Міністерство освіти і науки України

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»

Кафедра інформатики факультету інформатики

Кваліфікаційна робота

освітній ступінь – бакалавр на тему

«Розробка додатку для симуляції фішингових атак в компанії (з аналізом помилок клікерів)»

Виконала: студентка 4-го року навчання
Освітньої програми «Комп'ютерні науки»
Прокопеня Поліна Сергіївна
Керівниця: Хряпа О.І., старша викладачка

_____ 2025 р.

Київ 2025

ЗМІСТ

ВСТУП

Фішингові атаки є одним з найпоширеніших типів кіберзагроз, які націлені на обман користувачів з метою отримання конфіденційної інформації або компрометації систем. Вони використовують слабкі сторони людської поведінки, що робить їх надзвичайно небезпечними для організацій. За даними звіту Verizon, соціальна інженерія у вигляді фішингу та викрадення облікових даних причетна майже до 80% усіх витоків даних [1]. Це означає, що значна частина успішних атак починається саме з того, що хтось зі співробітників став жертвою фішингового повідомлення.

Статистика демонструє, що без належного навчання чимало працівників схильні піддаватися на фішингові трюки. Наприклад, близько третини непідготовлених співробітників не проходять перевірку фішинговою симуляцією, тобто натискають на шкідливі посилання або розголошують дані [2]. Технічні засоби захисту (такі як спам-фільтри) не в змозі на 100% відфільтрувати фішингові листи [3], тому остаточна ланка захисту - це свідомі та обережні дії самих користувачів. Таким чином, перед компаніями постає завдання зміцнення «людського файрволу» через підвищення обізнаності персоналу щодо фішингових атак. Одним із дієвих підходів для цього є проведення симуляцій фішингових атак - контрольованих розсилок фішингових листів усередині організації, які дозволяють перевірити пильність співробітників та навчити їх безпечній поведінці.

Розробка спеціалізованого додатку для симуляції фішингових атак у компанії є важливою, тому що такий інструмент дасть змогу систематично оцінювати вразливість персоналу до соціальної інженерії. Аналіз помилок співробітників-«клікерів» (тобто тих, хто перейшов за фішинговим посиланням) після проведення симуляцій особливо цінний для виявлення типових слабких місць в знаннях чи уважності користувачів. Таким чином, обрана тема роботи обґрунтована нагальною потребою підприємств у засобах для тренування і перевірки працівників у сфері протидії фішингу.

Проблема протидії фішинговим атакам є вкрай актуальною на сучасному етапі розвитку кібербезпеки. Кількість фішингових інцидентів неухильно зростає, особливо в останні роки. Зокрема, під час пандемії COVID-19 було зафіксовано різке збільшення фішингової активності (за деякими даними, на 600%-9000%) [3], що пов'язано з масовим переходом на віддалену роботу та загальним підвищенням рівня стресу користувачів. Окрім того, зловмисники дедалі частіше вдаються до спрямованих атак (спеар-фішинг), які ретельно персоналізують фішингові листи під конкретну організацію чи навіть людину. Такі таргетовані атаки ще важче виявити, тому традиційні методи захисту потребують доповнення у вигляді навчання персоналу.

Багато компаній сьогодні усвідомлюють критичність людського фактора та впроваджують програми підвищення кіберобізнаності, включно з регулярними фішинговими тестуваннями співробітників. Міжнародні стандарти і практики кібербезпеки (наприклад, ISO/IEC 27001, рекомендації NIST) наголошують на необхідності тренінгів для користувачів щодо протидії соціальній інженерії. Отже, розроблення інструментів, що дають можливість ефективно імітувати фішингові атаки та аналізувати поведінку персоналу, є своєчасним і затребуваним. Актуальність даної роботи зумовлена потребою підприємств у проактивних заходах проти фішингу, адже превентивне навчання значно дешевше та ефективніше, ніж ліквідація наслідків реальних кіберінцидентів.

Метою даної кваліфікаційної роботи є розробка програмного додатку для симуляції фішингових атак у корпоративному середовищі та проведення аналізу помилок, яких припускаються користувачі-«клікери» під час таких атак. Досягнення поставленої мети сприятиме вдосконаленню навчання працівників основам кібергігієни та зниженню ризику успішних фішингових атак на підприємство.

Для реалізації зазначеної мети необхідно вирішити такі завдання:

1. Дослідити природу та різновиди фішингових атак, а також сучасні методи протидії їм, особливо в контексті навчання користувачів.
2. Проаналізувати існуючі інструменти й підходи для симуляції фішингових атак та підвищення обізнаності співробітників (огляд літератури і доступних програмних рішень).
3. Розробити вимоги до функціоналу та архітектури додатку для проведення фішингових симуляцій у компанії (визначити сценарії атак, методи відстеження реакції користувачів тощо).
4. Реалізувати програмний додаток відповідно до розроблених вимог, забезпечивши можливість гнучкого налаштування фішингових сценаріїв і збору даних про дії користувачів.

1. ТЕОРЕТИЧНІ ОСНОВИ ФІШИНГОВИХ АТАК

1.1. Фішингові атаки: поняття та різновиди

Фішинг - це поширена форма кіберзлочинності, яка базується на використанні обманливих повідомлень з метою викрадення конфіденційної інформації у жертв. [4] З часу першої зафіксованої фішингової атаки на початку 1990-х років ці методи значно еволюціонували, перетворившись на складні схеми соціальної інженерії, що й досі становлять значну частину шахрайств в інтернеті. По суті, фішинг - це форма соціальної інженерії, що спрямована на крадіжку особистих даних шляхом імітації авторитетних джерел. Іншими словами, зловмисник (так званий «фішер») видає себе за довірену особу або організацію, щоб змусити жертву добровільно розкрити чутливу інформацію (наприклад, паролі або банківські реквізити) або виконати дії, які приносять вигоду зловмиснику. Наприклад, згідно з визначенням Агентства з кібербезпеки США (US-CERT), фішинг - це форма соціальної інженерії, що передбачає використання електронної пошти або шкідливих вебсайтів для отримання особистої інформації під виглядом легітимної організації. Аналогічно, Якобссон та Майєрс (2006) пропонують детальне визначення, описуючи фішинг як «форму соціальної інженерії, за якої зловмисник, також відомий як фішер, намагається шахрайським шляхом отримати конфіденційні облікові дані, імітуючи електронні повідомлення від надійної або публічної організації автоматизованим способом». Таким чином, фішингові атаки спираються на довіру людини, використовуючи як соціальні, так і технічні засоби маніпуляції - наприклад, підроблені електронні листи чи вебсторінки. Основна стратегія - переконати жертву в достовірності повідомлення, щоб змусити її розкрити дані або встановити шкідливе програмне забезпечення.

