
POLITECNICO DI TORINO

Master degree course in Cybersecurity

Master Degree Thesis

# PhiShield: Design and Implementation of a Web Platform for Cybersecurity Awareness Programs

**Supervisor**
prof. Andrea Atzeni

**Candidate**
Federico Ciminelli

December 2025

# Summary

Cybersecurity awareness has become essential in countering the rise of social-engineering attacks, which increasingly exploit human weaknesses through highly convincing and personalised attacks, with phishing, in particular, remaining one of the most damaging threats. This evolution exposes a persistent challenge: the human element continues to represent a critical vulnerability, particularly when users are not supported by continuous, contextual and adaptive learning experiences.

In response to this challenge, the thesis is grounded in the principles of embedded learning and AI-driven personalisation, promoting cybersecurity awareness not as a set of isolated activities but as an ongoing, integrated process. Embedded learning promotes training directly within users' workflows, ensuring that educational content is contextual, relevant and immediately applicable. Artificial Intelligence enhances this paradigm by enabling dynamic content generation, behavioural analysis and personalised learning support. Together, these principles provide a foundation for a modern awareness model that aims to bridge the gap between theory and everyday security-related behaviour.

Despite the relevance of these concepts, current market solutions, especially within open-source panorama, tend to focus mainly on phishing simulations while offering limited support for structured and adaptive learning. This separation forces organisations to adopt external Learning Management Systems (LMS), resulting in fragmented experiences and increased operational complexity. To address this gap, the thesis introduces PhiShield, a prototype platform specifically designed to integrate phishing simulations and learning paths within a unified, automated and AI-enhanced environment.

PhiShield natively incorporates embedded learning principles and leverages LLMs to generate contextualised cybersecurity content, assisting users through a chatbot and supporting operators in designing customized campaigns. The platform is built around two principal modules: training and learning, intended to deliver practical simulations, personalised lessons and continuous assessments. Developed using Django within a multilayer architecture, it provides automation, reliability, scalability and a coherent workflow aligned with different user roles.

The platform supports three primary use cases: stand-alone phishing simulations for baseline assessment; training combined with formative lessons; complete awareness cycles integrating simulations, learning and examinations. This last scenario fully reflects the thesis's conceptual model, providing an end-to-end embedded awareness lifecycle that continuously adapts to users' behaviour and strengthens their ability to detect and respond to social-engineering threats.

PhiShield was evaluated across two testing phases, during development and in a real-world deployment involving users from heterogeneous professional backgrounds. Results indicated that while participants generally understood core cybersecurity concepts, risky behaviours persisted, confirming the need for continuous, contextual and behaviour-aware training. Users involved in the embedded learning flow successfully passed the theoretical exam but some still demonstrated weak practical reactions, highlighting the gap that the proposed model aims to address. Overall feedback confirmed that the platform delivers a clear and effective experience, with particular appreciation for its integrated design, realistic content and AI-assisted support.

This project demonstrates the feasibility and innovative potential of combining Artificial Intelligence with embedded learning principles to redefine cybersecurity awareness. PhiShield represents a foundational step toward creating adaptive, scalable and context-aware training ecosystems capable of aligning simulations, instruction and behavioural analysis within a unified workflow. Future developments include anonymisation strategies to increase privacy support, dynamic AI-driven campaigns tailored to individual behaviour, expanded "X-ishing" techniques like smishing (SMS), voice calls (vishing) and qishing (QR codes) and multi-tenant architectures to support large-scale deployments. Collectively, these directions strengthen the platform's vision of becoming an advanced and comprehensive open-source tool for improving organisational cybersecurity posture through truly integrated and intelligent awareness processes.

# Contents

# Chapter 1

# Introduction

In today's IT environment, characterized by rapidly growing IT vulnerabilities and sophisticated threat vectors, the resilience of companies heavily relies on their ability to identify and respond to potential risks in a timely manner, thereby mitigating possible damage to their assets. Social engineering-based attacks represent one of the most critical challenges, as they exploit "human" vulnerabilities. Among these, phishing stands as one of the most prevalent and effective technique worldwide, making security awareness programs one of the most impactful initiatives that organizations can implement to counter such threats.

Alongside this, the evolution of learning methodologies has led to the diffusion of innovative approaches such as *Embedded Learning*, whose objective is to move beyond traditional, static, learning toward more immersive and contextualized activities. This approach transparently integrates learning materials, guidance and feedback directly within the operational workflows of users.
This thesis fully embraces this principle: the idea that learning should not be a separate or isolated process but, in order to be truly effective, should be seamlessly embedded in everyday activities, enabling security awareness to grow "on the field" following a continuous process, based on contextual practice and immediate reinforcement.
Within this technological context, a significant impact is introduced by *Large Language Models* (LLM), systems capable of understanding and generating natural language, adapting to specified contexts and providing support in a flexible and customised manner. Their increasing reliability and scalability unlocks new range of possibilities for integrating "intelligent" assistance within learning paths, contributing to make the user experience more natural and contextual.

Based on these foundations, this thesis presents PhiShield, a web-based platform designed to natively combine phishing simulations and learning paths within a single and automated ecosystem, enhanced through the integration of LLMs and built upon embedded learning principles. The objective is to provide a solid basis for the development of an innovative solution that supports companies in delivering effective security awareness training to their employees, while ensuring a smooth, coherent and engaging experience. At this stage, the implemented solution represents only a prototype: an initial implementation intended to demonstrate the

feasibility of the proposed conceptual model, assess its concrete benefits and limitations outlining potential directions for future developments.

## 1.1 Motivations

The motivations behind this project are driven by the important pressing that threat actors apply against everyday activities such as reading a simple email and clicking on a link inside it, which could cause huge problems firstly on the individual but also to its company as a consequence. Studies indicate that over 90 percent of cyber breaches involve some form of human error, and phishing remains one of the primary attack vectors [1]. This highlights that, despite the most advanced technologies are used by the companies for defending their assets, it happens very often that the main cause of email compromising is the so called "human factor", which involves aspects that only humans are vulnerable of, such as curiosity, pressure or urgency. Consequently, implementing strong and continuous security awareness programs should be very critical. By training employees to recognize and respond appropriately to phishing attempts, organizations can significantly reduce the likelihood of a successful attack. Evidence shows that realistic, ongoing phishing simulations can decrease the percentage of employees who fall for this kind of attacks.
A notable example is a benchmarking study conducted by KnowBe4, a major player in security awareness field, against an heterogeneous group formed by a total of over 30.000 companies working across 19 different industry sectors, more specifically on 22.558 small organisations (1-249 employees), 5876 medium-sized ones (250-999 employees) and 1709 large organisations (1000 or more employees). KnowBe4 established a baseline to measure the effect of security awareness training, called *phish-prone percentage* (PPP). The base PPP is the fraction of users who clicked on simulated phishing emails prior to any security awareness training being delivered, which it has been measured as nearly 32.4 percent on average. After 90 days of training, phishing tests have been repeated, resulting into a consistent drop reaching 17.6 percent on average out of all industry sectors under education. Upon a whole year of security training, the PPP decreased with another substantial fall, hitting an average of just 5 percent overall.
The final result of this case study equates to an 87 percent improvement among all the different companies engaged and clearly demonstrate the benefits of providing security awareness training to employees, proving that such programs must be considered as crucial point through the non-stop process of cybersecurity awareness [2].
It is important to highlight that engaging employees within a single awareness program is not sufficient to build the necessary mindset for identifying and responding to phishing-related threats. In fact, cybersecurity awareness cannot be considered a one-time achievement but rather a continuous process that requires regular activities. Continuous learning initiatives allow employees to cover the gap with the evolving tactics used by attackers, maintaining high levels of attention over time. Therefore, organizations should aim to establish an ongoing training strategy, where awareness is constantly provided via practical simulations and interactive learning tasks.

### 1.1.1 Embedded learning approach

Embedded learning is a concept that fits directly within the working environment, integrating educational activities into employees' daily routine. Typically, this approach is triggered by specific actions or errors made by the individual, transforming mistakes into immediate learning opportunities. This concept aligns perfectly with the objectives of security awareness, as it allows employees to learn from real scenarios rather than through abstract learning initiatives, often lacking in reality. As supported by the literature, embedded learning increases engagement and knowledge retention, turning everyday tasks into continuous learning opportunities. In particular, one of the key benefits of this approach is the realness of its context-aware learning paths, which are directly tied to concrete scenarios rather than offering generalised material, which results to be more difficult to acquire for employees. Moreover, another important aspect is the idea of "learning by doing" which allows to learn specific topics immediately by undergoing designed practical activities, supporting an efficient and durable improvement of cybersecurity awareness [3].
Different recent studies confirm the effectiveness of this approach, specifically in cybersecurity-driven contexts where employees awareness is the main fundament toward a strong and reliable security posture. In particular, a study conducted by a group of researchers at the Federal University of Lafia, Nigeria, proposes an innovative system that combines realistic phishing simulations with learning activities directly integrated within the operative users' experience. This immersive approach resulted into clear evidences that proved how the adoption of embedded learning strategies could significantly decrease phishing-related risks, with an outstanding 64 percent reduction of success for attackers, an overall improved accuracy in detecting malicious emails, enhancing the supervisory attitude of involved users. Lastly, the paper outlines how the adoption of embedded learning approaches, within a context of continuous evolution of attack techniques as cybersecurity landscape, is extremely significant to keep up, constantly updating learning material and varying training exercises, aiming to build a resilient stance in every single employee [4].
The main objective of this project is to design a platform that natively supports this learning approach, introducing additional benefits in terms of time efficiency and cost reduction. In fact, PhiShield completely removes the need to install, manage and update additional software enabling the seamless creation of a complete awareness path within a single application. This integrated structure not only simplifies management but also leads to significant cost savings, allowing organizations to establish a continuous and sustainable process aimed at reinforcing the cybersecurity knowledge and awareness of their employees over time.

### 1.1.2 LLMs influence

Artificial intelligence (AI) based on generative architectures, is rapidly transforming the whole cybersecurity landscape, offering advanced tools to enhance detection and prevention of incoming threats. AI could be used to automate tasks and processes, such as continuous threat monitoring, predictive attack mitigation or even supporting security teams during incident response operations. To give a concrete

example, specific machine learning algorithms can be deployed to analyse large loads of data, such as network traffic, with the aim to detect anomalies in advance and eventually avoiding significant harms to the company's infrastructure. Obviously, security awareness is not excluded, in fact the application of LLMs enables the creation of customised and interactive learning paths, adapted to the context of a specific organisation or the behaviour of a single user involved. In this case, algorithms could be leveraged for developing more realistic phishing simulations, based on real scenarios and trending threats, maintaining an high level of attention and ensuring a complete preparation. Moreover, AI's ability of generating and comprehending natural language allows to seamlessly integrate context-aware components to directly interact with end users, providing them with support and feedback, enhancing "on-demand" learning.
In this project, the integration of a simple chatbot will be discussed in the following chapters, constituting just a preliminary phase toward deeper usage of artificial intelligence within the context of security awareness. By addressing future developments suggestions, described in the final chapter of this thesis, PhiShield can be extended toward a concrete inclusion of proactive LLMs, being able to predict learning and training necessities, automatically adapting their contend to guarantee a continuous improvement over time.
To conclude, generative AI represents a tangible frontier to revolutionise cybersecurity awareness programs, making it more efficient, scalable and adaptive with respect to traditional learning approaches, which usually lack in touching contextual and important aspects that should be addressed for specific company backgrounds.

## 1.2 Context

### 1.2.1 Social engineering

Phishing attacks fall under the broader category of social engineering, also known as *human hacking* which is often defined as a practice of using psychological influence and deception to trick people into performing unwanted actions usually for fraudulent or unauthorized access purposes. Attackers are able to exploit human psychology leveraging emotions like fear, urgency or even trust to gather sensitive information that could be used in a more complex scenario. In fact it happens very often that social engineering techniques, such as phishing attacks, are primarily employed to obtain the so-called *initial access* to an organization, with the subsequent objective of inflicting significant harm on that targeted entity. Most social engineering attacks employs at least one of the following techniques:

- **Trusted entity impersonalisation**: attackers aim to impersonate known and trusted companies to let the victim fall more easily.

- **Posing as a government authority**: leveraging fake authorities messages could lead the victim to doubting and fearing possible consequences of not interacting with the phishing email.

- **Inclusion of fear and sense of urgency**: very common technique, it consists into writing phishing messages with matter of urgency, in order to let the victim to act rashly and under pressure.
- **Appealing to greed or curiosity**: take advantage of human curiosity, for example with a fake survey, to let the target interact with the email.

The impact that a successful phishing attack could have on targeted organisations is huge and can harm companies from different perspectives, in fact, the company might experience: economic damage, due to direct theft of assets or to fines due to data breaches; reputational loss, loosing customers' trust and directly damage the image of the company; operational loss, due to downtimes and interruption of businesses. Therefore, phishing attacks must be considered as a strategic and impacting risk and not only as a purely technical issue to address. These sophisticated but low cost attacks can be mitigated by employing a various set of practical solutions, for example by activating strict access control policies, including *multi-factor authentication*, to limit the activity of cybercriminals even if their phishing attack is successful, or enabling specific cybersecurity software, such as firewalls and extended detection and response (XDR), to limit the eventual interaction that victims could have with incriminated emails. Moreover, having a strong security awareness training program is highly critical as the mentioned software solution cannot prevent every single attack performed by malicious actors.

### 1.2.2 World scale impact

The relevance of phishing attacks is underscored by the alarming statistics that emerge annually from various cybersecurity reports, that record millions of individuals and organisations falling into phishing campaigns. The increasing sophistication of these attacks coupled with the ease with which they can be designed, necessitates ongoing research and the development of advanced prevention mechanisms as well as strict enforcement of security awareness programs.
Phishing is one of the most invasive cyber threats worldwide, with approximately 3.4 billion phishing emails sent daily (1.2 percent of the global email traffic). Its effect goes beyond, in fact phishing is considered the source of 36 percent of all data breaches, making it the most exploited method for attackers to gain an initial grip into the victim's infrastructure. The economic consequences are equally severe, with an average phishing related data breach charging organisations about 4.88 million dollars, where *Business Email Compromise* (BEC), which was the most destructive variant, caused over 2.7 billion dollars losses in the United States alone in 2024 [1]. Luckily, structured phishing awareness training has proven highly effective increasing by 30 percent the resilience of employees against phishing threats and 70 percent reduction in overall security related risks, underscoring the significant impact that comprehensive training programs could have on companies' security posture [5].

These results underline that while phishing remains a relentless and costly threat, proactive and interactive training through realistic simulation tools can transform employee behaviour and strengthen the global organizational resilience.

## 1.3 Document structure

This paper illustrates the activities conducted within the project, which had the main objective of implementing a simple but functional security awareness platform to allow companies creating stable basis and knowledge into such a risky environment as social engineering and in particular phishing. The following chapters revisit the main phases of the project:

- **Chapter 1: Introduction**
  Presentation of the project and general explanation of the context and the motivations underneath.
- **Chapter 2: Background**
  Global overview of phishing attacks and techniques, with particular emphasis on the relevance of a strong security awareness training program within a company. Exhaustive analysis of tools already available on the web with same set of functionalities.
- **Chapter 3: Design**
  Project requirements and design choices implemented to address them.
- **Chapter 4: Implementation**
  Details related to implementation phase with comprehensive enumeration of tools and technologies engaged as well as flows of operations.
- **Chapter 5: Testing**
  Description of the testing approach selected with contextual commentary of results.
- **Chapter 6: Conclusions**
  Final considerations about the project, critical aspects and further developments.

# Chapter 2

# Background

In the first chapter, a general context about the working environment in which PhiShield could operate has been outlined, understanding which are the roots and the main psychological techniques introduced and exploited by threat actors that aim to create a successful phishing campaign as well as the importance for companies of having a sharp security awareness program that aims to raise awareness in all employees and reduce compromise likelihood. However, in order to create a useful training program, it's not enough to just be aware of such general techniques, but it's also important to dig deeper and understand every single trick that cybercriminals use to build their attacks, with the aim of training employees with scenarios that they could face in reality, in a way that they will be conscious about threats they could tackle with possible related consequences for them and their company. In the following sections a detailed overview of both sides, one end represented by phishing and the other one by awareness programs, is depicted.

## 2.1 Phishing

Phishing, as seen earlier, is part of social engineering activities and is considered as type of cyberattack that uses emails and similar communication means to trick users with the final aim of achieving compromise, which sometimes means to get access to a specific email account and exfiltrate sensitive data, but very often it's used as a starting point for a broader attack, where the phishing email is exploited to allow the attacker to gain an authorized access to private resources, trying to hide evidences of the concrete attack. For this reason, it's quite uncommon to face against extremely sophisticated phishing attacks, at least from the technical point of view, in fact the necessary components that must be set up by attackers are:

- **Phishing domain**, to use for sending malicious emails;
- **Emulated web site**, to trick the user making him feel it's the legitimate one;
- **Web server**, to receive any sensitive data exfiltrated from the victim, account's password included. Very often this server will be the same that hosts the cloned web site.

### 2.1.1 Techniques

Usually attackers perform so-called *information gathering* activities, were they collect victim's related information, in a way that they will be able to craft more convincing and targeted phishing emails. This phase primarily involves leveraging various online tools to collect information about potential victims and their associated networks, such as their email address, their job position and eventually other personal information.
Moreover, nowadays, it's very usual to find tools on the internet that are able to cover all necessary setup operations at once, often named *phishing kits*, allowing the attacker to just deploy it on a virtual machine and prompt the list of email addresses they want to target, launching the campaign straight forward, which obviously facilitates these malicious activities. The most common characteristics exploited by threat actors in building phishing emails are:

- **Domain spoofing**: exploit weaknesses in email authentication protocols to falsify the sender's address, making emails appear to originate from trusted domains. This technique relies on user trust in familiar brands and organizations to increase credibility;
- **URL manipulation**: hyperlinks that redirect victims to fraudulent websites crafted to harvest credentials or install malware. Techniques include typosquatting and obfuscation through URL shortening. The first method consists in registering domains that are intentionally similar to legitimate ones (e.g. *goggle.com*, *amazno.com* or *youtube.con*); the second one leverages URL shortening services to mask the malicious url (e.g. *https://example.com/product?ref=01652* becomes *https://shorturl.com/1418u* after using the service);
- **Malicious attachment**: files disguised as legitimate documents but embedded with malware, ransomware or malicious macros. Common formats include PDFs, Microsoft Office files and compressed archives. Once opened, these attachments enable unauthorized code execution on the victim's device and eventually exfiltrate sensitive data;
- **HTML tricks**: exploit HTML and CSS features to disguise malicious intent, such as hiding links under legitimate-looking text. Techniques include overlaying transparent elements, obfuscating code or rendering invisible characters. These methods target both human perception and automated filter evasion.
- **Social engineering**: as previously discussed, attackers frequently exploit psychological factors such as urgency and fear to pressure individuals, increasing the likelihood of impaired reasoning and poor decision-making.

