Internship Project Report

Project Title: Phishing Awareness Simulation

Submitted by: Anand Kalirana

Table of Contents

Introduction

The threat of phishing has become increasingly prevalent in today's digital landscape. To counteract this, organizations are actively training employees to recognize and avoid phishing attempts. The goal of this internship project was to develop a Phishing Awareness Simulation Platform:-a safe, educational tool that mimics phishing scenarios and evaluates users' susceptibility to such attacks.

This platform aims to simulate a real-world phishing campaign, capture user actions (like entering credentials), and provide feedback or redirection for awareness. It is intended for internal use in organizations and educational settings.

Project Objectives

- Create a Laravel-based web platform to simulate phishing attempts.
- Allow administrators to create and manage phishing campaigns.
- Design fake phishing pages that capture login credentials for training purposes.
- Log user interactions (email opened, link clicked, credentials entered).
- Redirect users to the actual legitimate site after simulation.
- Raise cybersecurity awareness among users in a non-harmful environment.

Scope of the Project

This project simulates phishing scenarios without causing harm or using real credentials maliciously. It is strictly for internal awareness campaigns. The scope covers:

- Web-based simulation only (not SMS or voice phishing).
- Email and landing page simulations.
- Admin-side management with full CRUD operations.

Tools and Technologies Used

Tool/Technology	Purpose
Laravel (PHP Framework)	**Backend and Routing**
MySQL	**Database**
Mailtrap	**Email sending simulation**
Blade Templating Engine	**Frontend design**
HTML, CSS, Bootstrap	**UI/UX**
XAMPP / Laravel Homestead	**Local Development Server**

Project Architecture

I. Admin creates a phishing campaign (email content + fake login page).
II. Email sent to target users via Mailtrap.
III. User clicks the email link and lands on a fake login page.
IV. User enters credentials → credentials are logged.
V. User is redirected to the real website.
VI. Admin views captured credentials and interaction logs.

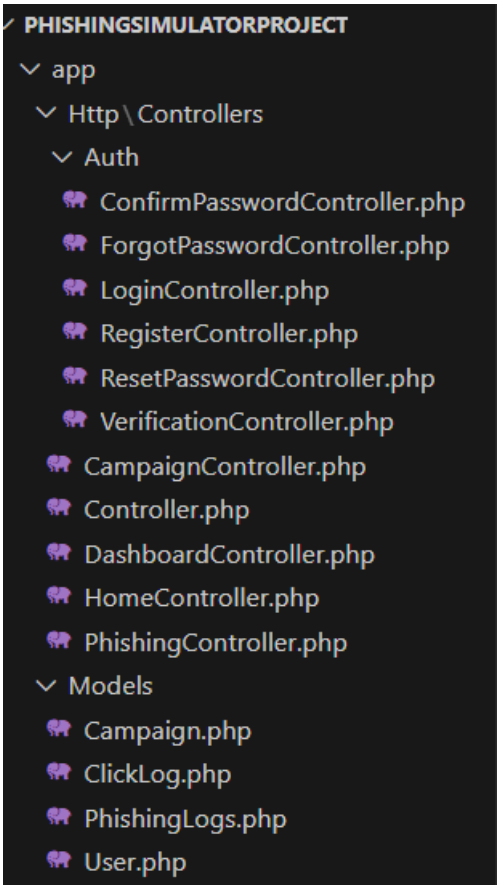
Detailed Implementation

I. Task 1: Develop the Phishing Awareness Simulation Platform

The goal of this task was to build a functioning phishing simulation platform using Laravel and set up the email sending mechanism through Mailtrap.

Steps Implemented:

- Initialized Laravel project using composer create-project.
- Created routes, controllers, and views for the landing and phishing pages.
- Designed a fake login form to simulate phishing.
- Set up Mailtrap in the .env file to simulate email sending in a sandbox environment.
- Validated user inputs and ensured proper form handling using Laravel's request classes.



Laravel Project Controllers and Models Structure

This screenshot displays the folder hierarchy within the Laravel project, specifically under the app/Http/Controllers and app/Models directories.