Різновиди фішингу: З розвитком цифрових технологій фішинг розвинувся в декілька підтипів, які зазвичай класифікуються за засобом зв'язку або за ступенем таргетованості. Ранні фішингові атаки здебільшого

здійснювалися через масово розповсюджені електронні листи, але сучасні зловмисники також використовують такі канали, як текстові повідомлення та голосові дзвінки, а також комбіновані підходи. Нижче наведено основні різновиди фішингу:

- **Фішинг через електронну пошту (Email Phishing)**: класична форма фішингу, що полягає у надсиланні підроблених листів, які імітують повідомлення від надійних джерел. Зловмисник надсилає підроблений електронний лист (з адреси, що нагадує легітимне джерело) широкій аудиторії, спонукаючи одержувачів перейти за шкідливим посиланням або відкрити заражене вкладення та розкрити облікові чи особисті дані. Фішинг електронною поштою залишається найпоширенішим типом фішингу, оскільки він може охопити тисячі потенційних жертв з мінімальними зусиллями.
- **Спрямований фішинг (spear phishing):** на відміну від масових розсилок, спрямований фішинг орієнтований на конкретну особу або організацію. Зловмисник попередньо вивчає ціль (наприклад, її посаду або зв’язки) і персоналізує повідомлення для підвищення довіри. Саме тому spear phishing часто використовується у шпигунстві або для проникнення в корпоративні системи.
- **Китобійний фішинг (Whaling)**: підтип spear phishing, орієнтований на керівників високого рівня - директорів, фінансових менеджерів, державних посадовців. Атаки whaling зазвичай імітують офіційні запити від юридичних структур, членів ради або ділових партнерів, і використовують вкрай персоналізований контент. Успішна атака може дати доступ до найкритичніших ресурсів компанії, тому whaling вважається однією з найскладніших форм фішингу.
- **Смішинг (SMS-фішинг):** форма фішингу, що здійснюється через текстові повідомлення (SMS). Зловмисник видає себе за легітимну установу (наприклад, банк чи службу доставки), надсилаючи

повідомлення з шкідливим посиланням або номером телефону. Часто такі повідомлення містять елементи терміновості (наприклад, попередження про безпеку або сповіщення про виграш), щоб змусити користувача діяти імпульсивно [5]. Зі зростанням використання мобільних пристроїв smishing набув масштабного поширення. Користувачі можуть менше очікувати шахрайства в текстових повідомленнях, що робить смішінг ефективною тактикою, якщо ціль не добре її розуміє.

- **Вішинг (голосовий фішинг):** атаки, що реалізуються через телефонні дзвінки або голосові повідомлення. Нападник видає себе за співробітника банку, технічної підтримки або урядової установи, намагаючись отримати конфіденційні дані або доступ до облікових записів. Часто застосовуються технології підміни номера (caller ID spoofing) або автоматизовані дзвінки (VoIP). Наприклад, шахрай може представитись банківським працівником і запропонувати «перевірити» дані рахунку, або стверджувати, що комп’ютер жертви заражено вірусом.

Кожна з перелічених форм фішингу ґрунтується на одній і тій самій стратегії: обман і маніпуляція довірою, однак реалізується через різні комунікаційні канали та рівні таргетування. **Фішинг електронною поштою** забезпечує широке охоплення, тоді як **спірфішинг** та **китобійний фішинг** звужують фокус до конкретних жертв (часто вимагаючи більш детальної розвідки цілі). **Смішинг** та **вішинг** розширюють вектор атаки на мобільні пристрої та голосові дзвінки, демонструючи адаптивність зловмисників до змін у цифровій поведінці користувачів. Розуміння різновидів фішингових атак є критично важливим для побудови ефективних програм кібербезпеки, тренінгів і симуляцій. Кожен тип вимагає окремих підходів до виявлення і протидії, а знання характерних

ознак фішингових повідомлень дозволяє користувачам вчасно ідентифікувати загрозу та уникнути потенційних втрат.

1. 2. Огляд симуляції фішингових атак як методу захисту організації

Згідно з статистичними даними, обсяги фішингових атак різко зросли за останні роки, а середня вартість порушення, пов'язаного з фішингом, досягла близько 4,9 мільйона доларів [9]. Така статистика підкреслює, що суто технічних засобів захисту (таких як спам-фільтри) недостатньо; зловмисники регулярно обходять їх, залишаючи співробітників останньою лінією захисту.

Одним із методів, який компанії застосовують для підвищення кіберпезпеки організації, шляхом підвищення обізнаності співробітників щодо фішингу, є **симуляція фішингових атак**. Структура типової програми симуляції фішингу включає кілька ключових кроків:

1. **Планування та проектування**: вибір фішингових шаблонів або сценаріїв, які відображають відповідні загрози (наприклад, фіктивні запити на скидання пароля або підроблені електронні листи-фактури)

2. **Виконання**: надсилання імітаційних фішингових листів цільовим користувачам (часто випадково або за відділом) і відстеження їх взаємодії (чи відкривають вони лист, натискають посилання, завантажують вкладення або повідомляють про лист як про підозрілий).

3. **Зворотній зв'язок і навчання**: негайно або згодом інформувати користувачів, які потрапили на імітовану фішингову атаку, про помилку та надання коригувальної освіти.

Фішингові симуляції мають дві основні мети: **вимірювання** та **навчання**. По-перше, симуляції оцінюють рівень людського ризику в організації, а по-друге, служать реалістичною навчальною вправою. З часом ця регулярна практика має на меті виробити звички пильності та покращити прийняття рішень під тиском, здавалося б, термінового електронного

листа. Найважливіше те, що симуляції фішингу не призначені для того, щоб присоромити чи покарати працівників. Вони повинні проводитися з орієнтованим на навчання підходом – мета полягає в тому, щоб допомогти людям вчитися на помилках, а не виокремлювати їх для дисциплінарних стягнень. Багато експертів застерігають, що ставлення до невдач у фішинговому тесті як до порушення, за яке працівника потрібно покарати (наприклад, утримання заробітної плати або публічне приниження співробітника) є контрпродуктивним, оскільки створює культуру страху, а не відкритості. Натомість організаціям рекомендується розглядати симуляції як можливість для покращення колективної безпеки, де навіть помилки конструктивно використовуються для підвищення обізнаності.

На відміну від пасивного навчання (такого як лекції чи відео), симуляції активно перевіряють, чи працівники робитимуть безпечний вибір, коли шахрайський електронний лист потрапляє до їхньої поштової скриньки. З’являється все більше доказів того, що добре впроваджені програми симуляції фішингу можуть значно зменшити ймовірність того, що співробітники стануть жертвами фішингових атак. Наприклад, один галузевий звіт показав, що люди, які пройшли таке симуляційне навчання, на 30% рідше натискали на посилання у фішингових електронних листах порівняно з непідготовленими працівниками [10]. У довгострокових дослідженнях спостерігаються постійні покращення: працівники, як правило, з часом зазнають невдач з меншою частотою, оскільки вони проходять повторні симуляції фішингу в поєднанні з підвищенням обізнаності [11]. Аналогічно, галузевий бенчмаркінговий аналіз 11,9 мільйона користувачів у понад 50 000 організаціях, проведений у 2024 році, показав «радикальне зниження необережних кліків після 90 днів та 12 місяців навчання з обізнаності щодо безпеки за новими стандартами» [12]. Залучення співробітників до постійного навчання з питань фішингу корелює з меншою кількістю інцидентів безпеки: одне дослідження

показало, що компанії, які проводять регулярні тренінги з підвищення обізнаності щодо безпеки (включаючи симуляції фішингу), зазнали 70% зниження загальної кількості інцидентів безпеки порівняно з тими, хто не мав такого навчання [10].

Хоча ця статистика та галузеві звіти малюють обнадійливу картину, інші дослідження повідомляли про **неоднозначні або протилежні** результати. У відомому рандомізованому контрольованому дослідженні Lain et al. надали одній групі співробітників вбудоване навчання з фішингу (через статичну інформаційну веб-сторінку, яка відображалася після кожної невдалої симуляції), тоді як інша група взагалі не пройшла жодного навчання. Дивно, але група із навчанням показала гірші результати – у них був вищий рівень кліків на фішингові симуляції, ніж у ненавченої контрольної групи. Автори дослідження припустили, що навчальний матеріал, що містить багато тексту, міг бути неефективним або навіть спричиняти надмірну впевненість у собі, що призводило до зниження обережності. Ці розбіжні висновки підкреслюють, що **структура навчання, якість контенту та методологія мають велике значення**. Різноманітні, збагачені контекстом симуляції з інтерактивним зворотним зв'язком, як правило, є більш ефективними, тоді як поверхневі або суто дидактичні тренінги можуть принести незначну користь. Також можливо, що деяким користувачам потрібні різні стилі навчання або повторні втручання, перш ніж їхня поведінка помітно зміниться. Тим не менш, незважаючи на деякі суперечливі результати, консенсус як у галузі, так і в академічних колах полягає в тому, що *добре розроблені симуляції фішингу можуть бути дуже ефективним методом підвищення обізнаності про безпеку.* Наприклад, після зарахування користувачів до адаптивної програми навчання з питань фішингу з персоналізованим зворотним зв'язком, організації спостерігали значне зростання рівня повідомлень співробітників про спроби фішингу (в одному випадку в середньому 60%,

що значно перевищує базовий показник ~7% в організаціях, які проводять лише щорічне навчання) [9]. Більш висока звітність у поєднанні з меншою кількістю кліків призводить до більш раннього виявлення атак та меншої кількості випадків реальних порушень безпеки.

