example /crone to add into crontab

## Docker Setup (Optional)

If you prefer to containerize the application, you can use Docker. Please refer to the Dockerfile and docker-compose.yml for setting up the environment.

## Additional Details

### Database Schema

The project uses a MySQL database for storing backup and schedule information. The database is initialized using sql/backup\_app.sql and includes tables like databases, backups, and schedules.

### Scripts Overview

* **Backup Handling**:
  + auto\_backup.php: Handles automatic backups based on a schedule.
  + manual\_backup.php: Allows manual triggering of backups.
  + check\_and\_run\_backup.php: Checks for pending backups and triggers them.
* **Backup Restoration**:
  + restore\_backup.php: Interface for restoring a selected backup.
  + restore\_backup.sh: Shell script to restore a backup from a file.
* **Scheduling**:
  + backup\_scheduler.php: Set up backup schedules for periodic backups.
  + update\_schedule.php: Update backup schedules.
* **Database Management**:
  + add\_database.php: Add new databases to be backed up.
  + modify\_database.php: Modify database connection details.
  + delete\_database.php: Remove a database from the backup system.