Highlights:

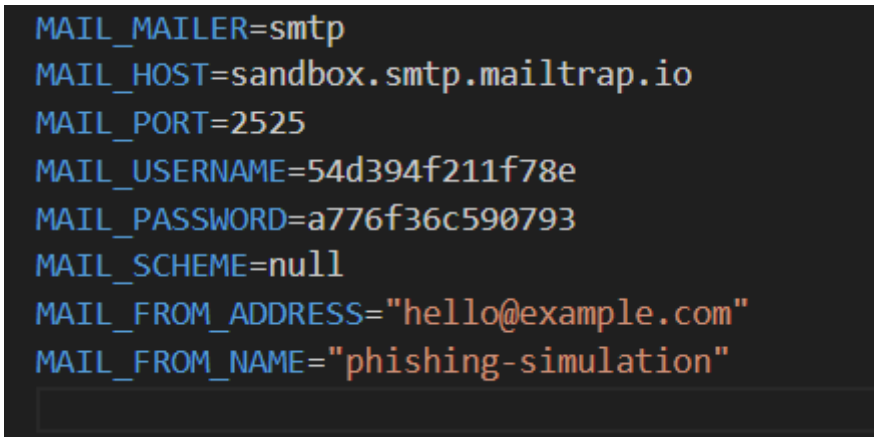
1. Controllers:

- PhishingController.php: Handles simulated phishing page logic.
- CampaignController.php: Manages campaign CRUD operations.
- DashboardController.php: Provides backend interface for administrators.
- Auth subfolder: Contains default Laravel authentication controllers for user management (login, register, reset, etc.)

2. Models:

- Campaign.php: Represents phishing campaign data structure.
- ClickLog.php: Stores logs when a user clicks a phishing link.
- PhishingLogs.php: Captures user credentials and logs during the simulation.
- User.php: Default user model extended for admin authentication.

This well-organized structure ensures clear separation of concerns following MVC principles in Laravel.



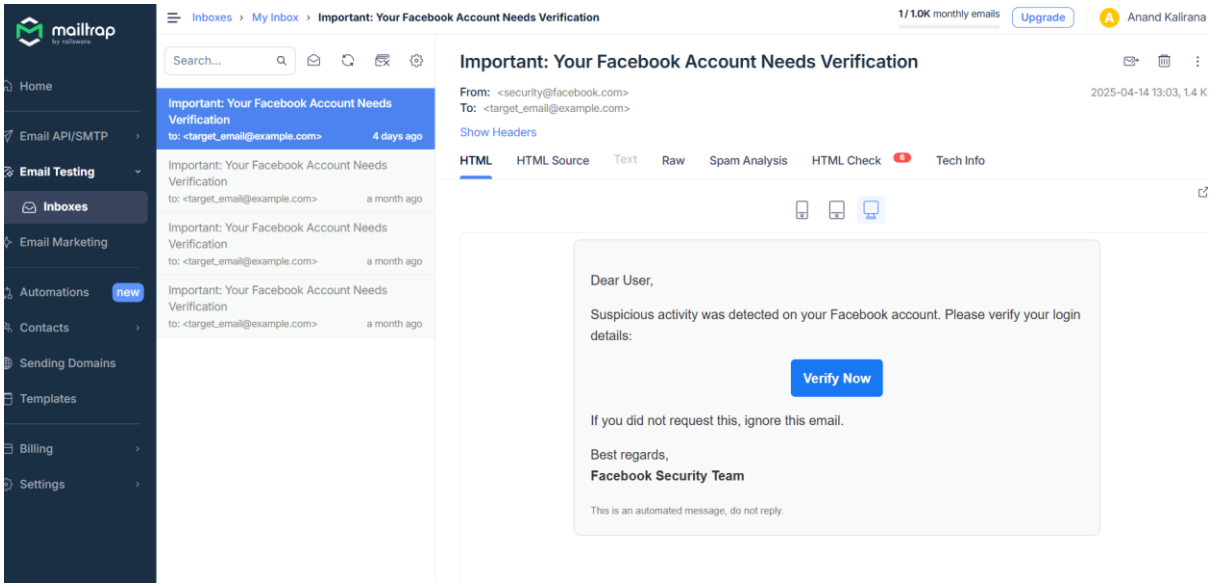
Mail Configuration for Mailtrap (SMTP)

This screenshot shows the Mailtrap SMTP configuration in the .env file of the Laravel project. Mailtrap is used as a sandbox SMTP server for testing email delivery during development.