Нижче наведено порівняльну таблицю між симуляцією фішингових атак і традиційними методами за критеріями залучення працівників, впливу на поведінку, вартості та масштабованості:

Критерій	**Симуляція фішингу**	**Очні тренінги**	**Онлайн-курси/відео**	**Інформаційні бюлетені/постери**
Залучення працівників	Високе, завдяки інтерактивності та змагальному елементу. Реалістичний сценарій краще привертає увагу.	Середнє, може бути ефективним при високій якості, але зазвичай важко утримати увагу довгий час.	Середнє-низьке, часто сприймаються як формальність і можуть швидко набриднути.	Низьке, легко ігноруються як фоновий шум.
Зміна поведінки	Висока, завдяки реалістичності та повторюваності вправ. Практичні вправи закріплюють безпечні звички.	Середня, корисна для передачі знань, але не завжди впливає на практичну поведінку.	Низька, переважно надають теоретичні знання без прямої перевірки практичних навичок.	Низька, не стимулюють практичного застосування знань.
Вартість та ресурси	Помірна, є витрати на ліцензії, адміністрування, але мають	Висока, трудомісткі, вимагають значних організаційних	Середня, мають початкові витрати на розробку,	Низька, дешеві, але мало впливають на

	високу окупність завдяки уникненню атак.	ресурсів та часу працівників.	потребують регулярного оновлення.	зменшення ризиків.
Масштабованість та частота	Висока, завдяки автоматизації легко проводити регулярно для великої кількості працівників.	Низька, важко масштабувати, складно підтримувати однакову якість.	Висока, але потребують періодичного оновлення.	Висока, але ефективність не зростає з масштабом.

Основні **переваги симуляцій фішингових атак**: практика та закріплення знань у реалістичних сценаріях, результати та показники ефективності легко виміряти (набагато важче виміряти вплив брошюри з безпеки чи семінару в конкретних термінах), посилена залученість працівників та поширення культури кібербезпеки (працівники починають пишатися своїми результатами, і можуть навіть ділитися досвідом фішингових тестів з колегами, тим самим підвищуючи обізнаність про безпеку; залученість ще більше посилюється, коли організації використовують позитивне підкріплення - наприклад, хвалячи відділи з хорошими показниками), реалістичне стрес-тестування захисних систем.

Основні **недоліки симуляцій фішингових атак**: стрес співробітників (у деяких випадках агресивні фішингові тести, наприклад, ті, що заманюють співробітників фальшивими обіцянками бонусів чи винагород, оголошення відділу кадрів про звільнення викликали гнів та розчарування [13]), втома та хибнопозитивні результати (занадто багато симуляцій може призвести до втоми, коли користувачі починають ігнорувати листи або імпульсивно натискати на них [11]; співробітники можуть почати бачити фішинг всюди - співробітник може ігнорувати легітимний електронний лист від нового клієнта, підозрюючи його як тестовий, або неодноразово

позначати рутинні повідомлення як підозрілі), негативні реакції та етичні міркування, складність впровадження для малих та середніх компаній, обмеження реалістичності (симуляції часто не можуть повторити сценарії, коли зловмисник телефонує співробітнику після надсилання електронного листа, або тривалу фішингову розмову).

Підсумовуючи, зазначимо, що тренінги з симуляції фішингу, якщо їх використовувати розумно, є ефективним методом посилення захисту організації від кіберзагроз. Організації, які успішно інтегрували фішингові симуляції, повідомляють не лише про меншу кількість фішингових інцидентів, але й про підвищену обізнаність щодо безпеки - співробітники, які двічі думають, перш ніж натиснути кнопку, та беруть на себе відповідальність за захист компанії. У постійно мінливій боротьбі із загрозами соціальної інженерії це поєднання людської пильності та усвідомленої поведінки, що розвивається завдяки постійному навчанню, є одним із найсильніших активів, які може мати організація.

1.3. Аналіз існуючих рішень для симуляції фішингових атак

Сьогодні організації мають у своєму розпорядженні низку інструментів для моделювання фішингових атак у рамках навчання з підвищення обізнаності щодо безпеки. Ці інструменти варіюються від фреймворків з відкритим кодом, що дозволяють налаштування всередині компанії, до комплексних комерційних платформ, що інтегрують навчальний контент та аналітику. У цьому розділі розглядаються три основні рішення для моделювання фішингу - **Gophish**, **KnowBe4** та **Cofense PhishMe** - окреслюються їхні функції, варіанти використання та оцінюються їхні переваги та недоліки з точки зору зручності використання, масштабованості, звітності, інтеграції, гнучкості та ліцензування.

Gophish (фреймворк для симуляції фішингу з відкритим кодом)

Gophish - це популярний набір інструментів з відкритим кодом для створення та управління кампаніями з моделювання фішингу [6]. Випущений у 2017 році, він розроблений для зручності використання як безпековими тестувальниками, так і організаціями для перевірки сприйнятливості співробітників до фішингу в контрольованому середовищі.

Gophish з власним хостингом та є безкоштовним у використанні, що робить його привабливим вибором для організацій з обмеженим бюджетом або тих, хто прагне повного контролю над своєю фішинговою платформою. Основні функції Gophish включають:

- **Веб-керування кампаніями:** Gophish надає веб-інтерфейс, який дозволяє користувачам легко створювати шаблони фішингових електронних листів та цільові сторінки (включаючи імпорт існуючих HTML-листів).
- **Кросплатформність, просте розгортання:** фреймворк постачається у вигляді єдиного бінарного файлу для Windows, macOS та Linux, що дозволяє встановлювати його без складного налаштування.
- **Відстеження результатів у режимі реального часу:** Gophish відстежує відкриття електронних листів, кліки на посилання, надіслані облікові дані та інші дії користувачів у режимі реального часу через свою інформаційну панель.
- **API та автоматизація:** вся функціональність доступна через REST API, що дозволяє інтеграцію з іншими інструментами або скриптами (доступний офіційний клієнт Python API).

Серед **переваг**: повний контроль, гнучкість налаштування (можна налаштовувати шаблони, цільові сторінки та навіть змінювати вихідний код за потреби), підтримка спільноти та прозорість (така модель, керована спільнотою, дозволяє швидкий обмін новими ідеями), а також Gophish зосереджений на фішингу електронної пошти та не перевантажує

користувачів сторонніми функціями, що сприяє простоті його використання для цієї конкретної мети.

Серед **недоліків**: обмеження фішингом електронною поштою, немає вбудованого навчального контенту (користувачі повинні створювати або імпортувати власні шаблони електронних листів та навчальний контент), відсутність розширених функцій (такі як автоматичний зворотний зв'язок/навчання користувачів, розширені аналітичні панелі), технічні витрати та витрати на обслуговування (налаштування SMTP-серверів, застосування оновлень, забезпечення доставки через спам-фільтри; немає підтримки - користувачі повинні звертатися за допомогою до форумів, якщо виникають проблеми).

