



Modern Authentication System

A fully functional authentication system built for the GuvihCL internship assignment, featuring modern UI design with glassmorphism effects and secure backend implementation.

✨ Features

- **Secure Authentication:** User registration and login with password hashing
- **Profile Management:** Edit personal information (age, DOB, contact)
- **Session Management:** Redis-based backend sessions with localStorage frontend
- **Activity Logging:** MongoDB logging for user actions
- **Modern UI:** Glassmorphism design with smooth animations
- **Responsive:** Mobile-friendly Bootstrap layout
- **AJAX-Only:** No form submissions, all jQuery AJAX requests
- **Security:** Prepared statements, password hashing, session validation

🔧 Tech Stack

- **Frontend:** HTML5, CSS3, JavaScript (jQuery), Bootstrap 5
- **Backend:** PHP 7.4+
- **Databases:**
 - MySQL (user authentication data)
 - MongoDB (activity logs)
 - Redis (session storage)

📋 Prerequisites

Before you begin, ensure you have the following installed:

1. **XAMPP** (or any Apache + PHP + MySQL stack)
 - Download: <https://www.apachefriends.org/>
 - PHP 7.4 or higher required
2. **Redis Server**
 - Windows: <https://github.com/microsoftarchive/redis/releases>
 - Mac: brew install redis
 - Linux: sudo apt-get install redis-server
3. **MongoDB**
 - Download: <https://www.mongodb.com/try/download/community>
 - Follow installation instructions for your OS
4. **Composer** (PHP dependency manager)
 - Download: <https://getcomposer.org/download/>

🔧 Installation Steps

Step 1: Download and Extract

1. Extract this project to your XAMPP htdocs directory
 - Path: C:\xampp\htdocs\internship-app (Windows)
 - Path: /Applications/XAMPP/htdocs/internship-app (Mac)

Step 2: Database Setup

MySQL Database

1. Open phpMyAdmin: <http://localhost/phpmyadmin>
2. Create a new database named `internship_app`
3. Run the following SQL:

```
CREATE DATABASE IF NOT EXISTS internship_app  
CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci;
```

```
USE internship_app;
```

```
CREATE TABLE users (  
  id INT AUTO_INCREMENT PRIMARY KEY,  
  name VARCHAR(100) NOT NULL,  
  email VARCHAR(150) NOT NULL UNIQUE,  
  password_hash VARCHAR(255) NOT NULL,  
  age INT NULL,  
  dob DATE NULL,  
  contact VARCHAR(20) NULL,  
  created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,  
  INDEX idx_email (email)  
) ENGINE=InnoDB;
```

MongoDB

MongoDB will automatically create the database and collection on first use. No manual setup required.

Redis

Redis requires no setup, just ensure it's running:

```
# Windows: Run redis-server.exe  
# Mac/Linux:  
redis-server
```

To verify Redis is running:

```
redis-cli ping  
# Should return: PONG
```

Step 3: Install PHP Dependencies

Navigate to the assets/php directory and install MongoDB driver:

```
cd C:\xampp\htdocs\internship-app\assets\php
composer require mongodb/mongodb
```

This creates a vendor folder with required libraries.

Step 4: Enable PHP Extensions

Edit your php.ini file (usually in C:\xampp\php\php.ini):

1. Uncomment (remove ; from) these lines:

```
extension=mysqli
extension=redis
extension=mongodb
```

2. If mongodb extension is not available, install it:

```
# Windows (XAMPP)
# Download from: https://pecl.php.net/package/mongodb
# Place .dll in C:\xampp\php\ext\
```

```
# Mac/Linux
sudo pecl install mongodb
```

3. Restart Apache from XAMPP Control Panel



Step 5: Verify Installation

Create a test file test.php in your project root:


```
<?php
// Test MySQL
$mysqli = new mysqli("localhost", "root", "", "internship_app");
echo "MySQL: " . ($mysqli->connect_errno ? "❌ Failed" : "✅ Connected") . "<br>";

// Test Redis
try {
    $redis = new Redis();
    $redis->connect('127.0.0.1', 6379);
    echo "Redis: ✅ Connected<br>";
} catch (Exception $e) {
    echo "Redis: ❌ Failed<br>";
}

// Test MongoDB
require_once 'assets/php/vendor/autoload.php';
```

```
try {
    $mongo = new MongoDB\Client("mongodb://127.0.0.1:27017");
    echo "MongoDB:  Connected<br>";
} catch (Exception $e) {
    echo "MongoDB:  Failed<br>";
}
?>
```

Visit <http://localhost/internship-app/test.php>

You should see three  marks. Delete the file after verification.