Key Configuration Parameters:

- MAIL_MAILER: Defines the mail transport method, set to smtp.
- MAIL_HOST: Mailtrap SMTP server URL (sandbox.smtp.mailtrap.io).
- MAIL_PORT: Port used by Mailtrap, set to 2525.
- MAIL_USERNAME & MAIL_PASSWORD: Credentials provided by Mailtrap for authentication.
- MAIL_FROM_ADDRESS: Email address used as the sender for phishing simulation emails.
- MAIL_FROM_NAME: Name shown as sender (set to "phishing-simulation").

This setup allows the system to send test emails without risking real email delivery during phishing simulations.



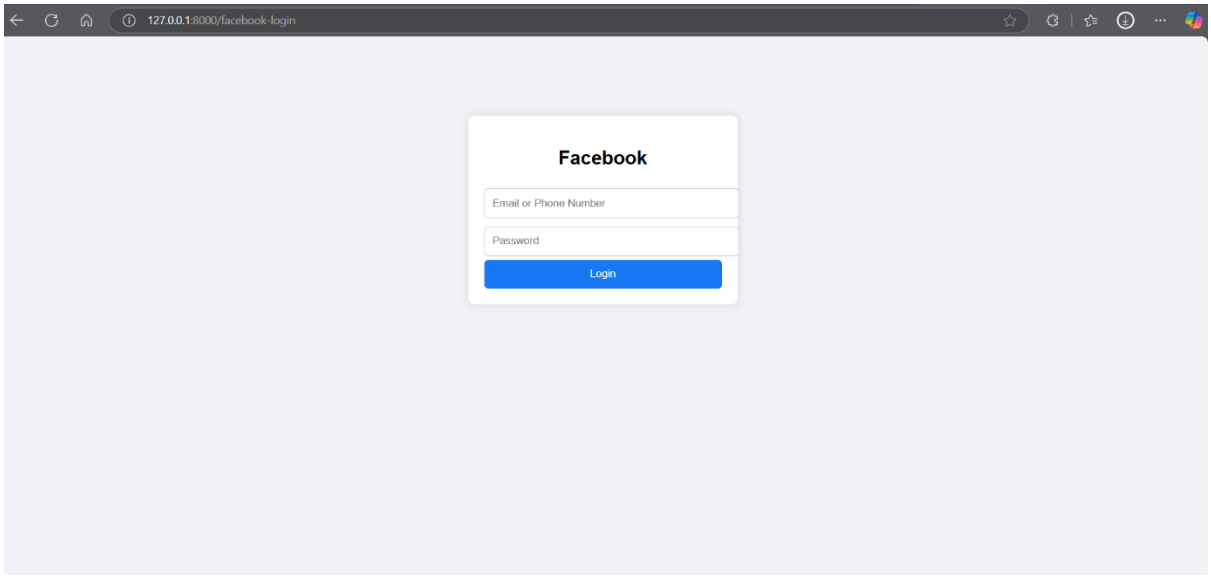
phishing simulation email received in Mailtrap

In this task, I developed the core feature of sending phishing simulation emails using Laravel. I integrated Mailtrap as the SMTP service for testing email delivery in a secure, non-production environment. The emails are designed to mimic real phishing attempts for the purpose of training and awareness.

The sample phishing email simulates a security alert from "Facebook," prompting the user to verify suspicious login activity, which is a common social engineering technique used in real phishing attacks.

Features Implemented:

- Setup Laravel Mail configuration with Mailtrap credentials.
- Created a phishing email template mimicking a real-world alert (Facebook login alert).
- Programmed Laravel's Mailable class to send emails with dynamic user data.
- Triggered email sending during phishing campaign simulations.
- Ensured email rendering and delivery inside Mailtrap's secure inbox.