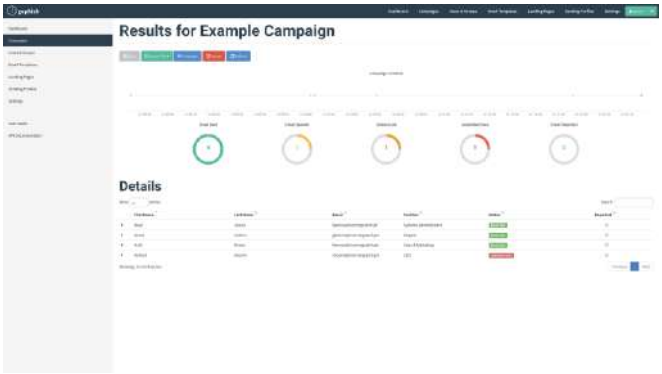


Рисунок 1.1 - Інтерфейс Gophish

KnowBe4 (навчання з безпеки та фішинг-платформа)

KnowBe4 — одна з провідних комерційних платформ у галузі для навчання з питань безпеки в поєднанні з симуляціями фішингу [7]. Запущена у 2010 році, вона зросла до понад 65 000 клієнтів по всьому світу. Платформа KnowBe4 (часто звана платформою Kevin Mitnick Security Awareness Training (KMSAT)) є хмарною та надає інтегрований пакет для тестових кампаній на фішинг, навчання користувачів та аналітики ризиків. Вона відома своєю інтуїтивно зрозумілою веб-консоллю адміністрування та великою бібліотекою контенту, що разом дозволяє навіть нетехнічним адміністраторам запускати ефективні програми захисту від фішингу.

Основні функції KnowBe4 включають:

- **Величезна бібліотека шаблонів та контенту**: KnowBe4 пропонує найбільшу у світі бібліотеку шаблонів фішингових електронних листів (понад 25 000 станом на 2025 рік) та сценаріїв атак, які постійно оновлюються, щоб відображати найновіші реальні фішингові загрози. Шаблони охоплюють поширені теми фішингу (наприклад, шахрайство з боку IT-підтримки, оновлення політики управління персоналом, рахунки-фактури постачальників) і можуть бути локалізовані для різних мов і культур. Крім того, платформа містить широкий спектр навчального контенту - відео, інтерактивні модулі, вікторини, постери та навіть ігри - які навчають користувачів передовим практикам безпеки, що виходять за рамки простого фішингу.
- **Автоматизовані фішингові кампанії та навчальні процеси:** Адміністратори можуть створювати фішингові кампанії, орієнтовані на своїх користувачів, з опціями планування електронних листів, встановлення рівнів складності та автоматичного зарахування користувачів на подальше навчання, якщо вони потраплять на фішинг. Система підтримує «Автоматизовані програми підвищення обізнаності з питань безпеки (ASAP)» - по суті, попередньо розроблені плани кампаній - і може надсилати заплановані електронні листи з нагадуваннями тим, хто не пройшов навчання. Кампанії також можна адаптувати за допомогою функцій на основі штучного інтелекту: KnowBe4 запровадив штучний інтелект для рекомендації та надсилання персоналізованих фішингових електронних листів на основі минулої ефективності та тенденцій загроз кожного користувача.
- **Гейміфікація та залучення користувачів:** Щоб заохотити участь, KnowBe4 включає елементи гейміфікації. Користувачі можуть

заробляти значки та змагатися в таблицях лідерів за повідомлення про фішингові електронні листи або проходження навчання.

- **Інтеграція та адміністрування**: KnowBe4 підтримує інтеграцію Active Directory та SSO для забезпечення доступу користувачів. Платформа також пропонує API для індивідуальної інтеграції та може підключатися до поштових систем організацій (наприклад, додаючи фішингові сервери до білого списку в SPF/DKIM для покращення доставки). Вбудована багатомовна підтримка, а консоль адміністратора та навчальний контент доступні понад 35 мовами, що дозволяє розгортання в багатонаціональних організаціях.
- **Звітність та аналітика в режимі реального часу:** Платформа відстежує статус кожного фішингового електронного листа (доставка, відкриття, натискання тощо) та прогрес у навчанні кожного користувача. Вона надає понад 60 вбудованих звітів та діаграм, включаючи такі показники, як відсоток «схильних до фішингу» (частка користувачів, які натискають на фішингові електронні листи) та тенденцію до зниження цього показника з часом після навчання. Адміністратори можуть переглядати результати за відділами, регіонами чи групами та навіть порівнювати ефективність своєї організації із середніми показниками по галузі.

Серед **переваг**: простота використання та масштабованість (базується на хмарі, масштабується до тисяч користувачів без локальної інфраструктури, а нові оновлення або контент автоматично доставляються постачальником), вичерпний контент та постійні оновлення, інтеграція навчання (користувачам, які натискають на фальшиві фішингові електронні листи, негайно вказуються пропущені червоні прапорці, допомагаючи змінювати поведінку, а не просто тестувати її), аналітика та управління ризиками, зручна інтеграція, а також KnowBe4 пропонує розширену підтримку клієнтів, документацію та посібники з найкращих практик.

Серед **недоліків**: вартість (KnowBe4 – це комерційний сервіс, і вартість може бути значною, особливо для великої кількості користувачів; для невеликих організацій може бути дорогим), модель ліцензування (зазвичай надається на одного користувача на рік, а доступ до певного контенту (наприклад, спеціалізованих навчальних модулів, гейміфікації або оновленої аналітики) може вимагати пакетів вищого рівня), менша гнучкість налаштування. Також, використовуючи KnowBe4, організації довіряють дані співробітників (імена, електронні адреси, результати навчання) сторонньому хмарному сервісу. KnowBe4 наразі не пропонує варіант локального розгортання, що може бути обмежувальним фактором для певних державних середовищ або середовищ із високим рівнем безпеки.

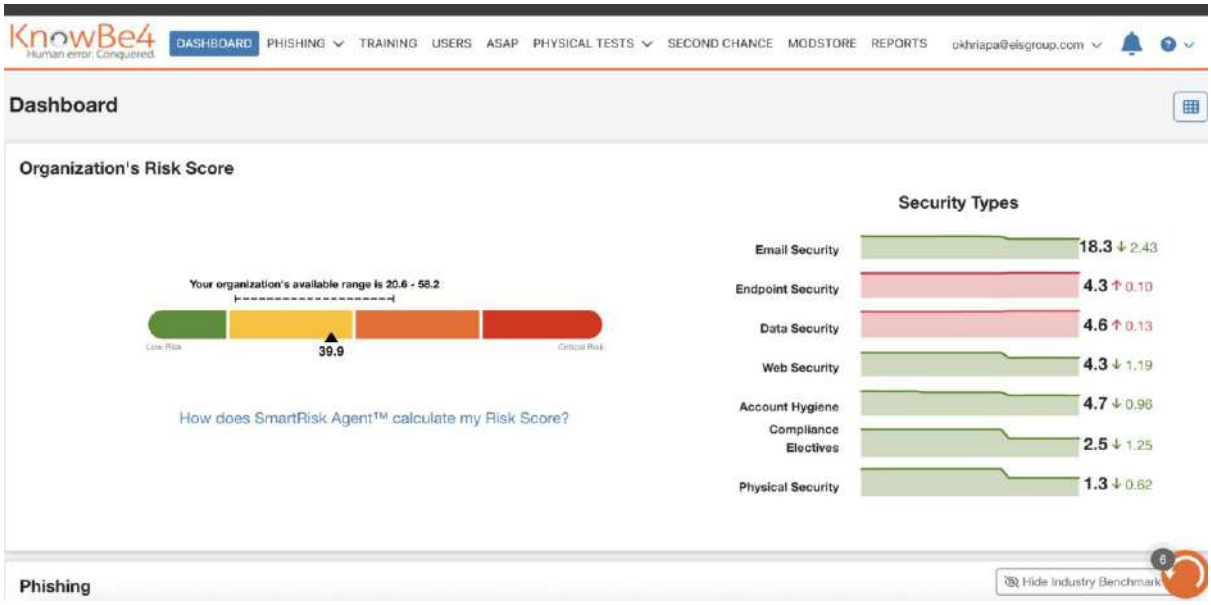


Рисунок 1.2 - Інтерфейс KnowBe4

Cofense PhishMe (платформа для симуляції та реагування на фішинг)

Cofense PhishMe (спочатку відомий просто як PhishMe, перш ніж компанія була перейменована на Cofense) – це ще одне провідне комерційне рішення для симуляції фішингу, з особливим акцентом на навчанні співробітників розпізнавати та повідомляти про спроби фішингу [8]. Cofense PhishMe постачається як хмарна SaaS-платформа. Вона існує на ринку з початку 2010-х років і використовується багатьма великими підприємствами та

державними установами в рамках своїх програм підвищення обізнаності про безпеку та реагування на інциденти.