In most cases, attackers do not rely on a single manipulation strategy but they rather combine multiple techniques at once. This approach enhances the plausibility of their deception, serving the goal of constructing a phishing email that appears credible and trustworthy to the intended victim.

### 2.1.2 Methods

Phishing comprehends a various set of attack types, each exploiting human trust and technological weaknesses to achieve fraudulent goals. While traditional phishing relies on mass distribution, modern variants have evolved to incorporate personalization, alternative communication channels and delivery mechanisms. The subsection below provides an overview of the most relevant phishing methods, highlighting their major aspects and implications for organizational security.

- **General phishing**: the most common and traditional form, where attackers send bulk emails hoping that a small part of recipients will interact and fall into the trap. These messages often mimic banks, online services or government agencies to trick users into providing sensitive information. The success of general phishing relies on scale rather than sophistication, as thousands of emails can be sent at very low cost. For companies, this means that even employees with limited privileges can become entry points for attackers.

- **Spear phishing and whaling**: variant targeting specific individuals or groups using information gathered from public sources or previous breaches. Because the messages feel relevant to the recipient, they are much harder to detect compared to generic phishing attempts. A particular subtype is *whaling*, which focuses on senior executives or directors, as compromising their accounts often leads to confidential data and high privileges on financial or other valuable assets. For organizations, these targeted attacks are a major risk since they exploit insider knowledge and authority hierarchies, potentially leading to severe data breaches or financial loss.

- **Business Email Compromise - BEC**: one of the most impactful phishing techniques, which combines impersonation and social engineering. Unlike typical phishing, BEC emails are often free of obvious red flags like links or attachments, making them more convincing. By posing as trusted figures such as CEOs or suppliers, attackers exploit organizational trust rather than purely technical flaws. From a security standpoint, BEC is particularly dangerous because it bypasses traditional detection systems, requiring strong internal procedures and protocols to mitigate.

- **Other phishing techniques**: Nowadays, phishing has expanded into alternative channels that take advantage of evolving communication means. *Quishing* uses QR codes to redirect victims to malicious websites, often disguised as payment processes. *Smishing* relies on SMS messages with urgent content, exploiting the speed and ease of mobile communications. Finally, *social media phishing* leverages platforms like LinkedIn or Instagram, where attackers impersonate brands or trusted contacts to distribute malicious links. These methods point out the need to extend security awareness beyond corporate email, since employees interact daily with personal smartphones, which may become potential attack vectors.

- **USB drops**: this is a more physical approach to phishing, where attackers leave infected USB drives in strategic locations such as parking lots, offices

or public areas. The idea is to leverage curiosity of employees and lead them to plug the device into a work computer, unknowingly installing malware. While it may sound low-tech and easy to avoid, this method has been shown to work surprisingly well because it targets human behaviour rather than system vulnerabilities. As well as for QR code or SMS phishing, USB drops illustrate the importance of overcoming minimal security awareness programs, introducing physical security policies and endpoint protection, since even a single compromised device can lead to a large-scale breach.

A breakdown of these methods reveals a clear evidence of the evolution that phishing emails have undergone over time, moving forward from "spray and pray" approaches, where attackers relied on bulk distribution, to highly sophisticated and targeted operations such as in spear phishing or whaling. This shift reflects a broader trend where attackers are investing more effort into information gathering activities, exploiting publicly available data to craft messages that closely mimic legitimate communications. Moreover, the recent growth of artificial intelligence has accelerated the process, enabling the automatic generation of convincing emails, realistic fake profiles, and even faked audio or video posted on social media accounts. These advances reduce the cost and effort of creating large numbers of tailored attacks while increasing their effectiveness.

For organizations' standpoint, this means that phishing is an adaptive threat that requires smarter defences, combining security systems and procedures with employee awareness.

### 2.1.3 Attack demonstration

To explore phishing in more detail, a simple yet comprehensive demonstration is presented. For the purposes of this document and for clear ethical reasons, the demonstration was conducted in a local, personal environment as the phishing tool was deployed on a personal computer. For the same reasons, this won't be a complete demonstration, but it will just give an overview of the activities required to set up a malicious framework. This activity aims not only to show how easily a malicious actor could set up a phishing environment, but also to accentuate the importance for organizations of perform effective security awareness training to its employees.
As previously discussed, attackers can rely on so-called phishing kits, which are stand-alone tools that do not require any additional utilities to perform malicious activities. One example is *SocialFish* [6], an open-source tool written in Python and published by UndeadSec on *GitHub*, which was originally implemented for penetration testers and other offensive security specialists and it makes possible to quickly set up an environment and begin sending fake emails. Its main features include the ability to "clone" a website by replicating its HTML and CSS code and deploying it on a malicious server, tricking the victim into believing they are interacting with the legitimate one. It also allows the configuration of a redirect URL, as the web location the user is sent after interacting with the fake web page. The tool can obviously capture submitted credentials as well as other information such as the victim's geolocation. In addition, SocialFish can support in distributing

phishing emails by enabling the creation of message templates and sending them to targeted users. This feature requires an SMTP server, although attackers can alternatively use a personal account.
To set up SocialFish, a few actions are required:

1. Clone the source code from the GitHub repository

   ```
   $ git clone https://github.com/UndeadSec/SocialFish/
   ```

2. Install Python3 and pip (if needed)

   ```
   $ sudo apt-get install python3 python3-pip python3-dev -y
   ```

3. Download dependencies

   ```
   $ cd SocialFish
   $ pip install -r requirements.txt
   ```

Now the tool is ready to be used. By running it with

```
$ cd SocialFish
$ python3 SocialFish.py demo demo
```

where the two arguments provided to the Python scripts are the username and the password for logging into the main web console. At startup, the script will show the URL to be used to perform the phishing attack. At the same address, by navigating to */neptune* endpoint, the web console will be prompted.
Additional material related to this demonstration is available in Appendix D.

In the top-right corner, there is a form for entering the URLs involved in the attack, specifically:

- **Clone URL**: the web page to mimic for the attack (e.g. Facebook login page)
- **Redirection URL**: the location where the user should be redirected to, after interacting with the malicious cloned web page

Let's look at a practical example, where the *clone* URL will be the GitHub login page *https://github.com/login*, while the *redirect* will point to a real GitHub error page, showing 404 HTTP status *https://github.com/fake-error*, to lead the victim into believing the login didn't complete successfully. The next step would be just a click on the flash icon next to the redirect URL, in order to deploy the malicious web page, GitHub login clone as for this case, which will result into a URL that if visited will firstly show an exact copy of the real login page, being able to record and store the prompted credentials and then redirecting the user to the fake error page.

At the bottom of the console, there is the live log of the successful attacks, which in this case shows the interaction just made, with the credentials used for the login attempt.

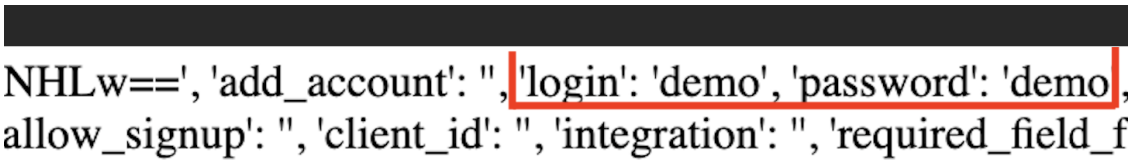


Figure 2.1. Exfiltrated credentials log

This interaction was done just to show the correct functioning of the malicious website, however, in a real attack scenario, the link would be inserted into a phishing email, embedded in a QR code or even delivered using SMS. To stay within the scope of this demonstration, the body of a possible malicious message could be as follows:

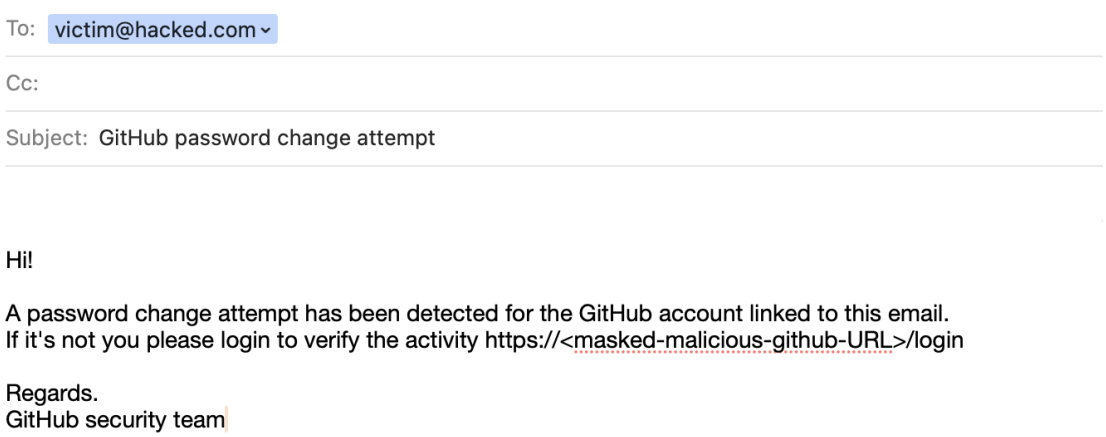


Figure 2.2. Phishing email example

This demonstration was carried out not only to highlight how simple it can be for a malicious actor to set up an effective phishing environment that mimics real-world websites, but also to reinforce the importance of building awareness among employees. Organisations must ensure that their staff are not only informed about the existence of such threats but are also trained to recognize suspicious signs, respond appropriately following secure practices. Developing this level of awareness transforms employees from being the weakest link into an active line of defence against phishing and similar forms of social engineering.

### 2.1.4 Relevant attacks

To provide a complete overview about phishing, two real-world attacks are briefly presented. The first scenario under exam is the 2020's Twitter account hijacking [7], which had its root cause in attackers using phishing via phone call (also called *vishing*) against Twitter employees tricking them into a malicious VPN portal, leading to credentials and MFA tokens compromise. With those credentials, attackers were able to completely bypass Twitter's security measures and consequently violate high-profile verified account, such as Bill Gates, Barack Obama, Warren Buffet and Jeff Bezos leveraging them to spread a bitcoin fraud via Twitter posts that promised to double the amount of bitcoin received from victim's accounts, within half hour. Approximately, 130 accounts were accessed and 45 used to post scam tweets with an estimated total amount of 12 Bitcoin involved. A huge reputational damage has been the consequence of this attack as well as operational disruption, where media organizations were forced to tweet from different accounts and for example the National Weather Service couldn't warn of tornadoes via Twitter. This resulted into a drop of nearly 0.9 percent stock market share price. Three individuals have been lately processed with fraud, identity theft and unauthorized access charges.
The second case depicted, happened in 2023, is the cyberattack that involved MGM resort and Caesars Entertainment [8], which fell victim of phishing-based attacks, notably by affiliates of *ALPHV/BlackCat*, a well known ransomware group. In this case phishing has been leveraged to gain access into victims infrastructure to further deploy and spread a double-extortion ransomware, encrypting data and exfiltrating the plaintext ones for coercion. Both companies experienced major service disruptions where check-in, guest loyalty and door-lock systems were offline during the attack.

These case studies underline how a single error can quickly escalate into financial losses, operational disruption and reputational damage. Strengthening resilience requires giving human knowledge the same priority as technical defences, through strong security training programs and awareness plans.

## 2.2 Security awareness programs

As previously discussed, despite engaging the most sophisticated security systems and controls, human behaviour continues to be a major risk for companies, because errors in judgement, urgency and lack of awareness aren't factor that an XDR system or a firewall can deal with, that's why enforcing security awareness training programs is at the first place as effective countermeasure against social engineering attacks, like phishing.
The goal of such plans is to educate employees about cyber risks and best practices, to build a *security-first* culture among them. It's not just about training for a marathon, it should be a behavioural transformation program that aims to spread the knowledge and change the approach passing from a compliance-driven one, towards a safer mentality. However, it's very difficult for companies to obtain this kind of transition, as they usually face against common challenges from employees

like training fatigue, when security programs are seen as boring and useless activities or lack of realism and generic content in simulations, that could lead to the failure of reflecting real life scenarios, thus making employees not enough aware of the threats and the attitude they should enforce to face such attacks.
A security awareness program is usually structured in different phases and should be designed as a continuous and cyclic activity, always proposing different contents and challenges, lining up with new trends and techniques used by threat actors. Typical components include:

- **Learning modules**: e-learning and videos about phishing risks, techniques and best practices;
- **Training modules**: actual phishing simulations and other activities, where users are trained by facing against ad-hoc phishing campaigns, obviously conducted by a professional team or the company itself;
- **Engagement modules**: quizzes and gamification approach, trying to attract users avoiding to bore them
- **Reporting tools**: such as "report phishing" buttons, deployed on users' email clients, collecting and analysing their interactions with phishing emails.

Conventionally, security programs follows a specific cycle, quite often very similar, which starts with the deploy of a phishing campaign, spreading emails to every targeted user eventually differentiating templates according to their role in the company. Then the interaction results are assessed, looking for employees that performed poorly and automatically enrolling them into specific training courses. As said before, there shouldn't be just a first round, but instead, the email templates should be fine-tuned every cycle, ensuring a dynamic approach to this training phase. Subsequently, all the users will have to follow security courses, participate at cyber-games and similar challenges, strengthening their knowledge by performing specific tests and quizzes to assess their actual improvements. After this learning phase, the first cycle could be considered as completed, hence the second one can be designed and organised taking into account all the results of the former one.

A company investing in effective awareness training and phishing simulations could spend only a fraction of the financial and reputational damage that a successful breach could cause, making it not just a security measure but a strategic requirement.

## 2.3 Security awareness platforms

To conclude this section, a brief analysis of the state of the art of security awareness tools is proposed, trying to frame what the market offers to companies, to support their security awareness initiatives. Generally, two categories of tool exists, which are phishing simulation tools, that help organisations to design and deploy their training campaigns and learning management systems (LMS), for providing learning resources to users. By analysing the current situation, it is possible to identify

a variety of solutions in the commercial sector, while there remains a clear scarcity of offerings in the open-source domain. Among the leading commercial platforms, several stand out thanks to their performance and capabilities, including:

- **KnowBe4**: one of the most popular platforms due to its broad library of learning content, covering both fundamental concepts and advanced topics in cybersecurity. Its usability, scalability and robust support for automated phishing campaigns constitute its key features, making it a user-friendly and learning-oriented solution [9].
- **Proofpoint**: originally founded to provide email security solutions, it has since evolved to include sophisticated capabilities that enable seamless integration between technical protection and user learning, allowing organizations to combine cybersecurity measures with human awareness initiatives [10].
- **Adaptive security** distinguishes itself for its machine learning-based approach, which allows to provide customised and coherent training with realistic and dynamic phishing campaigns. It is considered particularly effective for organizations aiming to implement behaviour-based simulations, based by in-depth analysis [11].

However, by moving the focus in the open-source domain, there is an evident lack of solutions that implement the Embedded Learning approach enhanced with AI, integrating in the same platform both training and learning capabilities. The most functional and mature open-source tool is by far GoPhish [12] which provides an agile platform with easy setup, intuitive campaign management and real-time reporting features. Nonetheless, it doesn't offer any learning-oriented functionalities, thereby limiting the scope of activities that companies could design in accordance with modern security awareness requirements.

To summarise, the current state of the art results in a market dominated prevalently by sophisticated commercial solutions, that combine advanced phishing simulations with immersive learning paths, both supported by the integration with LLMs. Despite being efficient and complete, they introduce non-negligible side effects for companies, such as a rapid escalation of costs due to user-based pricing and a strong dependency on the provider, requiring compliance with rigid licensing restrictions. On the other end, the absence of a comprehensive open-source solution that combines these functionalities, opens up a significant development opportunity for cybersecurity-oriented platforms, as PhiShield, meant to be accessible by everyone.

## 2.4 Role of Artificial Intelligence

Nowadays, AI represents a critical technology in the cybersecurity landscape since it could be used for both defensive and offensive activities. From defensive point of view, it could be engaged to automate the detection and the response to identified threats, for the automation of repetitive operations and the forecast of possible attacks using threat intelligence sources. In particular, advanced machine learning

algorithms are able to examine real-time data to identify anomalies and suspicious activities, even in the context of zero-day vulnerabilities, which are flaws that weren't discovered yet. This characteristic allows to overcome the limits of traditional detection methods, like signature-based ones. Moreover, AI can be employed to support companies during incident response procedures, potentially reducing the human intervention, thereby lowering the time required to mitigate the attack.
However, the advantages offered by AI can also be leveraged by malicious actors. For example, attackers can employ AI to automate large-scale campaigns and to generate sophisticated malware, inherently more difficult for traditional security systems to detect. Moreover, AI is increasingly used to craft highly customised phishing templates and to perform automated vulnerability exploitation, significantly enhancing the effectiveness of attacks and increasing the likelihood of success. This trend has become particularly evident in 2025, which has been identified as a critical year for cybersecurity, with 47 percent of documented attacks being AI-driven [13] [14].

### AI and security awareness

Artificial intelligence can be particularly effective in the context of security awareness, as it can directly support users in enhancing their cybersecurity knowledge. AI-based learning platforms are capable of adapting the content delivered according to the individual user's weaknesses, analysing performance in phishing simulations and tailoring learning materials accordingly, offering for example:

- Customised phishing simulations, calibrated according to the skills of each individual user;
- Chatbots and virtual assistant, to provide immediate feedback and support during learning activities;
- Continuous monitoring of user performance, aimed to provide coherent training scenarios, increasing the overall attention span and detection capabilities.

These approaches can concretely lower human-induced security incidents, spreading a more resilient and proactive security culture.

## 2.5 Why PhiShield

This project has been conceived to propose a solution that is currently absent from the market, offering a platform capable of reducing the gap between commercial and open-source solutions, while maintaining accessibility, openness and a set of functionalities that should be regarded as standard in today's cybersecurity landscape.
By combining AI-driven training simulations with an AI-supported learning environment, PhiShield aims to overcome the limitations of most existing open-source solutions, providing organizations, regardless of their budget or user base, with a comprehensive platform to support their security awareness programs, ultimately

enhancing both knowledge and resilience against such threats. From a critical perspective, the conceptual integration of embedded learning with artificial intelligence represents a significant advancement compared to the solutions currently available. With PhiShield, users can be involved in customised simulations designed to replicate real-world scenarios, allowing to test their ability to recognize and respond to suspicious emails while tracking and monitoring their progress. Additionally, they can be enrolled in tailored learning programs, consisting of concise cybersecurity courses aimed at enhancing general knowledge of these attacks and promoting best practices for practical mitigations.

Finally, the open-source nature of this project promotes transparency, flexibility and customisability, which are key aspects that organizations can leverage when adopting the platform, allowing them to tailor it to their specific context and requirements.
By developing a software platform that concretely incorporates these features, the opportunities to spread security knowledge are increased, thereby strengthening defences against such hazardous situations.