Fake Facebook Login Page for Phishing Simulation

This screenshot displays a simulated Facebook login page created as part of the phishing awareness platform. The page is hosted locally at 127.0.0.1:8000/facebook-login.

Purpose:

- Imitates a real Facebook login form to simulate a common phishing attack vector.
- Designed to capture user credentials when submitted, log the attempt, and then redirect the user to the actual Facebook page, completing the simulation.

Key Features:

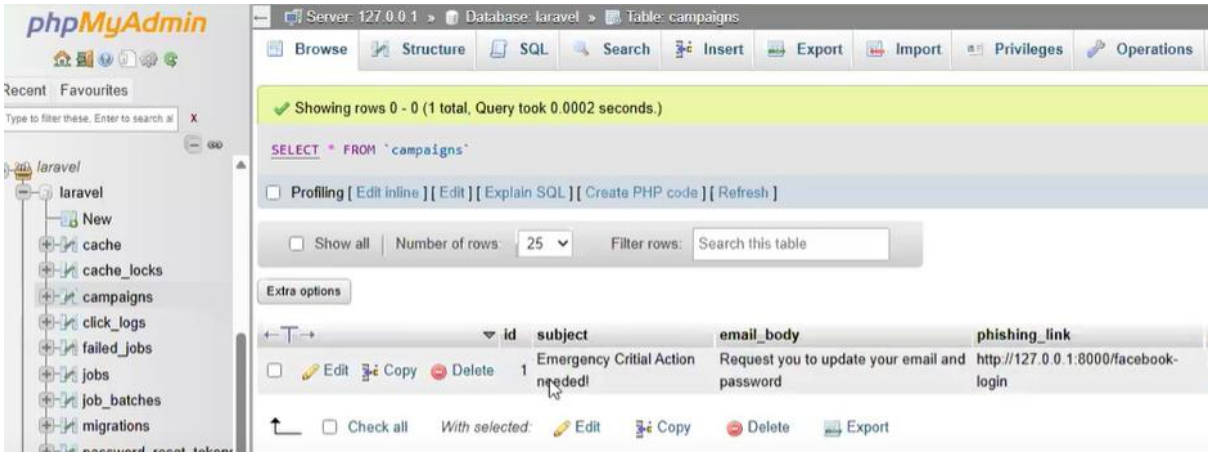
- Simple, responsive HTML & CSS to closely mimic Facebook's UI.
- Integration with the backend to store submitted credentials and log timestamps and IP.
- Used to educate users on recognizing suspicious login prompts.

II. Task 2: Implement Full CRUD for Campaign Management

This task involved building a backend admin panel to manage phishing campaigns.

Features Implemented:

- Create Campaign: Admin can create a new campaign with fields such as title, description, fake URL, email content, and status.
- Read Campaigns: List view with pagination to see all campaigns.
- Update Campaign: Admin can edit an existing campaign.
- Implemented using Laravel's Resource Controllers and Blade templates.



Campaign List Page (Read)