Основні функції Cofense PhisMe включають:

- **Реалістичні сценарії фішингу з миттєвим зворотним зв'язком:** Ключовим аспектом підходу PhishMe є «навчання: коли користувач потрапляє в симуляцію, система може надати миттєвий зворотний зв’язок та навчання в контексті. Наприклад, після натискання фішингового посилання користувач може бути перенаправлений на навчальну сторінку або короткий модуль, який пояснює червоні прапорці в цьому електронному листі, тим самим перетворюючи невдачу на навчальний момент.
- **Налаштовувані кампанії та контент:** Симуляції можна налаштувати, включивши різні типи вкладень, фальшиві сторінки входу та навіть поетапні електронні листи (для імітації спірфішингу).
- **Звітність та аналітика:** Для вимірювання покращень використовуються такі показники, як час звітування (як швидко користувачі повідомляють про фішинговий електронний лист за допомогою кнопки Cofense Reporter) та стійкість організації (скільки користувачів ігнорують, повідомляють чи потрапляють на електронний лист).
- **Інтеграція реагування на інциденти:** Повідомлені електронні листи – незалежно від того, чи це симуляції, чи справжній фішинг – надходять до системи Cofense Phishing Defense Defense/Triage (якщо організація її має) для аналізу. Крім того, Cofense керує великою краудсорсинговою мережею розвідки про фішингові загрози (з мільйонами звітників у своїй клієнтській базі).
- **Інтеграція з підприємством:** Розгортання базується на хмарі, але є варіанти для приватної хмари або локальних компонентів у середовищах з високим рівнем безпеки. Платформа підтримує кілька

мов як для навчального контенту для кінцевих користувачів, так і для шаблонів фішингу, що робить її придатною для використання в глобальних організаціях.

Серед **переваг**: реалізм симуляцій (часто включають поточні фішингові тактики, що спостерігаються в реальних умовах, завдяки даним розвідки про загрози), інтеграція з реагуванням на інциденти, детальна аналітика, широкий спектр контенту, підтримка клієнтів і навіть професійні послуги – деякі організації обирають керовані програми фішингу Cofense, де експерти Cofense допомагають розробляти та запускати симуляції, перевага в розвідці загроз.

Серед **недоліків**: висока вартість, адміністративні витрати (багатший набір функцій означає, що власники програм повинні витрачати час на налаштування та аналіз), складність системи (додавання до білого списку доменів відправлення Cofense, коригування правил фільтрації спаму), обмежена гнучкість та прив'язка до постачальника.

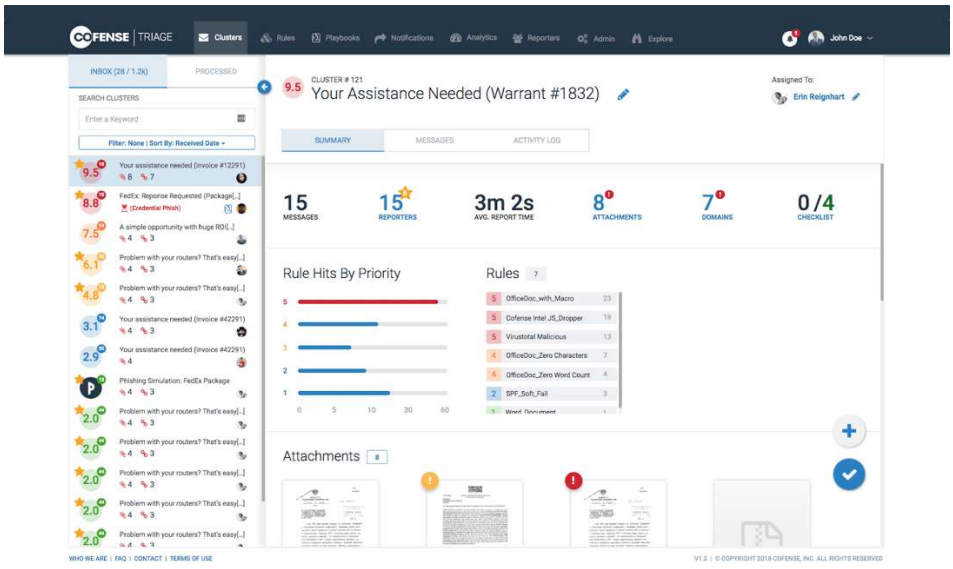


Рисунок 1.3 - Інтерфейс Cofense

Порівняльний аналіз рішень

Критерій	**Gophish**	**KnowBe4 (KMSAT)**	**Cofense PhishMe**
Розгортання / Хостинг	Власний хостинг; єдиний кросплатформний бінарний файл (Windows / macOS / Linux)	Хмарний SaaS (багатокористувацький)	Хмарний SaaS; додаткові компоненти приватної хмари для організацій з високим рівнем безпеки
Ліцензування / Вартість	Безкоштовний, з відкритим вихідним кодом (MIT). Тільки витрати на інфраструктуру.	Річна підписка на користувача; багаторівневі пакети функцій.	Річна підписка на користувача; вища ціна, кілька модулів (Reporter, Triage, Intel).
Вбудований навчальний контент	Немає (користувачі надають свої власні).	Велика бібліотека (понад 25 тис. шаблонів, відео, вікторини, ігри).	Великий набір шаблонів + модулі SCORM для підвищення обізнаності про безпеку; канали з мережі Cofense threat-int.
Зручність використання (UX для адміністратора)	Веб-інтерфейс простий, але налаштування та обслуговування SMTP здійснюється вручну.	Інтуїтивно зрозуміла консоль, конструктор кампаній на основі штучного інтелекту.	Сучасна консоль; найкраще підходить для зрілих команд.
Масштабованість	Залежить від інфраструктури клієнта та пропускної	Хмара масштабується до 10 тис.+ користувачів.	Масштабування від постачальників

	здатності пошти.		
Звітність та аналітика	Статистика кожної кампанії в режимі реального часу; експорт CSV/JSON.	Понад 60 вбудованих звітів, оцінки ризиків, галузевий бенчмаркінг.	Глибока аналітика; детальний аналіз кампаній.
Інтеграція	REST API, вебхуки; все інше вимагає спеціального коду/скриптів.	Синхронізація AD/SSO, відкриті API, кнопка сповіщення про фішинг; конектори SIEM.	Кнопка Reporter Сортування/SIEM; пакети SOAR та ServiceNow.
Гнучкість / Налаштування	Повний контроль вихідного коду; шаблони та цільові сторінки, що підтримують скрипти.	Дозволяються користувацькі шаблони/сторінки, але основні робочі процеси вирішує постачальник.	Користувацькі шаблони; логіка платформи фіксована, але доповнення та канали інформації про загрози можна налаштувати.
Унікальні сильні сторони	• Нульова вартість ліцензії • Повний контроль/локальні дані • Чудово підходить для досліджень та малих команд	• Найбільша бібліотека контенту • Персоналізований фішинг на основі штучного інтелекту • Гейміфікація та просте розгортання	• Цикл реагування на інцидент • Інформація про загрози, отримана з краудсорсингу
Ключові обмеження	• Тільки електронна пошта (без SMS/Vish)	• Вартість підписки • Закритий код; неможливість	• Вартість підписки • Адміністративні витрати/складність • Закритий код; неможливість

	• Без вбудованого навчання • Самостійне обслуговування та масштабування	налаштування робочих процесів	налаштування робочих процесів

Підсумовуючи, моделювання фішингових атак можна досягти за допомогою різних підходів. Фреймворки з відкритим кодом, такі як Gophish, пропонують практичний тестовий майданчик, який можна налаштовувати, ідеальний для технічно підкованих команд або обмежених бюджетів, надаючи детальний контроль над фішинговими симуляціями. Тим часом комерційні платформи, такі як KnowBe4 та Cofense PhishMe, пропонують комплексні рішення, що масштабуються та попередньо оснащені знаннями галузевих експертів, включаючи багатий контент та можливості інтеграції, що робить їх придатними для широкого розгортання в організаціях.