Running the Application

1. Start All Services:

- XAMPP: Start Apache and MySQL
- Redis: Run redis-server
- MongoDB: Run mongod or ensure MongoDB service is running

2. Access the Application:

- Open browser and go to: <http://localhost/internship-app/>

3. Test the Flow:

1. Click "Get Started" or go to Register page
2. Fill in registration form and submit
3. Login with your credentials
4. View and edit your profile
5. Logout to end session

Project Structure

```
internship-app/
├── assets/
│   ├── css/
│   │   └── style.css      # Modern glassmorphism styles
│   ├── js/
│   │   ├── register.js   # Registration logic
│   │   ├── login.js      # Login logic
│   │   └── profile.js     # Profile management
│   └── php/
│       ├── vendor/       # Composer dependencies
│       ├── config.php     # Database connections
│       ├── register.php   # Registration API
│       ├── login.php      # Login API
│       └── profile.php    # Profile API
├── index.html             # Landing page
├── register.html          # Registration page
├── login.html             # Login page
└── profile.html           # Profile page
```

Security Features

- **Password Hashing:** BCrypt with cost factor 12
- **Prepared Statements:** All SQL queries use prepared statements
- **Session Management:** Redis-backed sessions with 1-hour expiry
- **Input Validation:** Both client-side and server-side
- **XSS Protection:** All outputs are escaped
- **CSRF:** Session tokens in localStorage

Data Flow

1. **Registration:**
 - User submits form → jQuery AJAX → register.php
 - Validation → Hash password → Insert to MySQL
 - Log event to MongoDB → Return success
2. **Login:**
 - User submits form → jQuery AJAX → login.php
 - Verify credentials → Generate session token
 - Store in Redis → Return token to client
 - Client stores token in localStorage
3. **Profile Access:**
 - Page loads → Check localStorage for token
 - Send token via AJAX → profile.php validates with Redis
 - Fetch user data from MySQL → Return to client
4. **Profile Update:**
 - User submits changes → AJAX with token
 - Validate session → Update MySQL
 - Log event to MongoDB → Return success
5. **Logout:**
 - User clicks logout → AJAX with token
 - Delete Redis session → Clear localStorage
 - Redirect to login

Troubleshooting

Redis Connection Failed

Check if Redis is running
redis-cli ping

Start Redis if not running
redis-server

MongoDB Connection Failed

```
# Check if MongoDB is running
mongosh
```

```
# Start MongoDB service (Linux)
sudo systemctl start mongod
```

```
# Start MongoDB (Mac)
brew services start mongodb-community
```

MySQL Connection Failed

- Ensure XAMPP MySQL is running
- Check username/password in config.php (default: root with no password)
- Verify database internship_app exists

Composer Not Found

```
# Install globally
php -r "copy('https://getcomposer.org/installer', 'composer-setup.php');"
php composer-setup.php --install-dir=/usr/local/bin --filename=composer
php -r "unlink('composer-setup.php');"
```

PHP Extensions Not Loading

1. Locate php.ini: `php --ini`
2. Edit and enable extensions
3. Restart Apache
4. Verify: `php -m` (should list mysqli, redis, mongodb)

Assignment Compliance

This project strictly follows all GuviHCL internship requirements:

- ✓ HTML, CSS, JS, PHP in separate files
- ✓ jQuery AJAX only (no form submissions)
- ✓ Bootstrap for responsive design
- ✓ MySQL with prepared statements
- ✓ MongoDB for data storage
- ✓ Redis for session management
- ✓ localStorage for client sessions
- ✓ No PHP `$_SESSION` usage
- ✓ Complete Register → Login → Profile flow

UI Design

The interface features:

- **Glassmorphism:** Frosted glass effect with backdrop blur
- **Gradient Backgrounds:** Purple-to-blue animated gradients
- **Smooth Animations:** CSS transitions and keyframe animations
- **Responsive Layout:** Mobile-first Bootstrap grid
- **Loading States:** Button animations during AJAX calls
- **Form Validation:** Real-time feedback with color indicators

License

This project is created for educational purposes as part of the GuviHCL internship assignment.

Author

Created by [Your Name] for GuviHCL Developer Internship

Need Help?

If you encounter any issues:

1. Check all services are running (Apache, MySQL, Redis, MongoDB)
2. Verify PHP extensions are enabled
3. Check browser console for JavaScript errors
4. Check PHP error logs in XAMPP
5. Ensure database credentials are correct in config.php

For additional support, refer to the official documentation:

- [PHP Manual](#)
- [jQuery Documentation](#)
- [Bootstrap Documentation](#)
- [MongoDB PHP Library](#)
- [Redis PHP Extension](#)