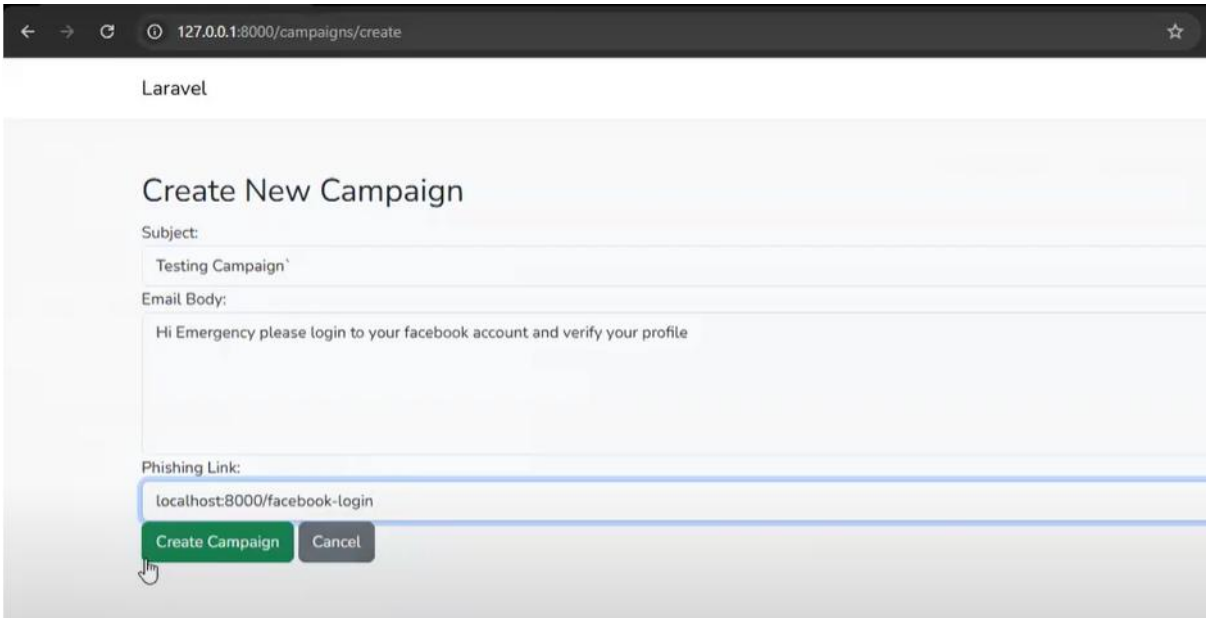
This screenshot displays the campaigns table within phpMyAdmin, showing a stored phishing campaign entry in the system's database. It serves as backend confirmation that campaign data entered through the admin interface is being successfully stored.

Key Details Visible:

- ID: 1 – Unique identifier for the campaign.
- Subject: Emergency Critical Action needed! – The email subject that the simulated phishing email would display.
- Email Body: Request you to update your email and password – The message content sent to targeted users.
- Phishing Link: http://127.0.0.1:8000/facebook-login – The fake URL used to simulate the phishing attempt.

Purpose:

This view is part of the Read functionality in the CRUD cycle, verifying that the campaigns created via the Laravel admin panel are accurately recorded in the backend database.



Add New Campaign Form (Create)

This screenshot illustrates the admin interface for creating a new phishing campaign within the Laravel application. It showcases the input form that allows administrators to define the contents of a simulated phishing email.

Fields Displayed:

- Subject: Testing Campaign – The subject line that will appear in the recipient's inbox.
- Email-Body:
  Hi Emergency please login to your facebook account and verify your profile-This message is crafted to entice users into clicking the fake link.
- Phishing-Link**:** localhost:8000/facebook-login – The URL where the simulated phishing form is hosted.

Actions:

- Create Campaign Button: Triggers form submission to store the campaign in the database.
- Cancel Button: Allows users to back out of campaign creation.

Backend Details:

- This form sends a POST request to the store() method of the CampaignController using Laravel's resource routing.
- Blade templating is used to render the form, providing CSRF protection and input validation.

III. Task 3: Simulate Credential Capture and Redirection

This task focused on simulating how a phishing attempt might work from a user's perspective.

Functionality Implemented:

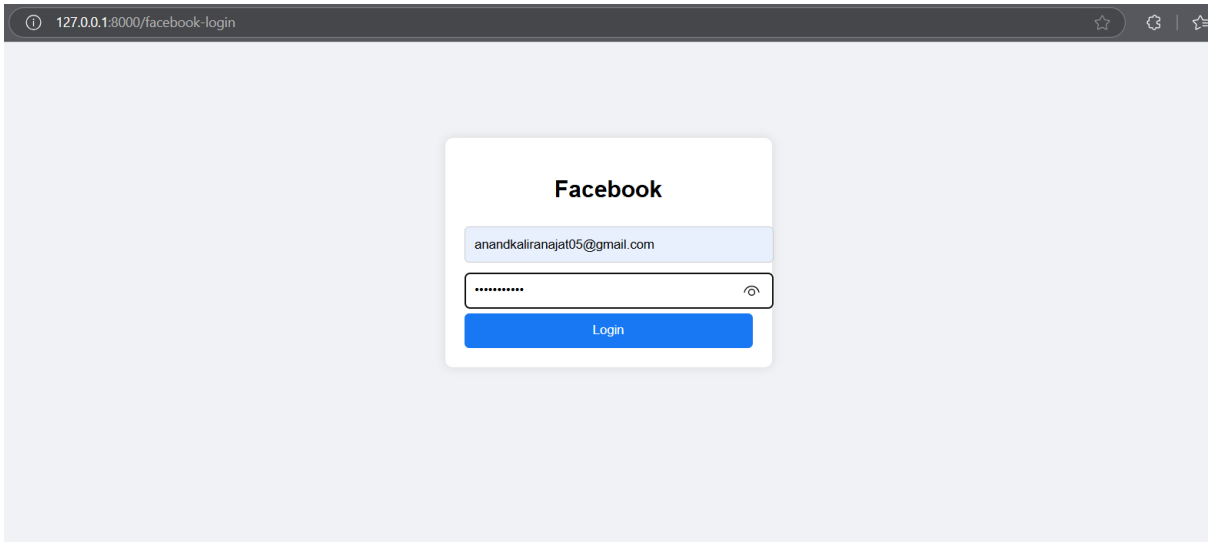
- Credential Logging:

When a user submits the fake login form, the following are logged:

  - Email address
  - Password (can be optionally masked in admin view)
  - Timestamp

- Redirection:

After submission, the user is redirected to the legitimate login page (e.g., Gmail or Microsoft), simulating the real phishing experience without actual harm.



Form Submission on Simulated Login Page

This screenshot displays the fake login form hosted on the Laravel application at 127.0.0.1:8000/facebook-login, simulating a Facebook login interface.

UI Elements:

- Email Field: anandkaliranjat05@gmail.com — Pretend victim's email input.
- Password Field: Entered and masked — Mimics a real login form behavior.
- Login Button: Triggers form submission.

Functionality:

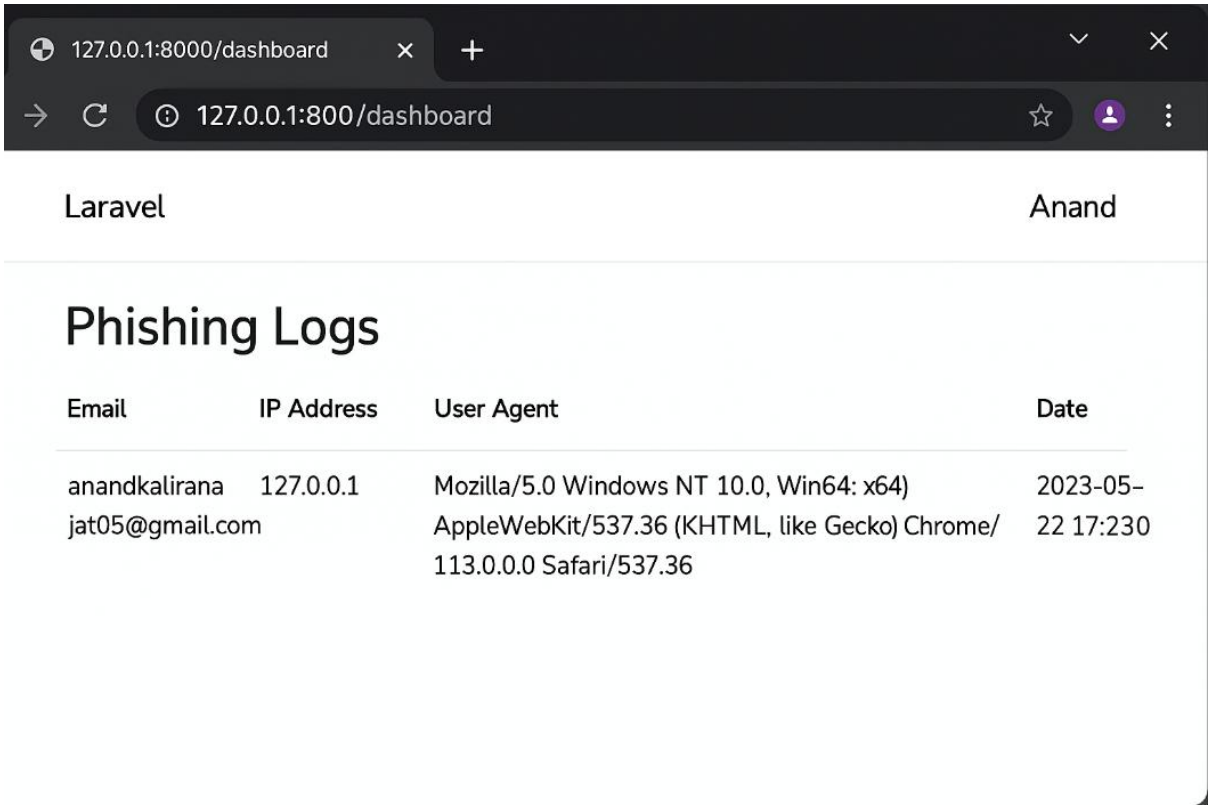
- This form captures both email and password when submitted.
- Data is stored in a secure database table, simulating how real phishing pages steal credentials.
- Once submitted, the user is redirected to the real Facebook/Gmail login page (handled in a controller).