2. РОЗРОБКА ДОДАТКУ ДЛЯ СИМУЛЯЦІЇ ФІШИНГОВИХ АТАК

2.1. Постановка завдання

Метою цієї роботи є розробка веб-додатку для моделювання фішингових атак у корпоративному середовищі. Додаток призначений для навчання та підвищення обізнаності, що дозволяє організаціям безпечно перевіряти вразливість своїх співробітників до спроб фішингу та аналізувати помилки, які роблять користувачі («клікери»), коли вони потрапляють на такі атаки. Контрольованим чином система надсилатиме фальшиві (але реалістичні) фішингові електронні листи користувачам, відстежуватиме їхню взаємодію та генеруватиме аналітичні дані про поведінку користувачів. У цьому розділі визначено основні функціональні вимоги до додатку та описано, як моделюються сценарії фішингових атак, надаючи чітке уявлення про передбачувані можливості системи.

Основні функціональні вимоги:

- **Керування користувачами:** Система повинна підтримувати надійне керування користувачами для співробітників організації. Адміністратори повинні мати можливість створювати та керувати обліковими записами користувачів, включаючи такі дані, як ім'я та адреса електронної пошти, а також організовувати користувачів у групи для цільових кампаній. Очікується контроль доступу на основі ролей, щонайменше з роллю адміністратора та роллю стандартного користувача. Це гарантує, що лише уповноважений персонал може розробляти фішингові кампанії та отримувати доступ до конфіденційних результатів, тоді як кожного співробітника, на якого спрямовано тестування, можна однозначно ідентифікувати в результатах.
- **Налаштування фішингової кампанії:** Додаток повинен надавати інтерфейс для розробки та виконання кампаній з імітації

фішингових атак. Це включає можливість для адміністратора визначати кампанію, вибираючи сценарій фішингу (або шаблон), вибираючи цільових одержувачів (окремих осіб або групи користувачів) та плануючи кампанію (запуск негайно або у заплановану дату/час). Кожна конфігурація кампанії об'єднуватиме контент, що надсилається, список одержувачів та параметри, такі як тривалість кампанії. Система повинна обробляти автоматичне надсилання фішингових електронних листів зазначеним користувачам та дозволяти запускати кілька кампаній протягом певного часу (наприклад, для проведення щомісячних тестів на фішинг). Функції управління кампаніями повинні включати активацію/деактивацію кампаній та огляд поточних або завершених кампаній з їх статусом та основною статистикою (наприклад, скільки надіслано електронних листів та скільки користувачів натиснули).

- **Налаштування шаблонів електронних листів:** основною функцією є можливість створювати та налаштовувати шаблони фішингових електронних листів. Додаток пропонуватиме бібліотеку попередньо розроблених шаблонів фішингових електронних листів, які імітують поширені стратегії фішингу (такі як повідомлення про скидання фальшивого пароля, повідомлення про безпеку або корпоративні оголошення). Адміністратори повинні мати можливість редагувати ці шаблони або створювати нові з нуля, налаштовуючи такі елементи, як тема електронного листа, вміст тіла, ім'я/адреса відправника (для імітації довірених осіб) та будь-які вбудовані посилання або вкладення. Слід підтримувати заповнювачі персоналізації (наприклад, %USER_NAME%) для вставки імені одержувача або інших даних в електронні листи для реалізму. Дозволяючи широке налаштування шаблонів, система гарантує, що імітовані фішингові електронні листи виглядають автентичними та

різноманітними, підвищуючи ефективність навчання. Шаблони можуть варіюватися від загальних масових фішингових листів до вузько цільових повідомлень-фішингових атак, що охоплюють різні рівні складності та наративи атак, відповідно до цілей навчання організації.

- **Реєстрація даних та аналітика:** Додаток повинен реєструвати всі відповідні взаємодії користувачів та надавати аналітику для інтерпретації даних. Кожен електронний лист, надісланий у рамках кампанії, слід відстежувати, щоб побачити, чи відкрив його користувач, чи натиснув він на будь-яке фішингове посилання, чи завантажив або відкрив вкладення. Система реєструє ці події з мітками часу та пов'язує їх з певними користувачами та кампаніями. Збираючи ці дані, додаток може обчислювати ключові показники, такі як коефіцієнт кліків (відсоток користувачів, які натиснули на фішингове посилання), коефіцієнт відкриття вкладень та відсоток користувачів, «схильних до фішингу» (тих, хто певним чином попався на атаку). Ця аналітика допомагає визначити, які користувачі або групи є найбільш вразливими та які типи атак є найефективнішими. Крім того, якщо політика організації заохочує користувачів повідомляти про підозрілі електронні листи, система може реєструвати випадки, коли користувачі повідомляють про змодельований фішинг (наприклад, за допомогою спеціальної кнопки повідомлення або ручного зворотного зв'язку) як позитивний результат. Усі зібрані дані надійно зберігаються для аналізу та представлені в узагальненому вигляді для збереження конфіденційності, одночасно висвітлюючи поведінкові тенденції. Ця детальна можливість фіксації та аналізу подій має вирішальне значення для оцінки ефективності зусиль щодо виявлення фішингу.
- **Звітність:** Спираючись на зареєстровані дані, програма повинна надавати комплексні функції звітності для передачі результатів

фішингових симуляцій зацікавленим сторонам. Це включає створення зведених звітів для кожної кампанії, що показують загальну статистику, таку як кількість користувачів, які були атаковані фішинговими атаками (і на якому етапі), які посилання були натиснуті та які відділи мали найвищі показники кліків. Модуль звітності повинен представляти дані в доступному форматі, використовуючи таблиці або діаграми (наприклад, стовпчикові діаграми показників кліків для відділів, лінії трендів для кількох кампаній тощо) для виділення закономірностей. Важливими показниками, які слід включити, є загальна кількість надісланих електронних листів, відкритих та натиснутих, кількість користувачів, які ввели облікові дані, та час, необхідний для відповіді користувачів. Звіти також повинні ідентифікувати повторних клікерів (співробітників, які постійно потрапляють під атаки), оскільки їм може знадобитися додаткове навчання. Крім того, система повинна дозволяти експортувати або завантажувати звіти (наприклад, у форматі PDF або CSV), щоб команда безпеки могла ділитися результатами з керівництвом або інтегрувати дані в ширші панелі безпеки. Надаючи чіткі та детальні звіти, програма дозволяє організації вимірювати ефективність своєї програми боротьби з фішингом з часом та документувати покращення або ризики, що залишилися.