# Chapter 3

# Design

As previously discussed, PhiShield has two main foundations: the training module, which focuses on phishing simulations and the learning module, designed to deliver cybersecurity content in the form of both courses, called *lessons* and assessments, referred to as *exams* where the lessons provide security-specific material to platform users, while the exams are intended to verify whether this knowledge has been properly acquired. To successfully implement a complete security awareness platform like PhiShield, the design phase is the most important one, allowing to understand the necessities and the features of the project and how to develop suitable solutions. This chapter aims to give an overview of the critical reasoning and thinking carried out for this project, explaining the logic behind its main components and how requisites are meant to be addressed. Key topics including goals, system requirements and conceptual data flow are discussed, along with the implementation choices made, showing how they map back to those requirements.

## 3.1 Requirements

In this section, the main requirements of PhiShield and the corresponding solutions are outlined in an abstract form, as further details will be discussed in Chapter 4.

### 3.1.1 System requirements

To understand the logic behind the design of PhiShield's features and components, the system requirements describe the platform's capabilities. These are divided into functional requirements, which specify the features that the project should provide and quality requirements, which describe how these features should be delivered and the standards they should meet.

#### Functional requirements

- **Automation**
  *Definition*: Strong automation mechanisms avoiding unnecessary user's intervention.

*Design decision*: Automated procedures have been integrated to support the normal operation of the application, particularly in areas such as email delivery, target behaviour reporting and learning engagement. A scheduled task manages the planned emails for a campaigns and delegates the actual sending process to an asynchronous worker, ensuring that resource-intensive operations are handled outside the main application flow. In addition, user interactions are automatically collected once a campaign is completed and targets who perform poorly are directly enrolled in specific learning paths.

- **Full customisation**
  *Definition*: Customisation capabilities for both training and learning programs.
  *Design decision*: Phishing campaigns can be designed by defining various parameters, including the landing page, which serves as the destination that the target accesses upon clicking the phishing link. Additionally, the email template is specified by creating the message content, attachments included, to be delivered to the recipient and the spoofed sender address used to impersonate a legitimate source. Furthermore, campaigns can be customized to target specific groups of recipients, such as managers, allowing for a more tailored approach. PhiShield also enables the integration of recipients' personal data within the email template, increasing the realism of the whole campaign. On the other end, learning paths can be adapted according to specific requirements, with constraints limited to the lesson file format, which must be PDF and the format of the examination questions, which are restricted to True/False ones.

- **Controlled campaign dispatch**
  *Definition*: Email distribution within the provided time range.
  *Design decision*: Phishing emails, after a new campaign is created and configured, are scheduled for being spread throughout the whole range specified. The scheduling unit selected is minutes, as most email service providers do not support finer-grained intervals such as seconds. If the number of email to be sent is lower than the number of available minutes within the defined interval, one message per minute will be sent, otherwise the system delivers them in minute-based batches.
  This scheduling approach not only fulfils the functional requirements of the system but also contributes to reducing the likelihood of triggering antispam alerts. Indeed, dispatching a large volume of emails within a very short time window increases the probability of detection by spam filters, while this solution aims to lower this risk.

- **Tracking**
  *Definition*: Reliable tracking procedures to analyse targets interaction with phishing simulations.
  *Design decision*: To monitor user behaviour towards phishing campaigns, a URL-based tracking mechanism is implemented. This system appends a unique identifier to each phishing URL, enabling the monitoring of every interaction performed by the targeted user. Specifically, the system can record three key campaign events: "email sent", which logs the successful delivery of

the message; "link clicked", which is triggered when the recipient accesses the phishing URL and "data submitted", which captures any information entered on the phishing website.
To facilitate this process, all landing pages are hosted directly by the PhiShield server, ensuring a consistent tracking procedure, allowing administrators to critically analyse user interactions throughout the campaign. It also enables a deep assessment not only of the campaign's effectiveness and its individual components, but also of the overall impact of the awareness program.

- **Reporting**
  *Definition*: Built-in reporting systems for statistical breakdown of awareness programs' performance.
  *Design decision*: For both the training and learning modules, informative dashboards are developed to enable managers and other non-technical stakeholders to assess program effectiveness and user performance. These dashboards provide an accessible overview of key performance indicators, specifically: within the training module, the focus points to the overall performance of the targeted users, complemented by more granular metrics such as campaign-specific results. In the learning module, the reporting underlines lesson activities and corresponding exam results. All reports can be exported in different formats, facilitating further analysis, and data sharing across different organizational contexts ensuring that both modules can be continuously monitored and analysed in real time, supporting evidence-based evaluation of user engagement and program success.

### Quality requirements

This section describes the main quality attributes considered in the design. While these attributes are not directly tied to specific functionalities, they still represent important properties that the system must hold to ensure a successful outcome for this academic project.

- **Security**
  Security is a fundament of PhiShield, given its responsibility for dealing with sensitive information and simulating real attack scenarios. The application must secure both user data and operational integrity from unauthorized access or manipulation. To this end, several security mechanisms can be considered, including user authentication, role-based access control and data encryption, particularly for sensitive information submitted by users through phishing simulation pages. Furthermore, security and privacy considerations extend to the management of campaign results, as improper handling of these data could compromise user confidentiality by revealing individual behavioural responses to phishing attempts and exposing the target himself to real attacks.

- **Usability**
  Usability is a key attribute to ensure that the application can be effectively operated by non-technical users, such as managers, who need to clearly interpret campaign results, user performance metrics and examination outcomes,

hence easily tracking the overall progress of the awareness program. A clear interface, intuitive navigation and meaningful reporting features are therefore essential in this context. Usability considerations also extend to PhiShield operators, which are the system administrators, who must be able to efficiently monitor not only campaign, lesson and user performance data, but also the overall system status, including access to event logs and operational activity, enabling comprehensive monitoring of PhiShield's internal processes.

- **Scalability**
  This aspect is crucial for the overall success of the project. PhiShield must support highly scalable processes, particularly when managing the dispatch of multiple campaigns and handling increasing volumes of user data, without compromising the performance of either the awareness programs or the whole system. To achieve this, robust mechanisms for asynchronous background task execution have been implemented and blended with automation strategies as described previously. These solutions are designed to offload intensive operations from the main application process, ensuring efficient resource utilization and maintaining system responsiveness.

- **Reliability**
  Reliability is a fundamental requirement for PhiShield, ensuring that the system can continuously operate. Mechanisms such as error logging and retry policies are implemented to facilitate recovery from unexpected issues, including failed email deliveries or temporary network issues. This property is particularly critical also in the context of reporting and result collection, as the accuracy of phishing campaign data depends directly on system reliability, ensuring that campaign outcomes are correctly recorded and can be visualized in their dedicated dashboards.

- **Portability**
  Given that security awareness represents the core focus of this project and its intended users, PhiShield must ensure high portability, supporting multiplatform compatibility and provide straightforward setup and deployment procedures, enabling users to adopt the solution with minimal effort. This approach ensures that organizations can quickly initiate their awareness programs without expending unnecessary time or resources on installation and configuration, thereby boosting accessibility and widespread usage of the platform.

### 3.1.2 User requirements

To conclude this requirements overview, the focus now shifts from a project-oriented perspective to that of the end user. This section wants to outline the objectives that users are expected to achieve through the system, complementing the functional specifications with a representation of the intended user experience. Therefore, the following discussion should be considered as a corollary to the preceding sections, offering a user-centred interpretation of the system's design.

In this context, three different categories of user have been identified: cybersecurity operators, management personnel and normal users.

- **Cybersecurity operators**
  This category refers to security specialists responsible for managing cybersecurity aspects within an organization. Their main objective is to design and monitor customized phishing simulations aimed at assessing the success likelihood of selected users and enhancing their ability to respond appropriately to such attack vectors. To effectively address these needs, the system must provide advanced automation, extensive customization and robust tracking and reporting functionalities. These features can ensure that cybersecurity professionals can conduct realistic and controlled simulations while obtaining accurate metrics of user performances.

- **Management**
  In cybersecurity context, managers and human resources personnel are responsible for evaluating employees' awareness levels and following their progress over time, therefore, their primary concern lies in monitoring users' outcomes rather than focusing on technical characteristics of campaigns and lessons. For these users, in fact, real-time dashboards and efficient reporting systems with export capabilities are essential, as they enable rapid assessment of both campaign and learning module results.

- **Normal user**
  The normal user category includes all the targets of the awareness programs within both the training and learning modules. Although their requirements are of lower priority compared to those of administrators or cybersecurity specialists, they are still essential for the overall success of the system. Normal users should be able to access a comprehensive overview of their progress across modules, including exam results and training performance, with the option to examine the details for specific a campaign or view aggregated statistics. Moreover, it's important to provide users with clear and immediate feedback on their inclusion in a phishing simulation. The feedback should explicitly highlight the aspects of the campaign that should have been critically evaluated to avoid falling into the "trap". PhiShield achieves this goal through the use of a dedicated redirect page, which is the final webpage to which the target is redirected after interacting with any element of the campaign. This page serves both an educational and corrective purpose, reinforcing awareness by illustrating the indicators of phishing that were exploited during the exercise.
  These functionalities not only supports the wider objective of enhancing user awareness but also provides a form of personal validation to the targets, serving as a certification of their completion of a specific awareness program along with the achievement of concrete results.

## 3.2 LLM integration

As discussed earlier, artificial intelligence can have a significant impact within the context of security awareness, supporting the user by creating interactive and customised experiences. To provide a concrete example of this powerful aspect, a minimal integration, leveraging a Large Language Model (LLM), has been included in the project. Although this functionality was not a fundamental system requirement, it was introduced to prove how artificial intelligence could be employed to facilitate learning and enhance the overall user experience. Specifically, it allows users to interact with the system by requesting clarifications on specific topics and receiving immediate and intuitive suggestions, while maintaining the focus on the primary objective of the platform: raising security awareness.
This design choice opens the door to the use of LLMs, concretely demonstrating one of the different potential application of artificial intelligence in this domain, serving as a basis for possible future developments of the project.

## 3.3 Conceptual data flow

To conclude this chapter, a simple, abstract data flow diagram is presented to visually illustrate the rationale underlying the design of the project. This diagram is abstract, by definition, as its purpose is to provide an high-level overview of the main data flows within PhiShield rather than a concrete technical aspects. Implementation details and specific process interactions will be discussed in the subsequent chapter.

Figure 3.1 illustrates the principal processes and actors involved in the main operational routine of the application. As described in the requirements section, the identified user categories of this platform are: security specialists, managers and human resources personnel and normal users, who represent the targets of the awareness programs. Each user class has a distinct influence on both the data flow and the overall functioning of the system. Accordingly, their participation can be characterized as active, passive or auditing, depending on the extent of their interaction with the application:

- **Security specialists** are considered active users, as they are responsible for creating and managing content across both modules. Specifically, they configure and submit new phishing campaigns, including all required components and handle the development of lessons and exams materials within the platform.

- **Management personnel** can be considered as PhiShield auditors. They do not perform any direct operations on the platform, as they only monitor the progress of each target participating in the awareness programs.

- **Normal users** referred to as "targets" within the platform, they are classified as passive users as all their actions are initiated by internal or external triggers or actors. In practice, they participate in awareness programs designed and

managed by security specialists. The only activity that could be considered a deviation from this passive role is their interaction with phishing emails, however, even this behaviour initiates from a broader process.

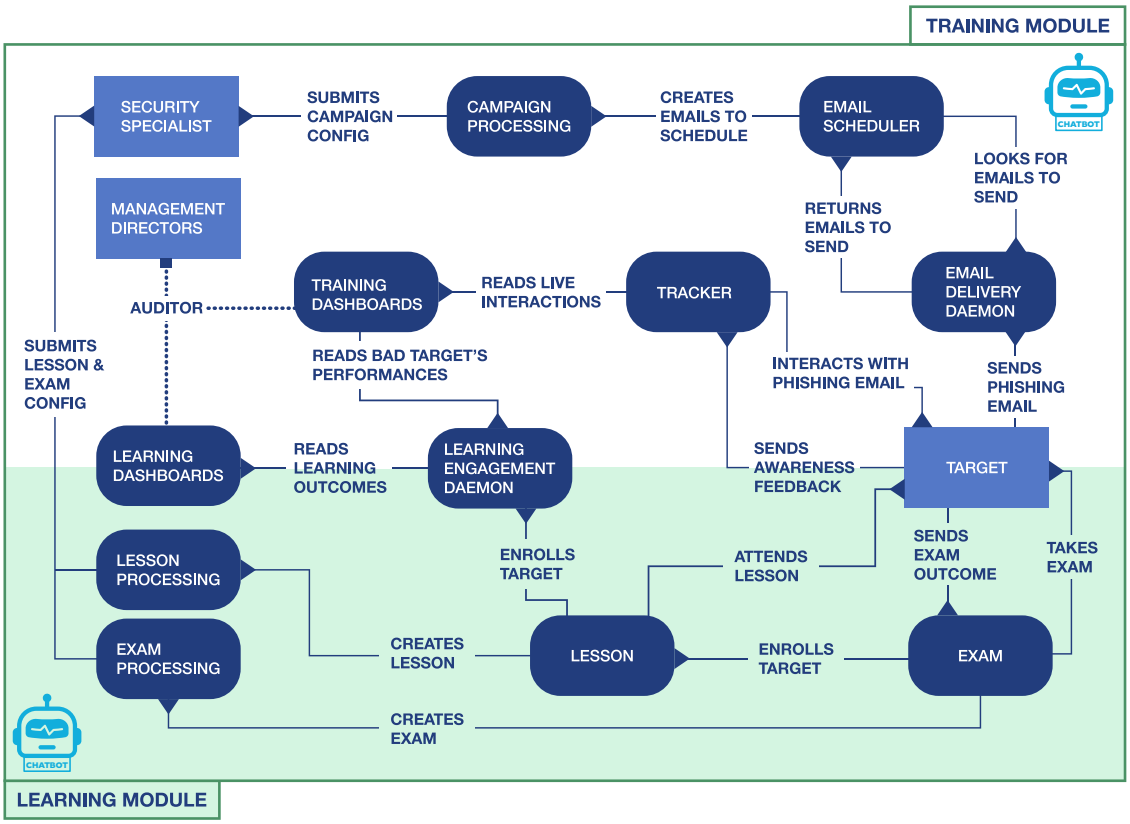


Figure 3.1. Data flow diagram

By examining the figure, the important contribution of the security specialist, being the initiator of the process, is confirmed. In fact, this actor is in charge of conducting and managing the awareness program as a whole, starting from submitting campaign's data to the system. Obviously, this action has an implicit pre-requisite: creating all the campaign components on the platform, like the landing page (i.e. phishing webpage), the redirect page (i.e. target's feedback webpage), the email sender (i.e. real profile used to send emails) and the email template (i.e. subject and body of the phishing email). Additionally, lessons and exams needs to be set up before launching a new campaign, which again requires the involvement of the security specialist as supervisor of their content.
By shifting the focus to the target, the passive role of this user category within the platform becomes evident and aligns with the requirements previously defined. The primary interest of the target is receiving training and learning content, enforced by feedbacks throughout the campaign. This process aims to enhance their understanding of cybersecurity principles and strengthen their attitude in recognizing and appropriately respond to real phishing threats.
Finally, human resources officers, managers, and executive directors are considered

supervisors of the overall process, as their interaction with the system is limited to the consultation of statistical dashboards. These dashboards facilitate the critical analysis and exploration of both aggregate progress metrics and more specific indicators, such as individual target or campaign performance, as well as learning outcomes. This enables a concrete evaluation of the security awareness programs and their overall effectiveness.

# Chapter 4

# Implementation

This chapter presents the implementation of this project, describing how design and requirements defined in the previous sections have been fulfilled, creating a working system. The objective is to provide an overview of architectural and technical solutions adopted during development, complemented with the description of the main components and their primary interactions. Moreover, the critical aspects and issues identified during this phase will be discussed, with an analysis of both the encountered obstacles and the solutions implemented for the project. Finally, a general overview of the expected usage of PhiShield is provided.
In the appendices, comprehensive user and developer documentation is included to facilitate a deeper understanding of the project. This material supports system maintainability and enables potential future extensions.

## 4.1 Technological Stack

The implementation of PhiShield was guided by the simplicity principle: the project was developed with the aim of producing an effective and functional tool. As this project is part of an academic context, the focus was mainly on demonstrating the technical usability for security awareness programs, while keeping the overall architecture lightweight and easily deployable. The technological stack reflects this philosophy, in fact, each component was selected to reduce unnecessary complexity, while preserving the fulfilment of the previously described requirements. Additionally, PhiShield was implemented to be easily extensible with future enhancements, as described in the last chapter of this paperwork.

### 4.1.1 Python 3

Even though PhiShield can be considered as a web application, Python 3 [15] is the backbone of this project as it's the only programming language used. It was chosen for its wide ecosystem of standard libraries and third-party packages available, allowing low effort integrations within different internal components and functionalities. Furthermore, Python is natively bounded with one of the major web frameworks available on the web: Django [16], written in Python itself, it's been used as the foundation of PhiShield's frontend and backend implementation.

### 4.1.2 Django

Django [16] is a free and open source Python web framework based on the Model-View-Template (MVT) design pattern, which enforces a clear separation between three main layers:

- **Model**: the underlying data model that interacts with the database engine.
- **View**: the business logic, which controls the data flow between the database and the client.
- **Template**: the presentation layer, i.e. the frontend of the web application.

This structure allows developers to build secure and scalable web applications. Django provides a wide range of built-in functionalities, including user authentication, automatic administrative interface and database-driven development supported by its native Object-Relational Mapping (ORM) system. Furthermore, it promotes modular development through its app-based architecture, allowing projects to be organized into independent components that can be easily integrated at both the data and business logic levels.
Within this project, Django is responsible for managing several critical aspects:

- **Persistence**: through Django's ORM system, PhiShield models are defined as simple Python classes which are automatically translated into database tables. This abstraction allows to work with high-level objects rather than writing raw SQL statements, increasing also the security of interactions with the database avoiding direct execution of SQL queries which can contain user-provided input and result into code injection vulnerabilities, such as SQL injection [17].
- **Modularity**: the app-based structure enables the project to be organized into stand-alone modules that can be independently developed and extended. Each application encapsulates a specific set of related functionalities, ensuring a clear separation of duties while still allowing effortless integration among different components of the system.
- **Security**: Django follows the principle of security by default, offering built-in protection against common web vulnerabilities such as Cross-Site Scripting (XSS) [18], Cross-Site Request Forgery (CSRF) [19], SQL injection [20] and clickjacking [21]. This approach prevents the developer from implementing security mitigations from scratch and rely on Django. Obviously, this framework provides just an essential set of best practices, which should be extended and enhanced by the developer, based on the nature of the web application being built.

These features strongly influenced the decision to adopt Django as the underlying web framework for the development of PhiShield. Its modular design enabled rapid application development while maintaining focus on the project's primary objective: the creation of a portable security awareness platform.