Technical Backend:

- Route: Route::post('/facebook-login', [PhishController::class, 'captureCredentials'])
- Controller Method: captureCredentials() stores input in a failed_jobs or custom phished_users table.
- Redirect: Laravel redirect()->away('https://facebook.com') or similar function ensures redirection after logging data.

Ethical Context:

This page is built solely for educational simulation purposes to train users on how phishing works and demonstrate red flag indicators.



Phishing Logs Dashboard

This screenshot captures the Laravel dashboard at 127.0.0.1:8000/dashboard, displaying the phishing logs collected through the simulated phishing campaign.

Table Columns:

- Email: anandkalirana...@gmail.com — Captured from the fake login form.
- IP Address: 127.0.0.1 — Localhost, indicating the simulation environment.
- User Agent: Browser and OS details captured at the time of phishing.
- Date: Timestamp showing when the credentials were submitted.

Technical Backend:

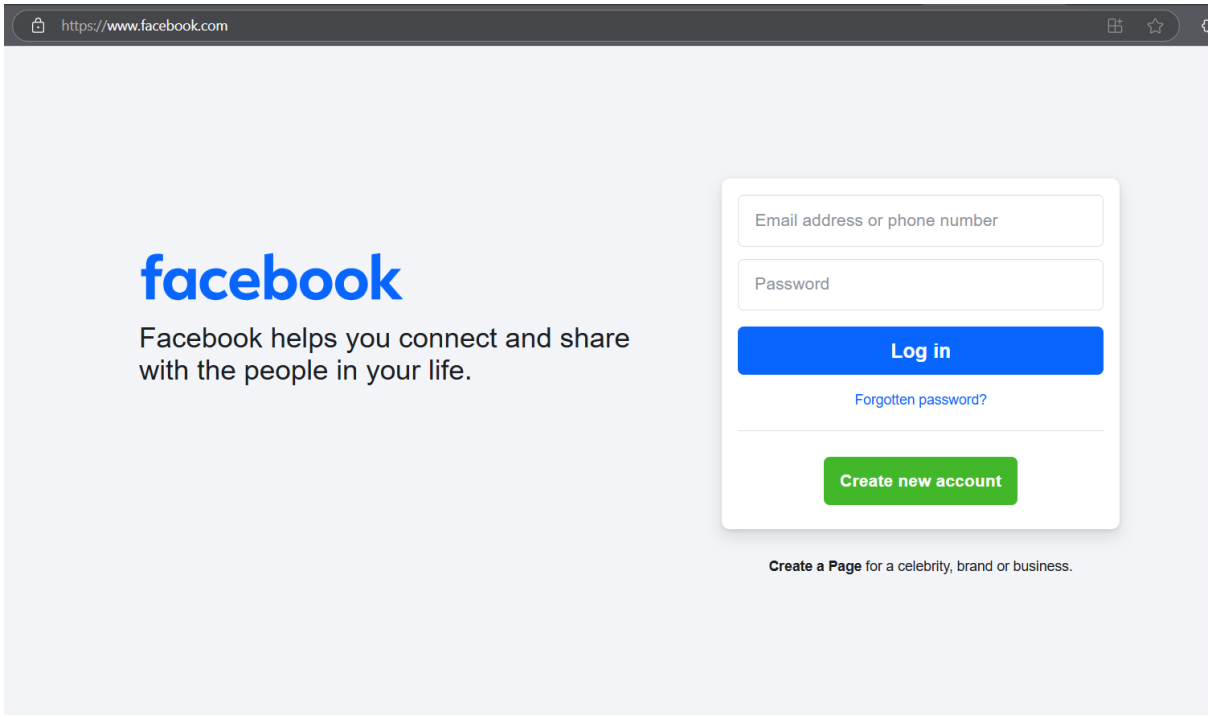
- This data is saved to a phishing_logs table (or similar) using Eloquent in Laravel.
- Populated via captureCredentials() method from the phishing form.
- Data fetched and displayed using a controller like DashboardController@index.