Моделювання сценаріїв фішингових атак:

- **Типи та шаблони атак:** Усі симуляції фішингових атак у цій системі базуються на електронній пошті та зосереджені на найпоширенішому векторі фішингу. Зокрема, однією з категорій є

атаки зі збором облікових даних, коли фішинговий електронний лист містить посилання на фальшиву сторінку входу. У такому сценарії шаблон може видавати себе за довірений сервіс (наприклад, поштовий портал компанії або відомий веб-сервіс), спонукаючи користувача натиснути на посилання та повторно ввести свій пароль. Інша категорія – це атаки зі шкідливими вкладеннями, коли електронний лист містить вкладення (наприклад, PDF-файл або документ Office), яке користувач спокушається відкрити. Хоча вкладення в симуляції нешкідливі, сам акт їх відкриття свідчить про готовність користувача ігнорувати попередження безпеки. Подальшим варіантом можуть бути оманливі посилання, вбудовані безпосередньо в тіло електронного листа (класичне фішингове посилання, яке не веде на сторінку входу, а, можливо, на завантаження шкідливого програмного забезпечення або цільову сторінку з попередженням). Бібліотека шаблонів системи відображатиме ці типи атак: наприклад, шаблони для електронних листів зі збором облікових даних будуть поєднані з відповідною фальшивою веб-сторінкою для входу, тоді як шаблони для приманок на основі вкладень включатимуть файл-фіктив. Адміністратори можуть вибрати тип атаки під час створення кампанії, і програма використовуватиме відповідний шаблон і метод відстеження для цього сценарію. (Інші форми соціальної інженерії, такі як SMS-фішинг або голосовий фішинг, виходять за рамки цього веб-інструменту)

- **Механізм відстеження відповідей:** Коли надсилається фішинговий електронний лист, система відстежує взаємодію користувача з ним. У всіх випадках кожна відповідна дія реєструється в базі даних системи: відкриття електронного листа, клік за посиланням, клік за вкладенням, надсилання форми тощо, кожна з яких пов’язана з конкретним користувачем та кампанією. Крім того, система може

відстежувати коригувальні дії: якщо користувач використовує процес звітності компанії, щоб позначити електронний лист як підозрілий, ця подія може бути зареєстрована як позитивна відповідь (користувач розпізнав загрозу). Деталізація цих даних відповіді дозволяє програмі аналізувати, наскільки далеко кожен користувач зайшов у фішинговій пастці — чи він просто відкрив електронний лист, клацнув посилання чи ввів облікові дані. Додаток використовуватиме ці дані для надання зворотного зв'язку в режимі реального часу на панелі інструментів кампанії та для створення аналітики та звітів після кампанії.

2.2. Вибір технологій та інструментів

Серверну частину створено за допомогою **Python 3** і веб-фреймворку **FastAPI**. Python було обрано основною мовою програмування завдяки його читабельності та багатій екосистемі бібліотек, що пришвидшує розробку. FastAPI є одним із найшвидших веб-фреймворків Python, часто перевершуючи більш усталені фреймворки, такі як Flask або Django, за пропускною здатністю обробки запитів. Асинхронні можливості FastAPI дозволяють програмі обробляти кілька запитів одночасно без значного зниження продуктивності під високим навантаженням, що покращує масштабованість.

Окрім самого FastAPI, на **серверній частині** було використано кілька допоміжних бібліотек Python. Серед основних:

- **SQLAlchemy:** Додаток використовує SQLAlchemy як об'єктно-реляційне відображення (ORM) для взаємодії з базою даних MySQL
- **Authlib**: Для реалізації автентифікації. У цьому проєкті Authlib забезпечує функцію «Вхід за допомогою Google» (інтеграція з Google OAuth 2.0)
- **JSON веб-токени (JWT):** Після входу користувача, бекенд видає JWT, який фронтенд включає до наступних викликів API.

- **APScheduler:** Бібліотека Advanced Python Scheduler (APScheduler) використовується для планування та керування фоновими завданнями. У контексті цього проекту APScheduler дозволяє використовувати такі функції, як планування надсилання фішингових електронних листів у певний час.

Фронтенд застосунку реалізовано за допомогою **React**. React було обрано через його ефективність у створенні динамічних інтерактивних інтерфейсів користувача та потужну підтримку спільноти. Однією з переваг React є його компонентна архітектура, яка сприяє повторному використанню коду та зручності його обслуговування. Інтерфейс користувача розділений на модульні компоненти, які можна розробляти та тестувати окремо, а потім повторно використовувати в додатку.

Під час створення **React фронтенду** було використано кілька бібліотек та інструментів, основними з яких є:

- **Tailwind CSS**: Для стилізації проект використовує Tailwind CSS (CSS-фреймворк). Підхід Tailwind мінімізує розмір таблиці стилів, видаляючи невикористовувані стилі у продакшені, та уникає необхідності використовувати багато користувацьких CSS-файлів, тим самим зменшуючи витрати на обслуговування.
- **React Router:** Популярність React Router та інтеграція з моделлю компонентів React зробили його очевидним вибором – він вважається «невід'ємною» частиною екосистеми React для управління навігацією у застосунках.
- **Recharts:** Recharts — це бібліотека діаграм з відкритим кодом, побудована на React та D3, яка дозволяє створювати інтерактивні діаграми. Її було обрано за баланс потужності та простоти: вона підтримує різноманітні типи діаграм (стовпчикові, лінійні, кругові тощо), яких достатньо для аналітики в цій програмі, та легко інтегрується зі станом та властивостями React.

- **Slate.js:** Для підтримки можливості створення та редагування фішингових електронних листів використовується фреймворк Slate.js для створення власного редактора тексту в React. Slate був обраний, оскільки це фреймворк із широкими можливостями налаштування.

Для постійного зберігання даних проект використовує реляційну базу даних **MySQL**. MySQL було обрано як систему керування базами даних завдяки її перевіреній надійності, продуктивності та знайомості в корпоративних середовищах.

Критичним компонентом платформи для симуляції фішингу є можливість надсилання електронних листів, що імітують спроби фішингу. Для цього проект інтегрується з **SMTP2GO**, хмарним сервісом доставки електронної пошти. SMTP2GO було обрано, оскільки він пропонує надійний, масштабований та безпечний сервіс SMTP без необхідності обслуговування власних поштових серверів. SMTP2GO вигідно відрізняється від інших SMTP-сервісів за вартістю та часом налаштування.

2.3. Архітектура додатку

AntiHook розроблено з розділеною клієнт-серверною архітектурою, що складається з фронтенду на основі React та бекенду на основі FastAPI, з базою даних MySQL для збереження даних. Фронтенд та бекенд - це окремі компоненти, які взаємодіють через RESTful HTTP-інтерфейс. Таке розділення покращує портативність інтерфейсу користувача на різних платформах та масштабованість сервера, розділяючи завдання. Весь обмін даними між фронтендом та бекендом відбувається у форматі JSON через API, що забезпечує чітке розділення між рівнями представлення та логіки. Загалом, високорівнева архітектура відповідає трирівневій моделі (клієнт, сервер, база даних) з чітко визначеними інтерфейсами між кожним рівнем.