### 4.1.3 Celery and Redis

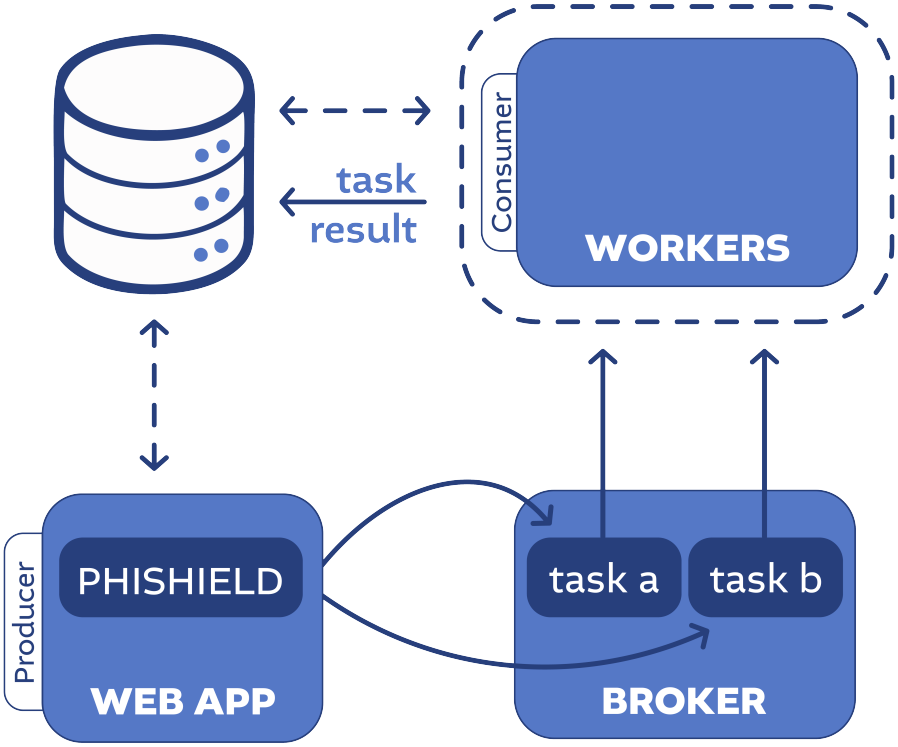


Figure 4.1. PhiShield interaction with Celery and Redis

To integrate and manage asynchronous and scheduled tasks, Celery has been chosen as a task queue system, with Redis acting as its message broker. Celery [22] is an open source, distributed task queue system which enables applications to offload time-consuming or resource-intensive operations so that these tasks can be executed outside the main process, resulting into continuous responsiveness of the application. It's based on a producer-consumer model where the main application, i.e. the producer, sends tasks to a message broker (Redis in this case), while one or more worker processes, i.e. the consumers, are triggered to execute them asynchronously. Redis [23], acronym for *Remote Dictionary Server*, is an in memory data structure store often used as a database, cache or message broker, which is the case of this project. Integrated with Celery, it acts as the message broker that manages the communications between PhiShield and the Celery workers.
In this context, Celery and Redis integrate seamlessly with Django framework thanks to dedicated Python packages [24] and provide robust background processing capabilities. PhiShield acts as the task producer and Celery tasks are defined within the application's modules. When these tasks are triggered, they are serialized and sent to Redis, that queues them for being processed. Celery workers, running as separate processes, continuously listen to the Redis queues, retrieving pending tasks, and executing them independently of the main server process. Moreover, with the help of *django-celery-beat* [25], which is a Django-based Python

package which improves the built-in schedule capabilities available in Celery by storing the schedule in the Django database and enabling dynamic interaction with the tasks, it's possible to define procedures that can be scheduled to be run based on specific time constraints, following different approaches, like:

- **Interval schedule**: to run tasks at fixed time intervals (e.g. every 20 minutes).
- **Crontab schedule**: follows cron-like rules (e.g. every day at 2 PM).
- **Solar schedule**: triggers tasks based on solar events (e.g. at sunrise or at sunset).

This integration allows the system to remain lightweight and responsive, even when performing resource-consuming operations.

### 4.1.4 SQLite

For data persistence, SQLite [17] was adopted as the database engine. Its file-based structure allows to run the application without external database configuration, simplifying both setup and portability. Moreover, SQLite is the default database engine supported by Django and although it's not designed for high-demanding environments, it offers fair performances in local or small deployments, which motivates its choice for this project. Additionally, Django's ORM system provides full compatibility with more robust and production-oriented databases such as PostgreSQL or MySQL, allowing an effortless migration, if needed, by just changing the database configuration in the project's internal settings.

### 4.1.5 Google Gemini

As outlined in the design chapter, a minimal integration of a Large Language Model has been implemented in this project, made possible through the use of Google's proprietary artificial intelligence model, Gemini [26]. Specifically, interactions are handled via Gemini's APIs, provided by installing the dedicated Python package, *google-genai*, which offers a fast and reliable LLM service with minimal setup, whose functionalities will be discussed later in this chapter.

### 4.1.6 Dependencies

To conclude the overview of the technological stack adopted for this project, a short list of external dependencies is presented, accompanied by a description of the specific functionalities and solutions that each package contributes to the project.

### Bootstrap 5

Bootstrap 5 [27] is a powerful frontend toolkit the provides built-in HTML and CSS components, easily embeddable via Content Delivery Network (CDN) systems.
In this project it's used to include its component to the frontend templates, enhancing the style and the appearance of the platform with low effort.

### Python-magic

Python-magic [28] is a Python package that implements an interface with the *libmagic* library, which is an operating system level library used for file type identification via file headers.
PhiShield uses this package to automatically detect the type of files uploaded by the user on the platform (e.g. email attachments and email template's body).

### Python-dotenv

Python-dotenv [29] is a Python package that allows to load environment variables stored in a specific file, usually named *.env*. This approach is useful to separate secret variables and configurations, such as API keys or authentication tokens, from the application code, boosting the overall security and portability of the project.

### Django-cleanup

Django-cleanup [30] is a Python package implemented as a Django application that automatically manages file handling within a Django project's filesystem.
In this context, it facilitates the management of files uploaded by the user, such as email attachment files and stored in the database. When such files are deleted at the application level, Django-cleanup ensures that the corresponding files are also removed from the filesystem, maintaining consistency and preventing orphaned objects.

### Beautiful soup 4

Beautiful soup 4 [31] is a Python library that allows easy web scraping. It sits on top of HTML or XML parsers, providing Python functions and methods for iterating, searching and modifying the parsed tree.
In PhiShield, this library is employed to automatically parse HTML pages uploaded by users, both for email templates and landing pages, to insert the Django-specific template variables required for their correct functioning. In particular, it automatically adds the CSRF token, a security measure implemented by Django to prevent Cross-Site Request Forgery (CSRF) attacks, to every form detected within the uploaded page, allowing to correctly track any data submission performed by the target while interacting with the landing page.

### Pandas

Pandas [32] is a well known, fast and powerful Python library mainly employed for data analysis and manipulation tasks.
In this project, it's leveraged to easily manipulate CSV files uploaded by the user, which will contain the list of users, the targets, that should be enrolled in the security awareness program, forming a targets group.

### Django-fernet-encrypted-fields

This Python package is natively implemented for Django applications, and allows to automatically encrypt sensitive fields of database tables, using Fernet's algorithm, without altering the structure of their models. In PhiShield it's leveraged to protect and store the password attribute of the *EmailSender* model as well as any captured data from target's interactions with the phishing landing pages.

## 4.2 Architecture and main components

The overall architecture of the system is composed of four main layers:

- **Presentation Layer**: the user interface with the application, implemented via Django views and templates.
- **Application Layer**: implements the business logic, through Django applications.
- **Data Layer**: ensures data persistence, using SQLite ad database engine.
- **Service Layer**: manages background and asynchronous tasks through Celery workers, using Redis message broker.

The external LLM component, implemented in this prototype as a chatbot, completes the overall structure of the project.

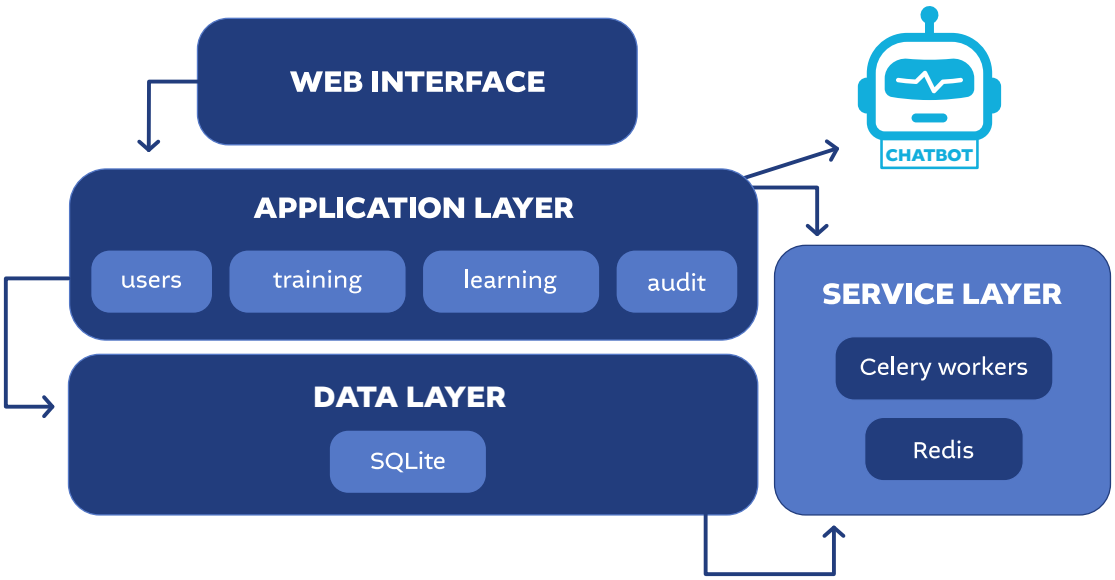


Figure 4.2. PhiShield architecture

The above figure, outlines the general PhiShield architecture, which reflects the four layers discussed earlier, which will be described in detail, explaining which are its main functionalities, components and what interactions it performs with the other ones, in order to provide a deeper understanding of the whole system.

### 4.2.1 Presentation layer

The presentation layer, by definition, is composed by the user interface and allows to perform every expected interaction with the system. This is made possible mainly by two Django built-in components:

- **URL dispatcher** is responsible for managing HTTP requests and performing routing operations, based on the demanded resource. Through the URLconf [33] file, it allows the developer to create custom routes and associate them with specific views, implicitly declaring routing constraints within the app. The URLconf is a Python file that contains the definition of every url-pattern that the URL dispatcher should try to match when an HTTP request is performed on the application.
  For this project, a clean URL scheme has been designed, trying to adhere to common best practices and naming conventions for URL path routes, creating a specific sub-path for each model declared in the database.
  For example, for the *LandingPage* model, the following routes are available:

- `/landing_pages/`: retrieves the list of the landing pages available.
- `/landing_pages/create/`: triggers the creation of a new landing page on the database.
- `/landing_pages/<int:pk>/`: retrieves the specific landing page that matches the specified parameter "pk". Django provides a useful set of path converters [34], `int` in this case matches zero or any positive integer, to introduce dynamic parameters in the url pattern. Moreover, *pk* is a Django default variable, that automatically maps to the primary key of the of the object's model.
  In short, by specifying such path converter, Django implicitly looks for a landing page in the database that matches the provided primary key.
- `/landing_pages/<int:pk>/update/`: triggers the update of a specific landing page on the database.
- `/landing_pages/<int:pk>/delete/`: triggers the deletion of a specific landing page on the database.

Additionally, Django URL dispatcher allows to create custom aliases for the defined url patterns, which will be associated to the Django app they belong. This aspect is extremely advantageous if used in templates. To give an example, if in the HTML template for landing pages, there is a delete button that, if clicked, should navigate to the delete path previously specified, instead of providing the path of the full url to visit, it's possible to just use the alias specified in the URLconf file, `<app_name>:deleteLandingPage` in this case, where *app_name* is the name of the Django app (i.e. the namespace).

- **Templates** are the standard method for creating the HTML pages. Being a web framework, Django proposes a convenient way to generate dynamic HTML code, in fact, a template file contains the static snippets of code of the desired HTML output as well as Django-specific syntax describing how dynamic content will be inserted and rendered at runtime in the frontend. Django has developed its native templating markup language, which is composed of different utilities:
  - **Variables**, which are passed to the template through the context dictionary. This object, is managed and configured as desired in the views and then is passed to the template as an argument, providing values to variables that are statically specified in the HTML code using a special syntax. At runtime, the static variables are replaced with the actual value provided in the context dictionary. For example, if the template contains the following line of code,`<p> Hello {{username}}! </p>` and the value of the username variable in the context is "John", the rendered page will contain `<p> Hello John! </p>`.
  - **Tags and Filters** will provide custom logic to the template. Although Django discourages introducing heavy-computational logic on templates side, which should be implemented in the views, it offers a wide set of, simple, default tags, mainly useful for performing loop operations or minimal modifications on context variables and filters, usually involved

to manipulate context variables content.
For example it's possible to iterate over a list of context objects with the for-loop tag and perform conditional operations:

```
# iterate over list context variable
{% for element in list %}

  # filter to transform a string in Title Case
  # 'my name is John' becomes 'My Name Is John'
  {{ element.name|title }}

  # conditional display of HTML code
  {% if element.has_some_property %}
    <p> Yes! </p>
  {% else %}
    <p> No </p>

{% endfor %}
```

Django by default, requires a standard filesystem structure for directories and files containing templates to be rendered, which can be eventually customised by adjusting specific variables in the settings configurations. For this project, the default structure has been adopted, which consists of having a general *templates* directory at project level, which contains generic templates and creating a specific *templates* folder for each Django app:

```
project_root/
-- project_folder/
--- templates/generic_template.html
-- app_folder/
--- templates/specific_app_template.html
```

Another valuable feature of Django templating system is the ability to encapsulate partial templates within other HTML files. This approach prevents the excessive concentration of code within a single file and inherently supports developers in writing cleaner, modular and reusable code. This characteristic results particularly useful while dealing with HTML pages that have multiple occurrences of the same component, which can be extracted and defined into another file, that will be included into the main one using the *include* Django tag, as follows:

```
# _stat.html partial template
<div class="some CSS classes">
 <h2>{{ statistic_label }}</h2>
 <p>{{ statistic_value }}</p>
</div>
```

```
# reports.html main template
<h1> User statistics </h1>
{% include "partials/_stat.html" with
   statistic_value=report.value1 %}
{% include "partials/_stat.html" with
   statistic_value=report.value2 %}
{% include "partials/_stat.html" with
   statistic_value=report.value3 only %}
...
```

It's important to highlight that the context of the main template is automatically passed to any included partial template, along with additional variables specified using the *with* keyword within the include tag. Alternatively, it's possible to provide a partial template with a minimal context, by appending the *only* keyword to the include statement, thereby limiting the variables available within that specific scope, as shown in the above snippet of code.

Frontend can be considered the user's access point for web applications, indeed it represents the starting point in the general flow of requests received by PhiShield, which usually follows the steps depicted in Figure 4.3 below.

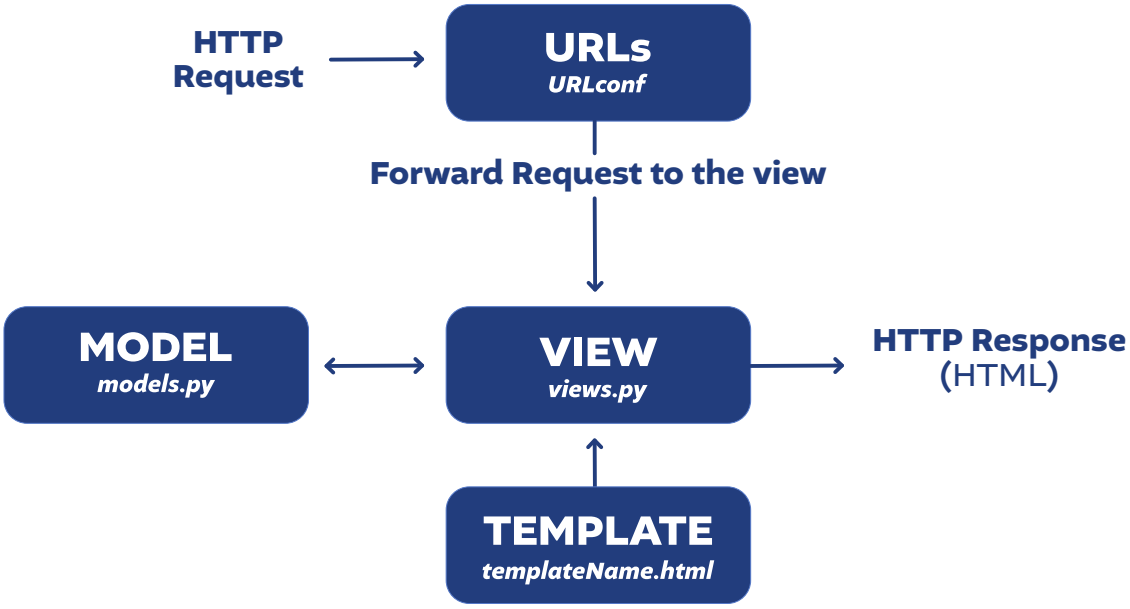


Figure 4.3. HTTP request flow

### 4.2.2 Application layer

The application layer is the most critical of the system, as it is responsible for managing the overall business logic. As previously discussed, this project is organized into multiple Django applications, each responsible for a specific functional domain, promoting modularity and a clear separation of duties within the architecture. The logic of each application is defined through so called Django views,

which are the intermediaries between the frontend, composed of templates and the database models.
PhiShield is structured into four different Django apps, where two of them, *training* and *learning* modules represent the core of the application, while other two, *users* and *audit* implement specific functionalities.

### Training app

As the name suggests, this Django app is in charge of every operation related to training activities, i.e. phishing campaigns, that have the main objective to simulate real attack scenarios and strengthen the ability of the targeted users to detect and to face such critical threats, leading toward an improved security posture. Specifically, this module manages the various components of a phishing campaign and provides a reporting dashboard that visualises the progress of each campaign. The dashboard also enables to perform detailed analyses by examining campaign-specific or target-specific performance metrics, thereby offering both an overall and granular view of the program's effectiveness.
The main activities that this application allows to perform, based on the authenticated user, are several:

- **Campaign components management**: as previously clarified, a campaign is composed of different elements, which can be managed in this module, where it's possible to perform create, update or delete actions. A campaign consists of:
  - **Difficulty**: indicates the level of sophistication of the whole campaign, it can assume three possible values: *easy*, *medium* and *hard*, which are selected based on the operator's discretion.
  - **Targets group**: the list of targets that will be involved in the simulation.
  - **Landing page**: the phishing page to show if the target clicks on the phishing link.
  - **Email template**: the structure of the phishing email to be delivered.
  - **Attachment**: optionally, the operator can include attachments to the email template, to make it more realistic
  - **Email sender**: the profile that contains the SMTP configuration that will be used to deliver phishing emails. The correctness of the information provided for this component are extremely critical for the success of the whole campaign.