Purpose:

This dashboard mimics how real attackers view and manage stolen credentials. In this case, it helps security professionals visualize how phishing works and how to detect it on real systems.

Learning Objective:

Shows the result of the simulated phishing process and provides a clear, visual proof of concept for credential capture.



Redirection to Real Site

In this task, I created a realistic phishing page that mimics Facebook's login interface. When a user enters their credentials on this simulated page, the input is securely captured and stored in a log table (for awareness/reporting purposes). Immediately afterward, the user is redirected to the real Facebook login page to maintain the illusion of a legitimate experience and to conclude the simulation.

This redirection helps demonstrate how phishing attacks deceive users and why it's important to recognize warning signs before inputting credentials.

Features Implemented:

- Created a fake Facebook login page using HTML/CSS within Laravel Blade templates.
- Captured user input using Laravel form submission logic.
- Stored credentials and metadata (timestamp, IP address, campaign ID) into the database for reporting.
- Used Laravel's `redirect()` function to take the user to the actual `https://www.facebook.com` site.
- Ensured the simulation does not harm or compromise any real accounts.

Technologies Used:

- Laravel Routing and Controllers
- Blade Templates (for phishing page)
- MySQL (logging credentials)
- HTML/CSS (UI design)
- Laravel Redirects

Project Outcomes:

- Successfully developed a functional Phishing Awareness Simulation Platform using Laravel.
- Demonstrated how phishing emails are crafted, sent, and received using Mailtrap.
- Built and tested a realistic phishing login page to simulate attacks safely.
- Captured and logged user interactions for analysis and awareness.
- Ensured ethical handling of simulations with no real account compromises.
- Helped understand real-world phishing techniques in a secure, educational setup.

Challenges Faced:

- Designing realistic yet ethical phishing templates without triggering spam filters.
- Ensuring email delivery in test environments like Mailtrap while keeping formatting intact.
- Creating an authentic-looking phishing page that mimics original platforms accurately.
- Managing routing and redirection securely within Laravel.
- Logging captured data without violating ethical or legal boundaries.

Learnings and Skills Gained:

- Practical understanding of phishing techniques and prevention strategies.
- Laravel-based web development and backend email handling.
- Working with Mailtrap, SMTP, and dynamic Blade templates.
- Data capture and logging within Laravel using MVC architecture.
- Real-world application of cybersecurity awareness through simulation.

Conclusion:

This internship project provided a hands-on learning experience in the domain of cybersecurity awareness and Laravel-based web development. It simulated the lifecycle of a phishing attack — from crafting the bait email to capturing login attempts — all within a controlled and ethical environment. The outcome contributes to better understanding of phishing tactics and helps build resilience against such social engineering threats.

Acknowledgement:

I would like to express my sincere gratitude to Onestop AI for providing me with the opportunity to undertake this internship project titled “Phishing Awareness Simulation.”

This internship has been a highly enriching experience, allowing me to apply my academic knowledge in a real-world context while learning valuable industry practices.

I am especially thankful to my mentor and the entire team at Onestop AI for their continuous guidance, support, and encouragement throughout the project. Their insights and feedback played a vital role in the successful completion of this work