  These permissions are granted only to the platform operators, i.e. administrators.
- **Campaign management**: upon the creation of a new campaign, the console allows operator users to have an overview of the whole set of campaigns existing as well as the ability to execute different actions on them, based on the campaign status, which can be:

  - **SCHEDULED**: campaign created, but no email have been sent yet.
  - **IN PROGRESS**: campaign's email are delivered to targets.
  - **STOPPED**: campaign has been manually paused, by the operator.
  - **COMPLETED**: every campaign's email has been sent.
  - **COMPLETED WITH FAILURES**: campaign completed, but at least one email had issues for its delivery.

- **Dashboard**: The results of training programs can be monitored, in real time, through dedicated scoreboards which offer overall statistics.

Another essential functionality implemented by the training app, which is not directly available on the console as it operates in background, is user interaction tracking. This feature enables the system to record the actions performed by each target during a specific phishing campaign. To achieve this, PhiShield has been designed to directly host the landing pages as regular web pages, allowing the platform to log every interaction executed by the user. Specifically, each instance of a campaign associated with a given target using a unique string, i.e. the tracker, represented by the *TrainingResult* model, which will be discussed later, can assume one of four possible statuses:

- **EMAIL SCHEDULED**: email programmed to be sent to the target.
- **EMAIL SENT**: email correctly sent to the target.
- **LINK CLICKED**: target interaction with the phishing email, by clicking on the phishing link, which points to the landing page specified for the campaign.
- **DATA SUBMITTED**: worst scenario, target visit to the phishing web page and data submission onto it, for example by providing credentials to a fake login form.
- **DELIVERY FAILED**: the system wasn't able to deliver the specific email.

Additionally, a remarkable implementation in this context is the email templating tool, that leverages Django's templating mechanism previously outlined to enhance the customisation capabilities of the platform. Specifically, this mechanism allows to provide a specified set of variables to the HTML body of the phishing email, which will be evaluated, through the context object before the sending procedure, displaying the target's information. At this stage, the following variables are available:

- **First name**: the target's first name.
- **Last name**: the target's last name.
- **Email**: the target's email address.
- **Job position**: the target's job position, within the company.

- **Phishing url**: the URL that points to the phishing web page.

Each variable can be included in the email template file, to be uploaded on the platform, by following Django's variables convention: `{{ <VARIABLE_NAME> }}`. To give an example, if the system is preparing to deliver a phishing email about password change to John Smith, an IT administrator, the template shown below could be crafted:

```
Hello {{FIRSTNAME}},

The security team has detected a suspicious access
to the {{EMAIL}} address associated with your person.
An URGENT action is required!
You MUST perform a password change at this link:
<a href="{{PHISHING_URL}}">Password portal</a>

Regards,
The security team
```

### Learning app

The second most significant application is the learning module, which is responsible for managing the educational content intended for employees. Specifically, it enables the administration of lessons and exams, which are the methods for delivering cybersecurity-related material to users. The main activities performed by this module include:

- **Lesson management**: allows to create and manage lesson's content within the platform. The lesson should be uploaded as a PDF file.

- **Exam management**: exams can be created and modified as required. Each exam consists of an arbitrary number of questions, which may be defined during the initial creation process or subsequently added at a later stage. The only question type allowed at the moment is "True/False".

- **Dashboard**: the progresses of the targets engaged in specific lessons, as well as their exam outcome is accessible into a dedicated dashboard.

Similarly to phishing campaigns within the training module, user activities related to learning purposes are monitored through an automated mechanism. For simplicity and efficiency, when a user submits the answers of an exam, the individual responses are not stored in the database. Instead, an evaluation is directly performed by calculating the final score, which represents the sole outcome associated with that specific exam attempt.

### Users app

Although this application is responsible for a relatively limited set of tasks, mainly concerning the authentication of PhiShield users, it nonetheless represents an important component for the overall security of the project. In fact, this module implements the core operations that every PhiShield user, regardless of their role, is authorized to perform, such as login and logout procedures.

### Audit app

To enhance monitoring and reliability of the web app, the audit module has been implemented to introduce a dedicated logging layer over the entire platform. Moreover, every PhiShield's operational event is logged and made accessible through a dedicated visual log register, available exclusively to users with administrative privileges(i.e. operators). The log register will be thereby populated with different type of events, which can be grouped into the following categories:

- **Training log**: training related events, such as campaign launch or target interaction.
- **Learning log**: learning related events, such as lesson creation or exam outcome registration.
- **Access log**: authentication events, such as user login or logout.
- **System log**: system related events, such as email schedule creation or email delivery.

### 4.2.3 Data layer

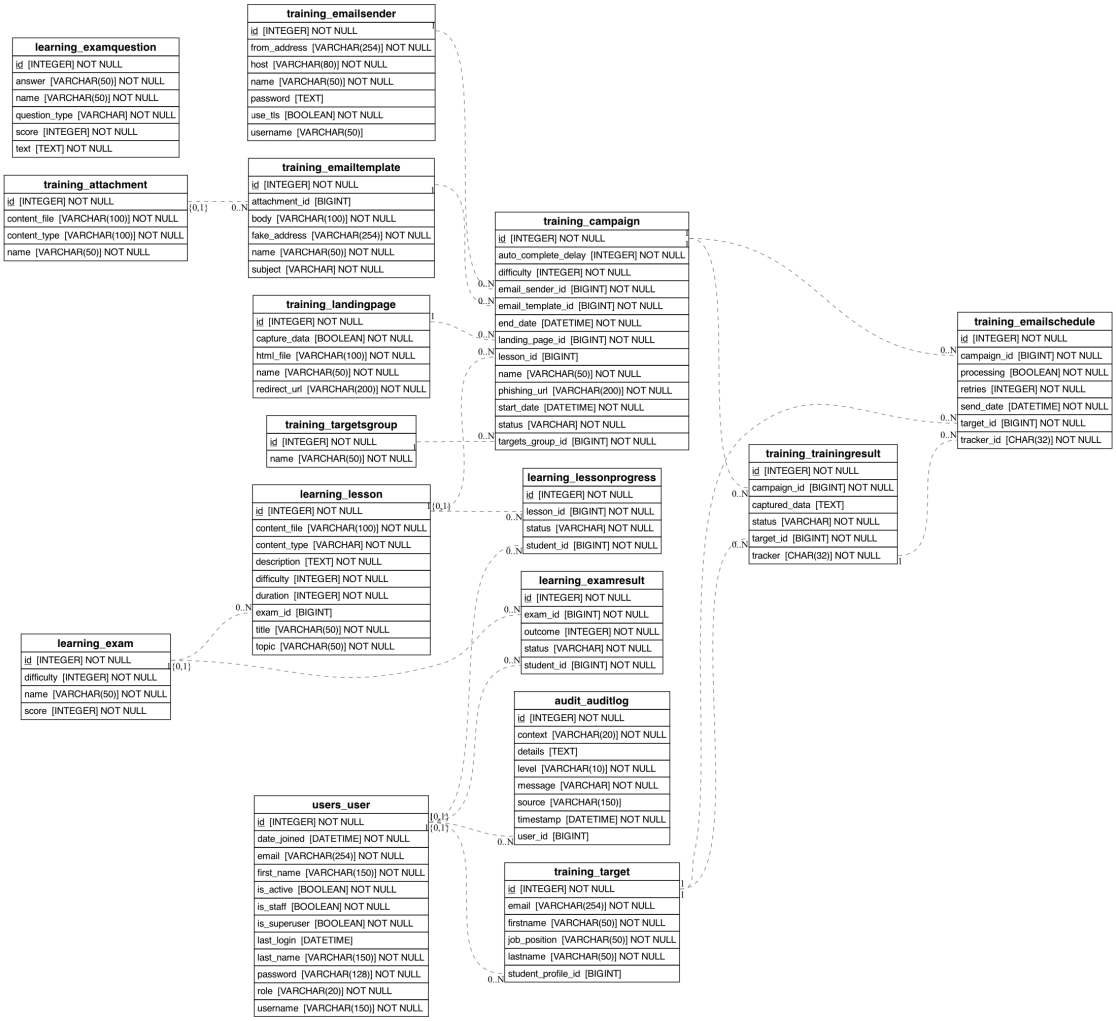


Figure 4.4. PhiShield ER diagram

In this project, data persistence is completely implemented with Django's built-in ORM system, which enables the definition of customized data structures by simply creating Python classes. This module natively provides a extended set of properties and methods for defining database tables and their corresponding fields. Furthermore, its database migration mechanism, facilitates the propagation of model modifications across all applications within the project. Each migration is automatically tracked by a auto-generated Python script, which records every change applied to the models and also enables rollback operations when necessary, thereby supporting consistency, traceability and maintainability of the database architecture.
The diagram in Figure 4.4 shows the entire database schema implemented in this project, which is composed by the following models:

- **Training app models**

  - **Target**: the users that will be engaged in awareness programs.
  - **TargetsGroup**: an aggregation of targets, to easily provide the campaign the list of targets to involve.
  - **LandingPage**: the phishing web page. This model requires to provide an HTML file that contains the structure of the page, it allows to specify whether to collect or not any submitted data by the target, by means of the *capture_credentials* field. Moreover, the *redirect_url* field, is used to store the web page where the target should be headed to, after any interaction.
  - **EmailTemplate**: the phishing email. This model, similarly to landing pages, requires an HTML file to be provided as the body of the email message. The *fake_address* specifies the address to be used as the fictitious sender of the message.
  - **Attachment**: an *EmailTemplate* can be provided with an attachment file, to create a more realistic scenario.
  - **EmailSender**: the SMTP profile used to send phishing emails. This sender is used to dispatch campaign's emails, by using the SMTP server specified in the *host* field, along with its credentials, if needed. The *from_address* field, should contain an existing and valid email address, which represents the actual sender.
  - **Campaign**: the main model of the training app, encapsulates every characteristics of the campaign to be dispatched.
  - **EmailSchedule**: support table, it's used to store the schedule of the emails to be sent for each existing campaign. This model serves as the entry point for the scheduled Celery task.
  - **TrainingResult**: the progress of a specific target within a given campaign.

- **Learning app models**
  - **ExamQuestion**: a question to include into an exam. The sole type allowed, so far, is "True/False".
  - **Exam**: an aggregation of *ExamQuestions*, which forms a complete exam, to test target knowledge against specific topics.
  - **Lesson**: the awareness material is provided through lessons, which requires to upload a PDF file, stored in the *content_file* field and specify the main subject by means of the *topic* field.
  - **LessonProgress**: support table, it's used to trace the progress of a target user, over a specific lesson.
  - **ExamResult**: support table, it's used to trace the progress of a target user, over a specific exam. Upon exam completion, the final mark is stored into the *outcome* field.

- **Users app models**

  - **User**: the data structure that encapsulates a PhiShield user.

- **Audit app models**

  - **AuditLog**: events that occur during the execution of PhiShield processes.

For the authentication mechanism, implemented through the users app, three roles have been defined to align with the user categories previously outlined in Chapter 3 during the definition of user requirements.

- **Operator**: the administrator of the platform. It is provided with the complete set of permissions and can access every section of the project.

- **Auditor**: read-only user that is provided with limited permissions, mainly restricted to the training and learning report dashboards.

- **Student**: this role is granted to *Target* users that are automatically engaged in any learning activity. The provided set of permissions allows to visit only student-related sections of the application as overviews of campaigns, lessons and exams where the user is involved.

This structure enforces a strict separation of concerns by implementing a simplified Role-Based Access Control (RBAC) model, ensuring that each user is granted access solely to the components of the application corresponding to their assigned role and permissions.

### 4.2.4 Service layer

This layer, as Figure 4.2 shows, interacts with both the data and application layers, defining and managing all tasks that are executed outside the main process of the application. Through the integration with Celery and Redis, the following task have been implemented:

- **Email schedule check**: a recurrent task, scheduled using the Crontab schedule mechanism natively available in Celery, thanks to *celery-beat* module. It's programmed to run every minute and retrieve from the database the list of emails that needs to be sent, meaning that the *send_date* field of every *EmailSchedule* instance fetched is lower or equal to the current time. Subsequently, it adjusts the status of the involved campaign and delegates the sending operation to an asynchronous task, sketched in the next point.

- **Email delivery**: an asynchronous task, defined to perform the sending operation for a specific email, within a given campaign. It additionally updates the status of the *TrainingResult* record related to the target, i.e. the receiver of the message.

- **Automated learning engagement**: asynchronous task implemented to automatically enrol targets who performed poorly in the training activities into a predefined learning path, which could consist of both lesson and exam or exclusively of the lesson part. Specifically, this task is triggered when a campaign is marked as *Completed* and it retrieves the set of *TargetResult* instances that have recorded an activity with status as either *Link clicked* or *Data submitted*. Subsequently, the corresponding users are automatically enrolled in the designated lesson, their student profile is created, if not present yet and it's linked to the *Target* record using a dedicated field.

### 4.2.5 Chatbot component

For this prototype version of PhiShield, the LLM integration has been implemented as a chatbot, functioning as an independent component accessible via a dedicated endpoint. From an architectural perspective, it could be considered part of the service layer, as it supports cross-functional interaction for both operators and students, who are the only two roles allowed to engage with this component. However, since its logic relies on external API services, this chatbot can be conceptually situated within the external service layer, that supports and enhances the core functionalities of the application.
The interaction with the chatbot has been designed to inherit an initial context, based on the user role. In particular, depending on the user, the LLM is initially prompted with a different set of contextual information, enabling it to provide the user with tailored and relevant responses. Specifically, it supports the operator users for the creation of realistic and effective phishing scenarios while acts as a supervisor for the creation of learning material and provides suggestions for the general exam structure. Furthermore, the chatbot can be leveraged by students to obtain clarifications about particular learning topics, supporting them throughout the whole learning process and enhancing their preparation for the final examination. As can be deducted, this component is available in both the training and learning modules for operator users, whereas for students it's accessible exclusively within the learning environment.

## 4.3 Use cases

This section outlines the main functionalities currently implemented, presented from the perspective of the different user categories previously identified. Several use cases are proposed, with the aim to cover the whole set of features available at the current stage of the project.
As PhiShield is a security awareness platform, the focus of this section concerns the awareness programs that can be configured. The application has been designed to provide a high degree of flexibility in the creation of training and learning paths, enabling their adaptation to the knowledge and maturity level of the targeted users.

### 4.3.1 Training only

In this scenario, the user is exclusively involved in a phishing simulation, without any subsequent learning activities. Selecting this path can be particularly useful for conducting a preliminary assessment, aimed at evaluating the baseline cybersecurity awareness of the targeted users and identifying those who require more comprehensive training. This approach ensures that each user is evaluated under appropriate and controlled conditions.

### 4.3.2 Full training and basic learning

This use case involves a training campaign followed by a formative lesson, without the inclusion of any examination phase. Such scenario is very suitable for scenarios that require periodic reinforcement of cybersecurity knowledge, such as a brief recall of key concepts rather than an extensive program. Balancing between efficiency and quickness is extremely valuable in corporate environments where time availability is limited.

### 4.3.3 Complete awareness

As the name suggests, this program represents the most comprehensive configuration, encompassing both complete training and learning activities. The objective of this scenario is fully aligned with the primary aim of this project: to provide users with an end to end awareness lifecycle that enhances their ability to detect and respond to social engineering threats. Developing a complete program that initiates with practical training simulations and progresses through corrective learning activities, subsequently evaluated through dedicated examinations, helps building an environment of continuous assessment and improvement over time.

### 4.3.4 Secondary use cases

In addition to the primary use cases described above, mainly related to the configuration and management of training and learning programs conducted by security specialists, secondary use cases can also be identified, maintaining consistency with the full range of user categories previously defined for this platform. The key actors in these scenarios are end users, managerial staff and other executive figures. They are granted specific permissions that restrict access to designated sections of the platform, specifically:

- End users, i.e. the targets, can view their overall performances for phishing campaigns to which they have been assigned, complete lessons and consult related exam outcomes. They are also allowed to interact with the chatbot, within the learning context, to ask clarifications about lessons and improve their preparation for the final exam.

- Directors and other executive staff, is provided with view permissions limited to aggregated performances of both modules, allowing them to monitor their progress and the overall impact and effectiveness of the awareness activities.

The term "secondary", here, is used merely as a descriptive adjective, distinguishing these scenarios from the primary ones managed by cybersecurity personnel. From a functional standpoint, however, these use cases are essential, as they complete the lifecycle intended for this project.

## 4.4 Privacy support

From a technical standpoint, a delicate and meticulous approach must be adopted when handling data submitted by targets during training simulations. Within phishing campaigns, users are induced to enter and submit sensitive information on simulated phishing web pages. Such information is stored in the database and must be treated as highly sensitive and it cannot be processed in the same way as other user-provided inputs. To address this requirement, PhiShield allows the operator to decide whether to store any captured data or not by configuring a specific flag during the creation of the landing page. If the flag is selected, any data submitted by the target is stored into encrypted fields, thereby mitigating privacy-related risks and ensuring that all activities remain fully compliant with both security and educational purposes.

# Chapter 5

# Testing

To verify the correctness of PhiShield's functionalities and ensure that a reliable application has been implemented, several testing approaches were carried out. As discussed during the design phase, the process started from the development stage, where individual components were tested, and continued toward the evaluation of the application as a whole. Being a prototype, PhiShield was primarily tested to verify whether the expected operational lifecycle was effectively implemented, ensuring that the platform can serve as a security awareness platform and assist organisations to efficiently assess their employees' security awareness posture.

## 5.1 Testing scenario

This project has been tested during two distinct phases of its lifecycle: the first at development stage, where actions performed by both operators and target users were simulated; the second with a real-world scenario, involving actual participants.

### 5.1.1 Development phase

At this stage, a manual testing strategy was adopted, with the aim of simulating every possible actions that could be performed by the three designed user roles: operators, students and auditors. This phase was necessary to verify several key aspects of the application, including automation, data persistence and user interactions' tracking. As previously discussed, automation and the delegation of resource-intensive tasks represent critical goals for this project, which must be achieved without compromising reliability. Being an awareness platform that supports the creation of phishing simulations, email delivery represents one of the most critical services and must therefore be tested accurately.

#### MailHog

Email delivery has been tested using MailHog [35], an open-source tool that intercepts any email messages sent by the application under test, eliminating the need

to configure a real email delivery service. Its setup is straightforward, especially when using the official Docker container, which allows the service to be up and running in just a few steps. One of its most useful features is the web console natively available after installation, which simulates a standard email client and displays all intercepted messages using a simple interface, as shown in Figure 5.1 below. Moreover, it allows direct interaction with those messages, as in real scenarios, which makes it particularly suitable for testing the tracking capabilities of the project.

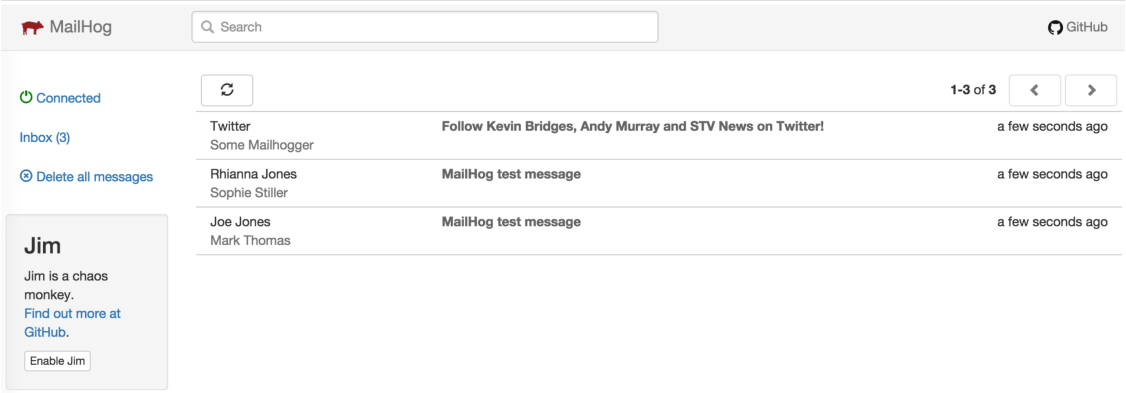


Figure 5.1. MailHog web console

This phase allowed to verify the correctness and robustness of the principal features designed and implemented for this project. In particular, it enabled the verification of each module of the application, assessing whether it performs as expected under different conditions, validating logical flow of operations. Moreover, this stage was fundamental to detect and fix possible inconsistencies or minor malfunctions before proceeding to more advanced and "real" testing scenarios.

### 5.1.2 Realistic scenario

After completing the development tests, the focus shifts to evaluating the overall functionality of the system, ensuring a smooth experience for PhiShield users. In this phase, aspects such as reliability and usability are assessed by creating a realistic environment, involving real participants. The scenario was designed to capture both the operator's and final user's perspectives, allowing for an assessment of the overall system's performance. In particular, the group of participants involved in this test was composed as follows:

- **Operator**: for this role, a technical user was selected to create both training and learning paths. For the purposes of this test, this user also assumed the role of *auditor*.
- **Targets**: an heterogeneous population of 30 users was involved, composed of individuals with varying levels of IT skills and representing different job roles: help desk technicians, finance professionals, human resources employees, project managers and IT specialists.

This scenario was structured in two sections. Initially, the operator was responsible for setting up the entire process, creating campaign components and uploading learning materials, exam included. The targets then participated in the campaign without being informed of their involvement, in order to ensure the most realistic scenario possible. Then, they were informed about the test and the ones who performed poorly were automatically engaged to complete the selected lesson and attempt the final exam. This phase was designed to assess the usability of the console for targets who, by this point, had automatically assumed the role of students. This scenario allowed to test multiple perspectives, ensuring that each user experienced the platform as intended. In particular, it made it possible to assess usability for both operators and students, represented by technical and non-technical individuals, thereby providing a various range of suggestions and feedback. Moreover, also the reliability of the system was verified, ensuring that automated processes, as email delivery and tracking, run smoothly throughout the whole process. Lastly, this test also enabled the collection of evaluations about the interaction with the chatbot component, assessing whether its responses were contextually appropriate and effectively supported the users during their activities.
**Note**: Since the targets were employees of a real company, all references to both the organisation and the individuals involved were redacted to preserve their privacy. For the purposes of this test, the company name and its associated domains were replaced with "Acme.org".

A comprehensive presentation of the artifacts used for this test, along with the detailed results obtained, is available in Appendix C.

### Training campaign context

For the purpose of this test, a specific phishing campaign was designed. Considering that the targets were employees of an Italian IT company, the chosen scenario focused on a recently introduced set of regulations issued by the Italian National Cybersecurity Agency, named *Agenzia per la Cybersicurezza Nazionale (ACN)*. Specifically, the regulation concerned an update to password policies, increasing the required length from 12 to 14 characters. Leveraging this new requirement, the campaign aimed to induce targets to change their company domain credentials through a phishing page replicating a typical web portal used for password changes. Based on its characteristics, the campaign has been categorised with *medium* difficulty. The operator used the application's chatbot to tailor this phishing campaign to the company context, guaranteeing consistency and realism.

### Email template

The template crafted for this test contained a brief explanation of the motivations behind the requested action. An examination of the template reveals two common patterns frequently employed in phishing emails: urgency and generality. For instance, the final sentence generates a sense of urgency by stating, in bold red text, that the user's account could be deactivated. Additionally, the template is written using generic sentences, with no references to the company context (e.g. sender signature) or to the supposed ACN regulation, despite the regulation being real.

### Landing page

As previously described, the landing page replicated a generic web portal that requested the submission of sensitive user information. Specifically, this campaign aimed to obtain company's domain credentials by asking users to provide the following:

- **Domain and username**
- **Current password**
- **New password**
- **New password confirmation**

This phishing page was designed to maximise realism. While a typical phishing form may request only username and password, additional information was prompted in this case, to enhance credibility. Specifically, the requested fields corresponded to those normally required to perform a password change within company's domain. The landing page was configured with the capture credentials flag set to false, meaning that the "data submitted" event is recorded for the campaign, while the actual submitted credentials are not, preserving the ability to track user interactions.

### Redirect page

The final stage of the campaign, from the target's perspective, was the redirection to a feedback page. This page informed the user of their participation in a phishing simulation and provided an explanation of the campaign elements that should have been critically analysed in order to avoid falling into the "trap".
In this case:

- **Sender address**: "Security team <sec-team@acme.org>" generic address, not related to the company, tries to emulate a possible address for the security team of the organisation.
- **Generic and urgent tone**: as previously explained, attackers commonly exploit this strategy to induce panic and agitation, leading victims to make rushed decisions;
- **URL**: "*https://login.ialaskfaadasf.it*" fictitious URL, not related to the company.

### Learning path

To complete the test, the operator created a concise yet structured learning path. As described previously, the learning module is composed of two core components: the lesson and the final exam. Also in this phase, the internal chatbot was employed to generate phishing-related educational material, ensuring that the content

was aligned with the company scenario used during the simulation. The lesson focused primarily on phishing, its main characteristics and the potential consequences that such attacks may have on an organisation, from financial loss to service disruption and reputational damage. The material included clear explanations, concrete examples and practical guidelines, aiming to reinforce users' ability to identify suspicious elements within emails and adopt safe behaviours when interacting with them.

#### Final exam

The final exam, composed of eight questions, was designed to evaluate whether the information presented in the lesson had been effectively acquired by the students. In particular, it aimed to assess the understanding of key concepts related to phishing and the ability to apply this knowledge in practical scenarios. The structure of the questions encouraged users not only to recall theoretical notions, but also to reason about concrete situations similar to those faced during the simulation.

## 5.2 Results

### 5.2.1 Training results

The phishing campaign designed for this test involved a total of thirty participants from different company departments. The primary objective of the activity was to observe users' behaviour and performance when exposed to a realistic scenario delivered entirely through the PhiShield platform, thereby assessing both their level of awareness and the effectiveness of the platform itself.

An initial analysis, presents the results aggregated in four different categories, representing the possible actions that targets performed during the campaign:

- **No interaction**: the email has been delivered to the target, but no interaction was registered.
- **Reported**: the target reported the email to the company's security team. Although this action is not currently supported in the existing implementation of PhiShield, it has been included in the analysis of results to provide a complete overview and to highlight the optimal possible outcome for the campaign.
- **Link clicked**: the target accessed the landing page.
- **Data submitted**: the target accessed the landing page and submitted its credentials. As previously explained, data collection was not configured, ensuring users' privacy.

A numerical summary is presented by Table 5.1 below.

| Interaction type | Result | Percentage |
|---|---|---|
| No interaction | 11 | 36.7% |
| Reported | 7 | 23.2% |
| Link clicked | 8 | 26.7% |
| Data submitted | 4 | 13.3% |

Table 5.1. Training results, grouped by interaction type

These results outline that a total of eleven users didn't perform any operation on the received phishing email. In a real scenario, this behaviour can be considered prudent and secure, as it doesn't expose the company to any immediate threat.
Seven participants chose to report the email to the company's security team. Although PhiShield doesn't currently support the tracking of this action, it has been included anyway in this analysis, since it represents the best possible outcome, demonstrating both awareness and proactive behaviour against potential threats. Overall, a total of eighteen targets, slightly more than 50% of the total, responded appropriately to the phishing attempt.

Twelve users accessed the phishing page by clicking the link included in the email. Four of them went further, by submitting their credentials, representing the worst-case scenario for the company. It's important to note that while the compromising rate amounts to 13,3%, even merely clicking on the phishing link can expose the company to potential malicious activities. Therefore, this action must also be considered when evaluating user behaviours that could compromise the organization.

An additional stratified analysis, summarised in Table 5.2, presents the training results from a different perspective. By grouping users according to their position within the company, significant differences emerged.

| Position | No int. | Reported | L. clicked | D. submitted |
|---|---|---|---|---|
| IT specialist - 7 | 4 (57,1%) | 2 (28,6%) | 1(14,3%) | 0 |
| Help desk - 10 | 1 (10%) | 4 (40%) | 3 (30%) | 2 (20%) |
| Finance - 4 | 4 (100%) | 0 | 0 | 0 |
| Human Resources - 3 | 1 (33,3%) | 1 (33,3%) | 0 | 1 (33,3%) |
| Project manager - 6 | 1 (16,7%) | 0 | 4 (66,7%) | 1 (16,7%) |

Table 5.2. Training results, grouped by target position

IT specialists, traditionally considered the most expert group in terms of security, mostly avoided risky interactions, with some even reporting the malicious attempt. However, one did click the link, clearly confirming that there isn't a "perfect" user profile, being able to guarantee a complete immunity from phishing-related threats. This highlights the necessity of continuously involving everyone in awareness programs, as demonstrated in this case.
Moreover, users from the Finance department represented the most careful and prudent group, as no dangerous interactions were recorded.

Help desk, human resources employees and project managers showed the most heterogeneous outcomes, ranging from reporting the email to submitting credentials on the phishing page. This points out the importance of fostering careful and consistent behaviour when handling suspicious emails, minimising potential risks for the company. Additionally, it highlights how less-technical roles could represent a significant vulnerability, emphasizing the need for targeted awareness programs tailored to the specific responsibilities of different departments.

A critical aspect that should always be considered while analysing awareness reports is the that the only truly "safe" outcome is represented by zero interactions with the phishing email. Even a single compromising is sufficient to expose the company to potential threats, including unauthorised access, data breaches and internal network compromises. The evaluation of users' behaviour must treat any interaction as a risk. Continuous training, reinforced through practical simulations and learning paths, is therefore essential to ensure that all employees, regardless of their technical expertise, develop a consistent and proactive approach to cybersecurity threats.
For the purpose of this test, not only the practical outcomes of user interactions were analysed, but also the overall success of the campaign itself, for evaluating the design quality and the current implementation of this platform. PhiShield has proved to be reliable, successfully dispatching the entire phishing campaign, thereby supporting the security awareness assessment for the involved company. In general, the effectiveness of a training campaign strongly relies on the coherency, realism and sophistication of its components, e.g. email templates and landing pages, however, even the most accurate campaign could fail, compromising the quality of the awareness program, without the support of an efficient and reliable platform.

To summarise, the results of the campaign, completely reported in Appendix C, show that, although there is a general level of awareness, risky behaviours are present over all groups. The analysis highlights the importance of combining high-quality content, a robust platform and continuous training in order to strengthen corporate security and reduce the risk that even a few mistakes could lead to significant consequences.

### 5.2.2 Learning results

At the completion of the campaign, every user who performed a risky action for the company, i.e. "Link clicked" or "Data submitted", was automatically enrolled in the designated learning path, which consisted of a lesson covering general phishing-related topics and a final exam composed of eight questions directly tied to the lesson's topics.
Every involved user completed the lesson and attempted the final exam, enabling an assessment of the effectiveness of the key content delivered. Exam scores ranged from 5 to 8, whit 8 being the maximum achievable score, pointing out an overall sufficient, but satisfactory, learning outcome. The Table 5.3 below, summarises users' performances during their learning program.

| Position | Involved users | Outcomes | Average |
|---|---|---|---|
| IT specialist | 1 | [8] | 8 |
| Help desk | 5 | [6,7,5,7,6] | 6,2 |
| Human Resources | 1 | [5] | 5 |
| Project manager | 5 | [5,8,7,5,5] | 6 |

Table 5.3. Learning results, grouped by target position

A deeper analysis of these results highlights a general conclusion: all users involved successfully passed the exam, achieving at least the minimum passing score. This proves that they have effectively acquired the theoretical concepts presented during the learning path. In other words, from a cognitive perspective, participants are aware of the threats posed by phishing.
However, by comparing learning and training results, a significant gap between theory and practice becomes evident. In fact, despite their satisfactory outcome for the exam, different users have actually clicked the phishing link or submitted sensitive data during the simulation. This remarks how the "human factor" represents the most critical vulnerability in this context, where stress, distraction or excessive trust could negatively influence their behaviour. This underscores the behavioural nature of security awareness, which cannot be fully addressed through theoretical content alone, but requires repeated, realistic and practical simulations over time. Moreover, the differences observed among company roles suggest that prior skills and contextual knowledge within a specific operational environment can significantly influence an individual's ability to correctly apply learned concepts in practice. A significant example is the outcome of the single IT specialist involved, who achieved the maximum score on the exam yet still clicked the phishing link during the training. Other groups likewise showed heterogeneous outcomes, further confirming that every department, regardless of the perceived technical level, should be continuously included in awareness programs.

In summary, the primary aspect emerged by this test, which can be easily generalised to the global security awareness posture, is that the theoretical knowledge alone is never sufficient for avoiding the exposition to phishing-related threats. Learning paths should always be designed in harmony with realistic simulations.
PhiShield, thanks to its native integration of both training and learning capabilities, proves to be a strategic tool for addressing this challenge, enabling the translation of theoretical knowledge into practical behaviour in a measurable and replicable manner. This continuous cycle of simulation, feedback and consolidation represents a concrete step toward reducing human-related vulnerabilities, promoting a more resilient security culture.

### 5.2.3 User feedback

To complement the qualitative analysis of users' performance during the test, a dedicated questionnaire was prepared to receive feedback about the platform and the overall experience from all participants. To ensure consistency, a mixed approach has been adopted in designing the questionnaire, combining both objective

and subjective considerations, in order to merge users' perceptions with the operational behaviour of the platform. The complete set of questions is reported in Appendix C, while this section presents only relevant insights.
As previously described, both the operator and the students took part in this complementary survey, by answering to a dedicated set of questions. However, only the twelve students who were enrolled in the learning path were asked to complete the questionnaire.

The operator's responses, although referring to the execution of a single campaign, highlight an overall positive perception of the operational workflow, considered clear and moderately complex. The time required to configure and launch the campaign was also deemed acceptable, indicating a good level of usability and efficiency of the platform. A particularly meaningful insight concerns the chatbot component: while it was perceived as a highly useful tool, it was not significantly employed, suggesting that the operator engaged it mainly for specific corner cases. Despite this, the overall satisfaction levels were positive and the high percentage of error-free operations confirms that the platform enables a smooth and coherent management of the entire process. Notably high is the evaluation regarding the validity of this method as a tool for security awareness, as well as the appreciation for having all functionalities integrated within a single platform, perceived by the operator as an important added value.

Students' responses show a substantially different, yet still meaningful, set of results. The visualisation of the campaign content, i.e. email template and landing page, was judged extremely clear, as demonstrated by the maximum score assigned by all participants. Opinions were more heterogeneous regarding the feedback page: although considered useful, it was perceived as not sufficiently incisive. While this judgement may depend more on the content itself rather than on its delivery through PhiShield, it still indicates that improvements in its visual presentation could be helpful. A further aspect that clearly requires enhancement is the section dedicated to the campaign results, which received noticeably lower evaluations. This suggests wide margins for improvement in the way such information is represented and communicated to the user. The overall accessibility of the lesson was positively appreciated, as well as the support provided by the chatbot component, which was used by approximately half of the students. Finally, the average time required to complete the entire learning path was considered satisfactory and aligned with expectations, taking into account the synthetic yet structured nature of the training materials.
Regarding the effectiveness of the platform in the context of security awareness, students expressed a positive opinion, showing a strong inclination to recommend PhiShield for similar activities. Additionally, also in this case, a positive evaluation was assigned to the overall validity and usefulness of the method for security awareness initiatives within organisational environments. Overall, the feedback provided by the students can be considered satisfactory and appropriate, especially taking into account the prototypical nature of the project.

### 5.2.4 Discussion of results

From this test it becomes evident that PhiShield, also in this prototype version, represents a solid base: the idea of an integrated platform that combines phishing simulations and immediate feedback with learning initiatives constitutes a valid approach in security awareness field. Its advantages include: integration of different functionalities, reliable tracking of the most interesting events from a security standpoint and the presence of an automated support channel, currently represented by the chatbot component. However, these results clearly underscore a wide set of opportunities for improving this platform:

- **Improve feedback delivery**: make the feedback mechanism more explicit and eventually persistent in the platform, allowing users to interact with the chatbot, discuss specific aspects of the campaign and receive valuable suggestions for future scenarios.
- **Strengthen operational usability**: reducing friction points in the operator's workflow by providing contextual guidance, pre-compiled examples and the direct integration of "how-to" documentation within the chatbot component could significantly enhance the overall user experience.
- **Improve AI integration**: the chatbot component could be further enhanced by providing it with campaign and learning metrics, enabling the model to deliver comprehensive, context-aware and user-centric support.

Considering the current prototypic nature of PhiShield, these results show that the direction of the project is appropriate and that it already provides a solid foundation. The platform can, in fact, be employed as a functional operational tool to support real awareness activities. At the same time, the need for further enhancements remains evident, particularly aimed at increasing the realism and overall effectiveness of the awareness campaigns. Introducing such improvements would allow PhiShield to deliver an even more impactful and engaging experience, ultimately strengthening users' behavioural readiness against phishing threats.

# Chapter 6

# Conclusions

This project, practically supported by the prototype designed and implemented, proposed a solid and concrete foundation for the development of a reliable, innovative and highly automated platform capable of supporting companies of any size in carrying out security awareness activities, implicitly improving their overall security posture.

## 6.1 PhiShield contribution to the community

This thesis wants to propose several significant contributions to the field of security awareness through the design and development of PhiShield.
The central achievement of this work lies in the implementation of an integrated platform that combines phishing simulations capabilities with a dynamic learning environment, supporting a continuous user engagement and reinforcing their ability to face social engineering threats. This approach is fully aligned with the concept of Embedded Learning, which promotes the inclusion of security awareness initiatives within everyday operational activities, making it an inherent part of the daily routine. By adopting this model, the traditional separation between phishing simulation tools and learning platforms is effectively removed, resulting in a more coherent, unified and significant awareness program.

Additionally, PhiShield introduces an innovative aspect into the project: the integration of Artificial Intelligence, represented in the prototype by the chatbot component, which is capable of supporting users over the entire process. This feature should be interpreted as an initial implementation, aimed at increasing user engagement and providing immediate feedback, which could be comprehensively extended, for example by introducing behavioural analysis methods, enhancing the overall effectiveness of the training experience.

The potential of the application has been evaluated through the creation of a real-world scenario which, although conducted on a limited group of individuals, confirmed both the concrete feasibility and the practical applicability of Embedded Learning models directly integrated with LLMs. These results represent a robust starting point for the development of advanced security awareness tools, introducing an innovative approach that can be further enhanced in several directions.

## 6.2 Critical aspects and challenges

PhiShield, besides offering an innovative and awareness-oriented approach, presents several aspects and challenges that cannot be disregarded and should be carefully addressed.

Personal data protection represents a foundational element. Any information collected during the process, including, for example, user interactions with phishing landing pages, must be handled maximising confidentiality. A project with these characteristics requires full transparency regarding how data is collected, stored and processed, ensuring that such information is used strictly for formative and security-related purposes. This is essential to avoid any misuse of user data, maintaining trust and complying with ethical and legal requirements.

Artificial Intelligence integrations, within this context, represent a powerful addition that can significantly enhance the operational flow of security awareness initiatives. Beyond improving the overall user experience, AI can be directly embedded within internal processes to automate and optimise several tasks. One of the most relevant examples concerns the automation of learning material generation or the creation of final assessments used to evaluate users' knowledge. However, despite their potential, AI systems cannot guarantee a complete accuracy or contextual adequacy for these activities, where human supervision remains essential to validate, refine and ensure the quality of AI-generated outputs.
Moreover, Artificial Intelligence is strictly tied with privacy. By introducing behavioural analysis capabilities, AI models may need to access to users' performance-related data, eventually increasing the risk of violating the confidential nature of such information. For this reason, it becomes essential to design systems that minimise the collection of personal data, adopting techniques like anonymisation, pseudonymisation or even edge computing, with the goal of reducing the potential exposure of sensitive information.

It is important to emphasise that, as outlined by a recent study conducted by researchers at ETH Zurich [3], the concrete threat posed by phishing attacks goes far beyond the mere mechanical acquisition of theoretical concepts and the periodic reminder of such threats may, in practice, be even more impactful than the learning content itself. For this reason, the system should serve as a tool that enhances users' awareness and critical thinking, promoting continuous learning and periodic reinforcement, to ensure that defence mechanisms remain effective over time.

A critical approach to these aspects will allow PhiShield to evolve as a dynamic project, capable of continuous adaptation and improvement.

## 6.3 Further developments

PhiShield aims to evolve by incorporating multiple functionalities spanning different IT domains, thereby enhancing its overall effectiveness, scalability and capacity to adapt to more complex and dynamic scenarios.

To ensure full privacy compliance and maintain users' data strictly confidential,

a potential enhancement could be the implementation of anonymisation mechanisms directly at database level, allowing the use of real data without risking the exposure of sensitive information.

The integration of LLMs algorithms could enable an in-depth behavioural analysis during phishing simulations, identifying individual risk patterns, such as propensity for link clicks. Based on this information, the platform could leverage AI to generate dynamic campaigns completely tailored to each user, adapting both frequency and difficulty over time. This approach would significantly enhance security awareness outcomes, transforming PhiShield into a fully adaptive tool capable of learning directly from user interactions and continuously improving the realism and effectiveness of its simulations.

Tracking capabilities for email attachments, could allow the system to register whether a file has been downloaded. This functionality would open the scope of phishing simulations, enabling scenarios that incorporate malware delivery, where phishing serves as the primary propagation vector.

Extending the scope of simulations to "X-ishing" techniques, such as smishing (via SMS), vishing (via phone calls), or qishing (using QR codes), allows users to confront a wider range of attack vectors beyond email. This expansion significantly broadens the platform's applicability, transforming it into a comprehensive tool capable of raising awareness about the diverse threats present in real-world scenarios.

To enhance portability and simplify installation, a reliable and efficient deployment method should be introduced to streamline the entire configuration process. Additionally, containerising the whole project represents a valuable option, as it would further improve scalability and ease of distribution across diverse environments.

Finally, from an architectural perspective, the adoption of a multi-tenant structure could enable companies to organise their awareness programs according to specific criteria, such as by customer or department, creating independent and isolated tenants for each. Additionally, integrating the platform with corporate domain services, such as Active Directory [36], could streamline user provisioning, simplifying the setup of campaigns by automatically synchronising with the organisation's existing structure.

The research foundation established by this thesis provides a robust basis for future advancements, as the modular architecture allows for incremental refinements, seamless integration of new technologies and adaptation to evolving cybersecurity threats. By combining a unified platform, AI-driven analytics and user-centric design, PhiShield lays the groundwork for a comprehensive and resilient system capable of enhancing security awareness and mitigating phishing risks across multiple channels.

# Appendix A

# User manual

This manual describes how to set up the technological stack of PhiShield, as described in this thesis. The whole project is provided in a supplementary archive. This guide is intended for users with basic knowledge of Python and Linux operations.

## A.1 Prerequisites

- **Python 3**: to run the application and Celery workers.
- **Pip**: to install project dependencies. Although creating a virtual environment is recommended, the final choice is left to the user.
- **SMTP email server**: to be used for delivering phishing emails.
- **Valid Google account**: to generate Gemini's API key.
- **Communication**: PhiShield running instance should be reachable by every target included in the security awareness program.

## A.2 Redis installation

Redis must be installed on the same host where the application will be executed.

1. Install Redis

   ```
   sudo apt install redis-server
   ```

2. Setup Redis service

   ```
   sudo systemctl enable redis-server
   sudo systemctl start redis-server
   ```

3. Verify Redis service

```
redis-cli ping
# it should respond "PONG"
```

## A.3 SQLite database

The database file, *db.sqlite3* will be already present in the project directory. Every table will be empty by default. No additional action is required for the database setup.

## A.4 Environment setup

### A.4.1 Python packages

Every Python module necessary for this project will be installed by running the following command:

```
pip install -r requirements.txt
```

### A.4.2 Gemini API key

To generate an API key, the following steps are required.

1. Navigate to https://aistudio.google.com
2. Login with a valid account.
3. Select "Get API key" from the menu on the left.
4. Select "Create API key" on top-right corner.

The available models are listed here: https://ai.google.dev/gemini-api/docs/models

### A.4.3 Configuration file

PhiShield uses *.env* file for managing its internal configuration. An empty *.env* file is already available in the project root directory and the following variables must be included:

```
GEMINI_API_KEY=<YOUR_API_KEY>
GEMINI_MODEL=<YOUR_GEMINI_MODEL>
DEFAULT_STUDENT_PASSWORD=<YOUR_DEFAULT_PWD>
```

## A.5 Run PhiShield

To execute the application, two terminal windows are required. To facilitate the process, a Makefile has been made available in the root directory.

1. Verify the status of the Redis service.

   ```
   redis-cli ping
   # it should respond "PONG"
   ```

2. Run Celery workers (Terminal window n.1)

   ```
   # activate environment, if any
   # path: src/
   make celery
   ```

3. Run PhiShield main process (Terminal window n.2)

   ```
   # activate environment, if any
   # path: src/
   make server
   ```

The application will be available at the following link: `http://<HOST_IP>:8000` PhiShield *How to* section will cover every detail related to training and learning activities configuration.

## A.6 Default operator user

By default, PhiShield provides one user with superuser privileges, already created in the database:

```
username: operator
password: uah$fiuafk%jbkj1[h14$14bj12b4{faf
```

This user will have complete access to the whole application.

# Appendix B

# Developer manual

## B.1 Project structure

### B.1.1 Overview

The structure of the project follows the standard convention used by default when creating a new Django project. The main directory is *src/*, which contains different files and folders:

```
.env
/src
|__ audit/
|__ learning/
|__ media/
|__ phishield/
|__ static/
|__ templates/
|__ training/
|__ users/
|__ db.sqlite3
|__ Makefile
|__ manage.py
```

Every Django app, when created, is provided by default with a standard set of files, used for developing different aspects of the project:

- **admin.py**: it allows to register database models to the administration interface, configuring how they are managed and visualised in the admin panel, available at `/admin`
- **apps.py**: it defines the configuration of a single Django app.
- **forms.py**: it defines the structure of the forms used to perform create and update operations on database models.

- **models.py**: it defines Python classes which represent the models of the database entities, with their relations.
- **urls.py**: it defines the configuration path for URLs related to the app.
- **views.py**: it defines apps' business logic. To support modularity, some apps doesn't have a single `views.py` module, while instead, a `/views/` directory is present and contains entity-related views (e.g. campaign_views.py)

### B.1.2 .env file

This file contains environment variables which are loaded at the startup of the main process.

### B.1.3 /src directory

This directory is the project root.

### B.1.4 /src/audit/ directory

This directory contains every file related to the *audit* app, which is in charge of logging operations. It presents the following structure:

```
migrations/
admin.py
apps.py
models.py
```

### B.1.5 /src/learning/ directory

This directory contains every file related to the *learning* app, which is in charge of managing learning entities, such as Lessons and Exams. It presents the following structure:

```
migrations/
templates/
views/
|__ exam_question_views.py
|__ exam_views.py
|__ lesson_views.py
|__ report_views.py
admin.py
apps.py
forms.py
models.py
urls.py
```

### B.1.6 /src/media/ directory

Standard path designed to contain user-uploaded files.

### B.1.7 /src/phishield/ directory

Main project's directory, it contains global configurations of the whole application. The structure is the following:

```
common/
|__ base.py
|__ chatbot_prompts.py
|__ logger.py
asgi.py #for future production environment
celery.py
settings.py
urls.py
views.py
wsgi.py #for future production environment
```

`urls.py` file contains the declaration of every URL related to PhiShield. App-related URLs are located in the specific app directory and included by this one.

#### /phishield/common/ directory

This directory contains utility material, which is users in different sections of the project:

- **base.py**: module that provides a set of base classes, useful for encapsulating repeated operations.
- **chatbot_prompts.py**: contains the definition of two different prompts to provide an initial context to the chatbot component.
- **logger.py**: defines a utility function to easily create a log event, which will be registered in by the audit app.

#### /phishield/celery.py

This module provides the basic configuration for Celery workers.

### B.1.8 /src/static/ directory

It contains static files, such as CSS, JavaScript and images, which are directly served to the clients, without any django elaboration.

### B.1.9 /src/templates/ directory

It contains generic HTML files used by Django to dynamically render web pages. Each application maintains its own *templates/* directory, which stores the templates specific to that particular context. It also includes the *partials/* directory, which stores partial templates that can be incorporated into other ones, as described in the Implementation chapter.

### B.1.10 /src/training/ directory

This directory contains every file related to the *training* app, which is in charge of managing training entities, such as phishing campaigns and its related components. It presents the following structure:

```
migrations/
services/
templates/
views/
|__ attachment_views.py
|__ campaign_views.py
|__ email_sender_views.py
|__ email_template_views.py
|__ landing_page_views.py
|__ report_views.py
|__ target_views.py
|__ targets_group_views.py
|__ tracker_views.py
admin.py
apps.py
forms.py
models.py
tasks.py
tracker_urls.py
urls.py
```

#### /training/services/ directory

This directory contains internal utility functions.

- **email_controller.py**: contains the procedure which creates the email schedule for a provided campaign.
- **tracker_controller.py**: contains a function which manipulates landing pages' structure, adding the standard Django variable `{% csrf_token %}`, required when the user submits any data.

#### /training/tasks.py

This module contains the implementation of different asynchronous tasks, which will be executed by Celery workers:

- **check_scheduled_emails()**: periodic task, configured to be run every minute. For each campaign, it checks if there are any emails that must be sent, by verifying the "send_date" field. If any, it delegates the delivery to another asynchronous task described below.
- **send_phishing_email()**: asynchronous task, in charge of managing the delivery of a single phishing email.
- **engage_students()**: for each completed campaign, if a lesson was configured with the campaign, it manages the enrolment of the targets, which will be selected based on their performance during the campaign. If not present, a user with *student* role is created and linked with the related target. By default, it's possible to configure a standard password, common for every newly created student, defining it in `.env` file.

#### /training/tracker_urls.py

This module contains the URL configuration used for tracking targets' interactions with phishing campaigns.

### B.1.11 /src/users/ directory

This directory contains every file related to the *users* app, which is in charge of authentication operations. It presents the following structure:

```
migrations/
templates/
admin.py
apps.py
forms.py
models.py
signals.py
urls.py
views.py
```

#### /users/signals.py

This module, which is a standard Django file, contains the definition of different methods with the purpose of logging authentication operations, in particular it registers:

- Successful logins

- Successful logouts
- Failed login attempts

### B.1.12 /src/db.sqlite3

SQLite database file.

### B.1.13 /src/Makefile

Created to simplify the execution of the project. This solution cannot be adopted in production environments.

### B.1.14 /src/manage.py

Main script, used to interact with the Django project and starting the server instance.

# Appendix C

# Testing material

## C.1 Training campaign components

### Email template

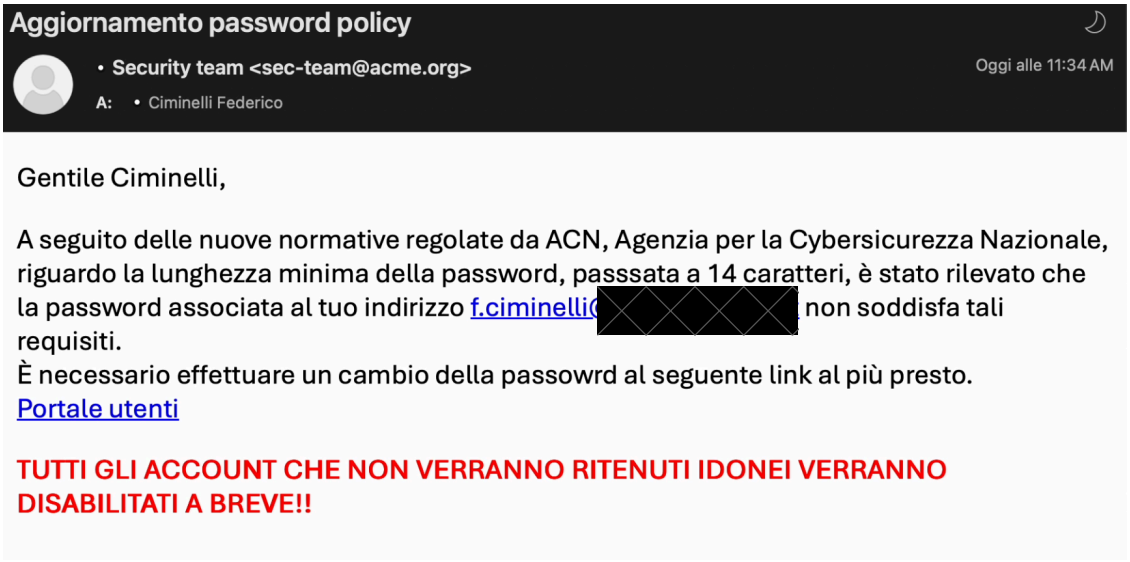


Figure C.1. The phishing email delivered to targets

## Landing page

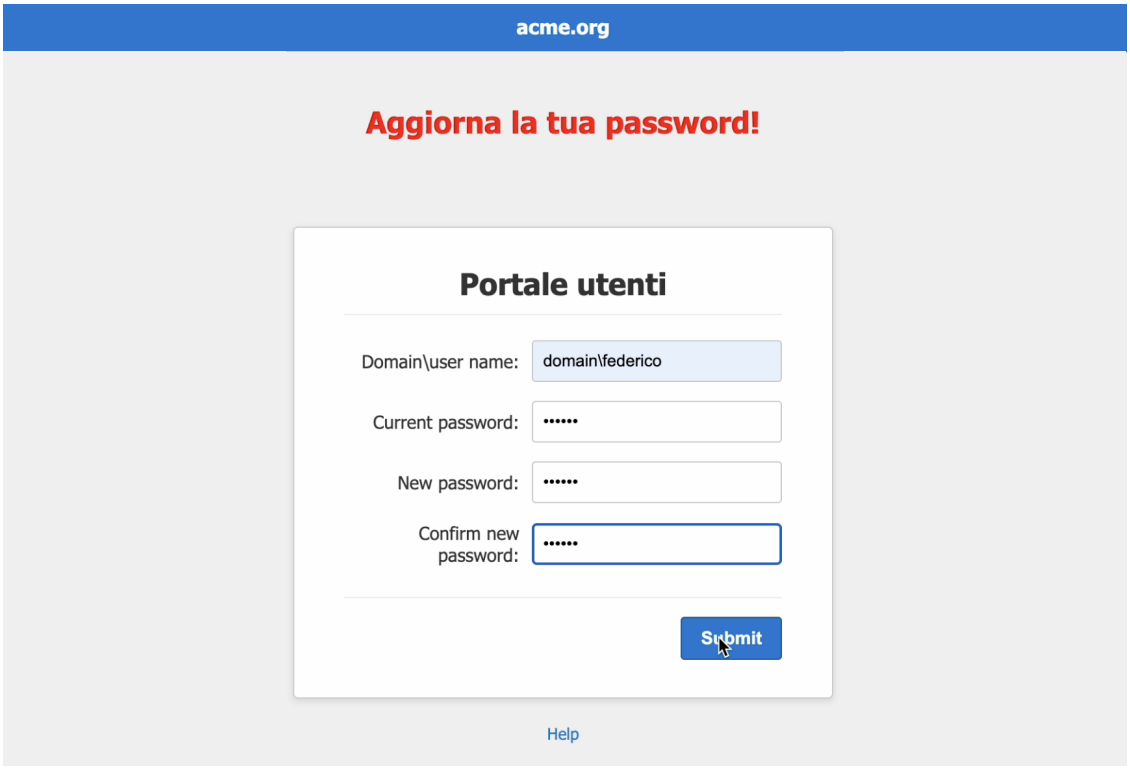


Figure C.2. The landing page, simulating a password change web portal

## Feedback page


Figure C.3. The redirect page, containing the notice of the target involvement

## C.2 Training results

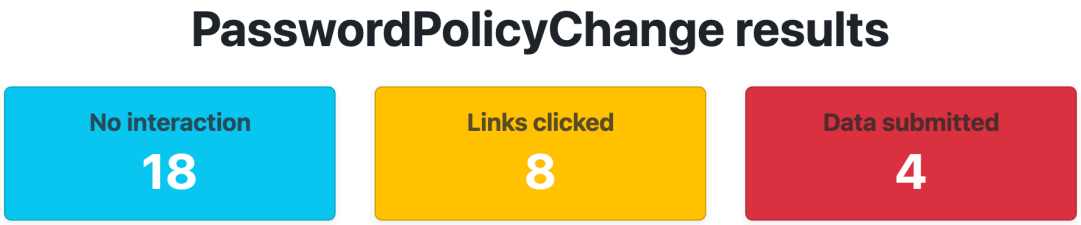


Figure C.4. A screenshot of the aggregated training results

| Campaign name | Target position | Interaction type |
|---|---|---|
| PasswordPolicyChange | IT-specialist | Reported |
| PasswordPolicyChange | IT-specialist | Reported |
| PasswordPolicyChange | IT-specialist | Email sent |
| PasswordPolicyChange | IT-specialist | Email sent |
| PasswordPolicyChange | IT-specialist | Link clicked |
| PasswordPolicyChange | IT-specialist | Email sent |
| PasswordPolicyChange | IT-specialist | Email sent |
| PasswordPolicyChange | HelpDesk | Data submitted |
| PasswordPolicyChange | HelpDesk | Reported |
| PasswordPolicyChange | HelpDesk | Reported |
| PasswordPolicyChange | HelpDesk | Reported |
| PasswordPolicyChange | HelpDesk | Reported |
| PasswordPolicyChange | HelpDesk | Data submitted |
| PasswordPolicyChange | HelpDesk | Link clicked |
| PasswordPolicyChange | HelpDesk | Link clicked |
| PasswordPolicyChange | HelpDesk | Link clicked |
| PasswordPolicyChange | HelpDesk | Email sent |
| PasswordPolicyChange | Finance | Email sent |
| PasswordPolicyChange | Finance | Email sent |
| PasswordPolicyChange | Finance | Email sent |
| PasswordPolicyChange | Finance | Email sent |
| PasswordPolicyChange | HumanResources | Data submitted |
| PasswordPolicyChange | HumanResources | Reported |
| PasswordPolicyChange | HumanResources | Email sent |
| PasswordPolicyChange | ProjectManager | Email sent |
| PasswordPolicyChange | ProjectManager | Data submitted |
| PasswordPolicyChange | ProjectManager | Link clicked |
| PasswordPolicyChange | ProjectManager | Link clicked |
| PasswordPolicyChange | ProjectManager | Link clicked |
| PasswordPolicyChange | ProjectManager | Link clicked |

Table C.1. Training results

## C.3 Learning components

### Exam questions

The exam designed for the test was composed of eight questions, one point each, for a total of eight points achievable.

- **Q1**: Phishing emails often create a sense of urgency to trick users into taking quick actions (*True*).
- **Q2**: Legitimate organizations will never ask you to verify your credentials via email (*True*).
- **Q3**: It is always safe to click on links in emails if the sender name looks familiar (*False*).
- **Q4**: Checking the sender email address and domain can help detect phishing attempts (*True*).
- **Q5**: Phishing can only happen through email, not through phone calls or text messages (*False*).
- **Q6**: Phishing emails often contain spelling mistakes and unusual grammar (*True*).
- **Q7**: Using two-factor authentication (2FA) can help protect against phishing attacks (*True*).
- **Q8**: Phishing attacks can lead to identity theft and financial loss (*True*).

## C.4 Learning results

| Exam name | Target position | Outcome (out of 8) |
|---|---|---|
| PhishingExam | IT-specialist | 8 |
| PhishingExam | HelpDesk | 6 |
| PhishingExam | HelpDesk | 7 |
| PhishingExam | HelpDesk | 5 |
| PhishingExam | HelpDesk | 7 |
| PhishingExam | HelpDesk | 6 |
| PhishingExam | HumanResources | 5 |
| PhishingExam | ProjectManager | 5 |
| PhishingExam | ProjectManager | 8 |
| PhishingExam | ProjectManager | 7 |
| PhishingExam | ProjectManager | 5 |
| PhishingExam | ProjectManager | 5 |

Table C.2. Learning results

## C.5 Surveys

### Operator questionnaire

- **O1**: Quanto e' chiaro il flusso di gestione di campagne, lezioni ed esami?
  *How clear is the workflow for managing campaigns, lessons, and exams?*

- **O2**: Quante volte hai dovuto consultare la sezione "how to" durante la creazione di una campagna, una lezione o un esame?
  *How many times did you need to consult the "how to" section while creating a campaign, lesson, or exam?*

- **O3**: Quanto tempo medio hai impiegato per configurare e lanciare una campagna?
  *What was the average time you spent configuring and launching a campaign?*

- **O4**: Quanto trovi utile e ben integrata la chatbot all'interno dell'applicazione?
  *How useful and well-integrated do you find the chatbot within the application?*

- **O5**: Quante volte hai consultato la chatbot durante le attivita' operative?
  *How many times did you consult the chatbot during operational activities?*

- **O6**: Quanto e' utile ed esaustiva la sezione dei risultati?
  *How useful and comprehensive do you find the results section?*

- **O7**: Quanto ti senti soddisfatto complessivamente dell'esperienza come operatore?
  *How satisfied are you overall with your experience as an operator?*

- **O8**: Quante operazioni hai completato con successo senza errori o messaggi di validazione?
  *How many operations did you successfully complete without errors or validation messages?*

- **O9**: Quanto consideri valido questo metodo per la gestione della security awareness in contesti aziendali?
  *How effective do you consider this method for managing security awareness in corporate environments?*

- **O10**: Quanto valuti fondamentale avere una piattaforma che unisce sia training che learning?
  *How essential do you consider having a platform that combines both training and learning?*

### Operator feedback

Every column of the Table C.3 corresponds to one question in the survey (except for the first one).

| User | O1 | O2 | O3 | O4 | O5 | O6 | O7 | O8 | O9 | O10 |
|---|---|---|---|---|---|---|---|---|---|---|
| Operator | 3 | $< 50\%$ | 15 | 4 | $50\% < x < 75\%$ | 2 | 3 | $< 50\%$ | 4 | 5 |

Table C.3. Operator feedback

**Students' questionnaire**

- **S1**: Quanto valuti chiara la visualizzazione di email di phishing e landing page?
  *How clear do you find the visualization of phishing emails and landing pages?*

- **S2**: Quanto ha aiutato la pagina di feedback finale, con spiegazioni sulla campagna?
  *How helpful was the final feedback page with explanations about the campaign?*

- **S3**: Quanto e' stata chiara e utile la visualizzazione dei risultati delle campagne?
  *How clear and useful was the visualization of the campaign results?*

- **S4**: Quanto e' stata accessibile la lezione erogata tramite la piattaforma?
  *How accessible was the lesson delivered through the platform?*

- **S5**: Quanto ti sei sentito supportato/a dalla chatbot durante la fruizione della lezione?
  *How supported did you feel by the chatbot during the lesson?*

- **S6**: Quante volte hai richiesto chiarimenti alla chatbot durante il percorso?
  *How many times did you request clarifications from the chatbot during the process?*

- **S7**: Quanto tempo medio hai impiegato per completare la lezione e sostenere l'esame finale?
  *What was the average time you needed to complete the lesson and take the final exam?*

- **S8**: Quanto ritieni efficace la piattaforma per il tuo apprendimento in merito a temi di cybersecurity?
  *How effective do you consider the platform for your learning regarding cybersecurity topics?*

- **S9**: Quanto consigli questa modalita' di apprendimento ad altri studenti?
  *How likely are you to recommend this learning method to other students?*

- **S10**: Quanto consideri valido questo metodo per la gestione della security awareness in contesti aziendali?
  *How effective do you consider this method for managing security awareness in corporate environments?*

### Students' feedback

Every column of the Table C.4 corresponds to one question in the survey (except for the first one). The questionnaire was anonymous.

| **User** | **S1** | **S2** | **S3** | **S4** | **S5** | **S6** | **S7** | **S8** | **S9** | **S10** |
|---|---|---|---|---|---|---|---|---|---|---|
| Stud.1 | 5 | 4 | 2 | 3 | 4 | 50% | 30min | 4 | 5 | 5 |
| Stud.2 | 5 | 4 | 1 | 4 | 4 | 50% < x < 75% | 45min | 5 | 4 | 4 |
| Stud.3 | 5 | 2 | 2 | 3 | 4 | 50% < x < 75% | 35min | 4 | 5 | 5 |
| Stud.4 | 5 | 5 | 2 | 4 | 5 | 50% | 35min | 4 | 3 | 4 |
| Stud.5 | 5 | 5 | 4 | 4 | 5 | 50% < x < 75% | 40min | 5 | 5 | 5 |
| Stud.6 | 5 | 2 | 2 | 2 | 3 | 50% | 45min | 3 | 4 | 4 |
| Stud.7 | 5 | 3 | 2 | 3 | 4 | 50% | 40min | 4 | 4 | 5 |
| Stud.8 | 5 | 4 | 3 | 3 | 4 | 50% < x < 75% | 35min | 3 | 4 | 5 |
| Stud.9 | 5 | 4 | 2 | 3 | 2 | < 50% | 40min | 3 | 3 | 4 |
| Stud.10 | 5 | 4 | 4 | 4 | 4 | 50% | 35min | 4 | 4 | 4 |
| Stud.11 | 5 | 3 | 2 | 4 | 4 | 50% | 40min | 4 | 3 | 3 |
| Stud.12 | 5 | 2 | 2 | 4 | 3 | < 50% | 45min | 2 | 2 | 3 |

Table C.4. Students feedback

# Appendix D

# Attack demo evidences


Figure D.1. SocialFish execution

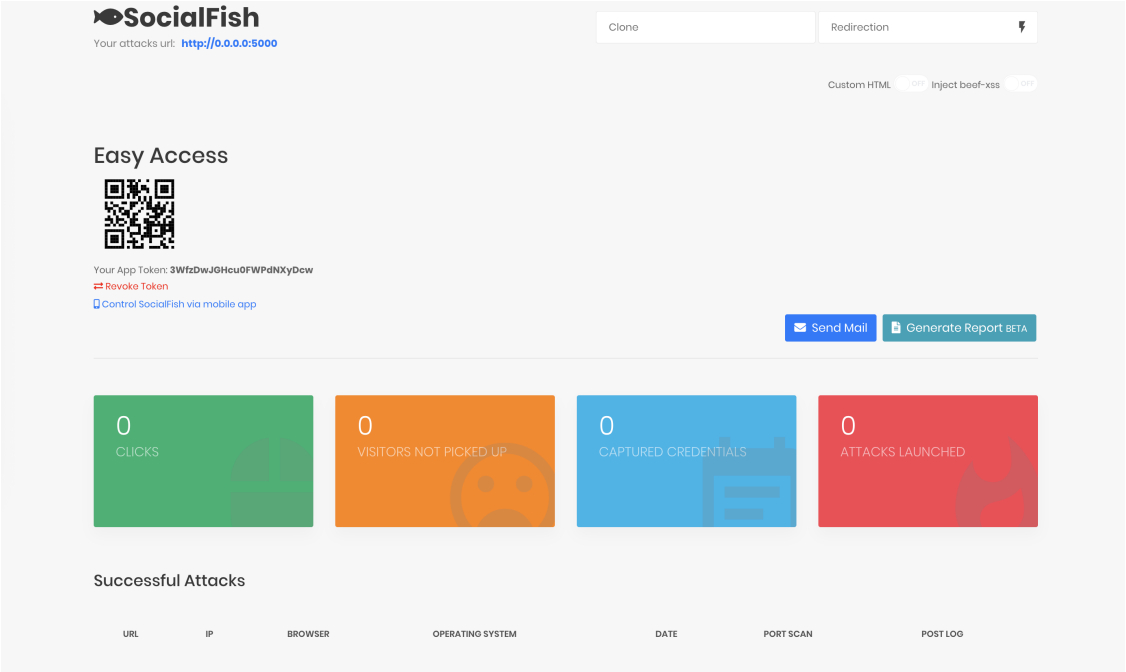


Figure D.2. SocialFish configuration console

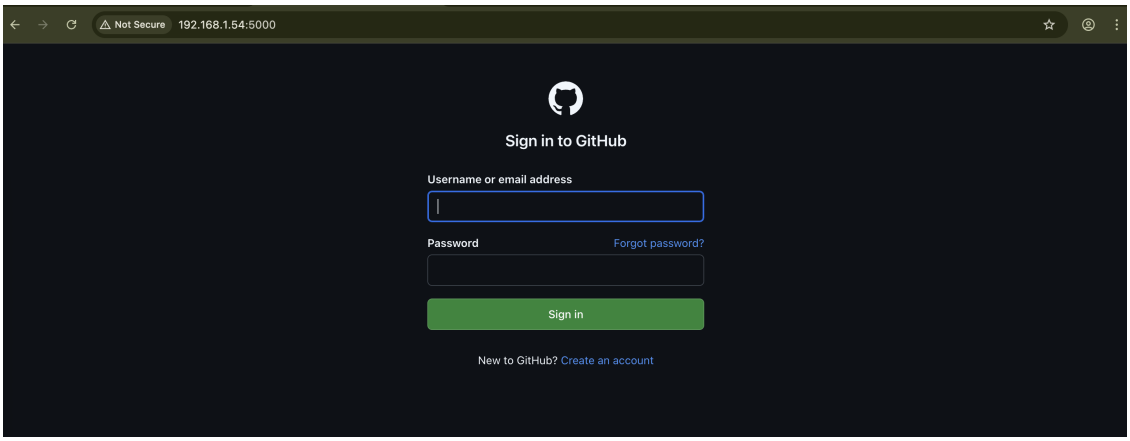


Figure D.3. Clone URL page

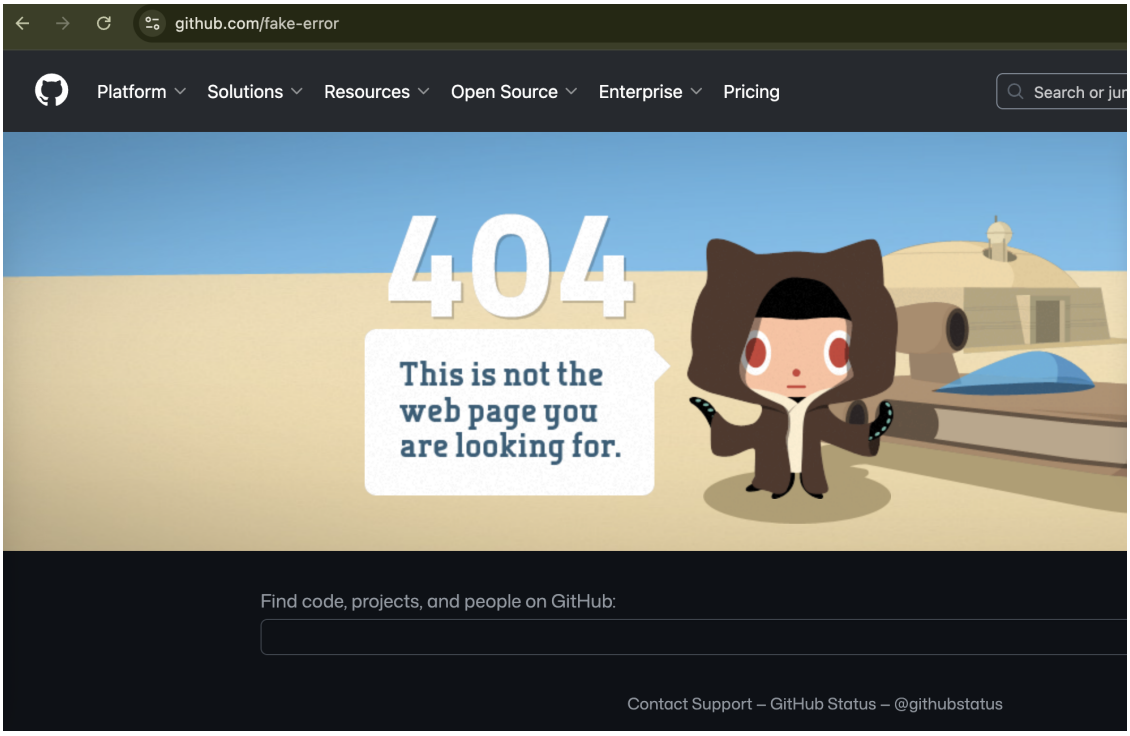


Figure D.4. Redirect URL page

# Bibliography


[1] M. Khalil, "Phishing statistics 2025: Ai-driven attacks, costs and trends", DeepStrike Blog, 2025

[2] S. Alder, "Knowbe4 phishing benchmarking study", The HIPAA Journal, 2022

[3] D. Lain, T. Jost, S. Matetic, K. Kostiainen, and S. Capkun, "Content, nudges and incentives: A study on the effectiveness and perception of embedded phishing training", 2024

[4] M. B. Ahmad and M. A. Shehu, "Enhancing phishing awareness strategy through embedded learning tools: A simulation approach", Archives of Advanced Engineering Science, 2023, DOI10.47852/bonviewAAES32021392

[5] Keepnet, "Security awareness training statistics", Keepnet Blog, 2025

[6] UndeadSec, "Socialphish", 2019, https://github.com/UndeadSec/SocialFish/wiki

[7] Wikipedia, "2020 twitter account hijacking", 2025

[8] Cymulate, "A grand attack on the palace-mgm resorts and caesars cyber attacks", Cymulate Blog, 2025

[9] KnowBe4, "Knowbe4", https://www.knowbe4.com

[10] Proofpoint, "Proofpoint", https://www.proofpoint.com/us

[11] TeamGuard AI, "Adaptivesecurity", https://www.adaptivesecurity.com

[12] J. Wright, "Gophish", 2017, https://getgophish.com

[13] Maticmind, "Ai threat landscape 2025", AnalisiDifesa, 2025

[14] R. Bhandari, "Ai and cybersecurity: Opportunities, challenges, and governance", EDPACS, 2025, DOI10.1080/07366981.2025.2544363

[15] Python Software Foundation, "Python", https://www.python.org

[16] Django Software Fountation, "Django", https://www.djangoproject.com

[17] SQLite Consortium, "Sqlite", https://sqlite.org

[18] Wikipedia, "Cross-site scripting", 2025

[19] Wikipedia, "Cross-site request forgery", 2025

[20] Wikipedia, "Sql injection", 2025

[21] Wikipedia, "Clickjacking", 2025

[22] A. Solem, "Celery", https://docs.celeryq.dev/en/stable/

[23] Redis, "Redis open source", https://redis.io/docs/latest/operate/oss_and_stack/

[24] A. Solem, "Celery: Using redis", https://docs.celeryq.dev/en/stable/getting-started/backends-and-brokers/redis.html

[25] Celery, "Django celery beat", https://github.com/celery/django-celery-beat

[26] Google, "Google gemini", https://gemini.google/it/about/

[27] Bootstrap Team, "Bootstrap", https://getbootstrap.com

[28] A. Hupp, "python-magic", https://pypi.org/project/python-magic/
[29] S. Kumar, "python-dotenv", https://pypi.org/project/python-dotenv/
[30] I. Shalyapin, "django-cleanup", https://pypi.org/project/django-cleanup/
[31] L. Richardson, "beautifulsoup4", https://pypi.org/project/beautifulsoup4/
[32] NumFOCUS, "pandas", https://pandas.pydata.org
[33] Django Software Foundation, "Django url dispatcher"
[34] Django Software Foundation, "Django path converters"
[35] mailhog, "Mailhog", 2020, https://github.com/mailhog/MailHog
[36] Wikipedia, "Active directory", 2025