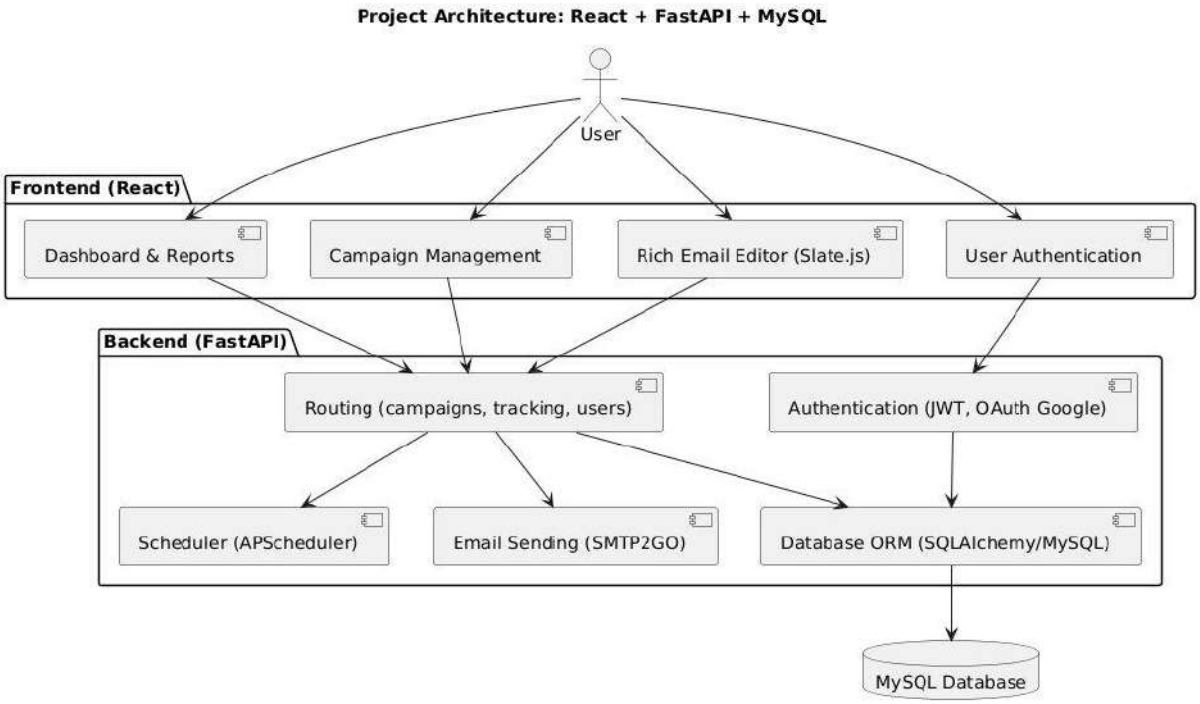


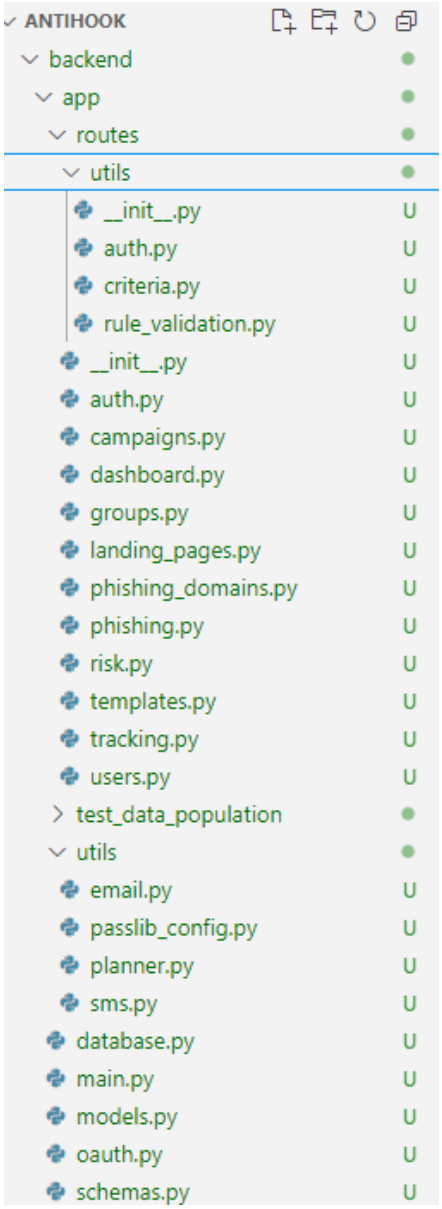
Рисунок 2.1 - Архітектура додатку

Кодова база бекенду організована в директорії *app/* з підпапками, такими як *routes* (для кінцевих точок API) та *utils* (для фонових завдань та допоміжних програм). Ключові модулі маршрутизації API включають:

- **Автентифікація (*auth.py*)**: Обробляє вхід/вихід користувачів, виклики Google OAuth та видачу токенів JWT. Після успішного входу серверна частина видає JSON Web Token (JWT), який клієнт використовує для автентифікованих запитів.
- **Керування кампаніями (phishing.py та пов'язані сервіси):** керує кампаніями симуляції фішингу. Це включає кінцеві точки для створення кампаній (з такими параметрами, як шаблон електронної пошти, цільова група користувачів, розклад), запуску або планування кампаній та відстеження стану кампанії. Коли кампанію заплановано, серверна частина генерує окремі завдання надсилання електронних листів для кожного цільового одержувача. Основна логіка тут використовує внутрішні допоміжні функції (наприклад, модуль планувальника) для заповнення бази даних записами запланованого

надсилання електронних листів та підготовки токенів відстеження для кожного електронного листа. campaigns.py також надає статистичні дані до фронтенду.

- **Керування користувачами та групами (*users.py, groups.py*):** Забезпечує операції CRUD для керування обліковими записами користувачів та групами в організації. Адміністратори можуть створювати або імпортувати користувачів, призначати їх до груп та керувати ролями (наприклад, адміністратор проти звичайного користувача). Групи можуть бути статичними або динамічними (розумні групи, визначені правилами), що дозволяє проводити цільові фішингові кампанії. Цей модуль гарантує, що лише авторизовані адміністратори можуть виконувати конфіденційні операції, використовуючи безпеку FastAPI на основі залежностей (наприклад, *Depends(get_current_user)* з перевірками ролей) для контролю доступу.



• **Відстеження ризиків та навчання:** Хоча це не окремий модуль, серверна частина реалізує логіку для запису результатів фішингових тестів та коригування оцінок ризику користувача. Наприклад, коли користувач потрапляє на симульований фішинг (наприклад, клацання посилання або введення облікових даних на фальшивій сторінці входу), подія реєструється (як запис PhishingEvent), а оцінка ризику користувача збільшується. З часом накопичені події дозволяють системі обчислювати аналітику (наприклад, як змінюється оцінка ризику користувача або які відділи найбільш вразливі), яку серверна частина надає через кінцеві точки API аналітики фронтенду.

Рисунок 2.2 - Структура проєкту (серверна частина)

Сервер FastAPI використовує моделі **Pydantic** для перевірки даних і передачі між рівнями. Бекенд дотримується типових принципів **RESTful-проектування**, з чітким розділенням методів HTTP та кінцевих точок для різних ресурсів (наприклад, GET /api/users/ для списку користувачів, POST /api/campaigns/ для створення кампанії тощо).

Помітним аспектом бекенду є використання **фонового планування** для певних завдань. Зокрема, після того, як фішингові електронні листи плануються в базі даних (кожен з часом надсилання), фоновий виконавець завдань періодично перевіряє наявність запланованих електронних листів та надсилає їх. Модуль *app.utils.planner* проекту визначає планувальник, який працює асинхронно разом із додатком FastAPI. Він використовує

AsyncIOScheduler (варіант APScheduler), щоб додати повторюване завдання, яке запускається для відправлення будь-яких електронних листів, що очікують на відправку, час надсилання яких настав. Ця конструкція відокремлює надсилання електронних листів від циклу запит/відповідь – фронтенд може планувати кампанію через виклик API, а фактична доставка електронних листів відбувається у фоновому режимі. APScheduler працює доти, доки працює додаток, і забезпечує виконання завдань за розкладом.

Кожне виконання завдання планувальника запитуватиме базу даних на наявність будь-яких електронних листів зі статусом «pending» та запланованою міткою часу <= now, надсилатиме ці листи (за допомогою функції утиліти електронної пошти *app.utils.email*, пов’язаної з SMTP2GO) та оновлюватиме їхній статус (на «sent» або «failed») разом із записом часу надсилання. Цей підхід є надійним: навіть якщо сервер перезавантажиться, завдання, що очікують на розгляд, залишатимуться в базі даних і будуть оброблені під час наступного запуску планувальника.

Фронтенд реалізовано як односторінковий додаток (SPA) з використанням бібліотеки **React 18+**. Будучи SPA, програма не перезавантажує всю сторінку під час навігації; натомість він отримує дані з бекенду через виклики API та відповідно оновлює представлення (що дозволяє створювати кілька переглядів або «сторінок» в SPA, таких як Dashboard, Campaigns, User Management тощо, без повного перезавантаження сторінки).

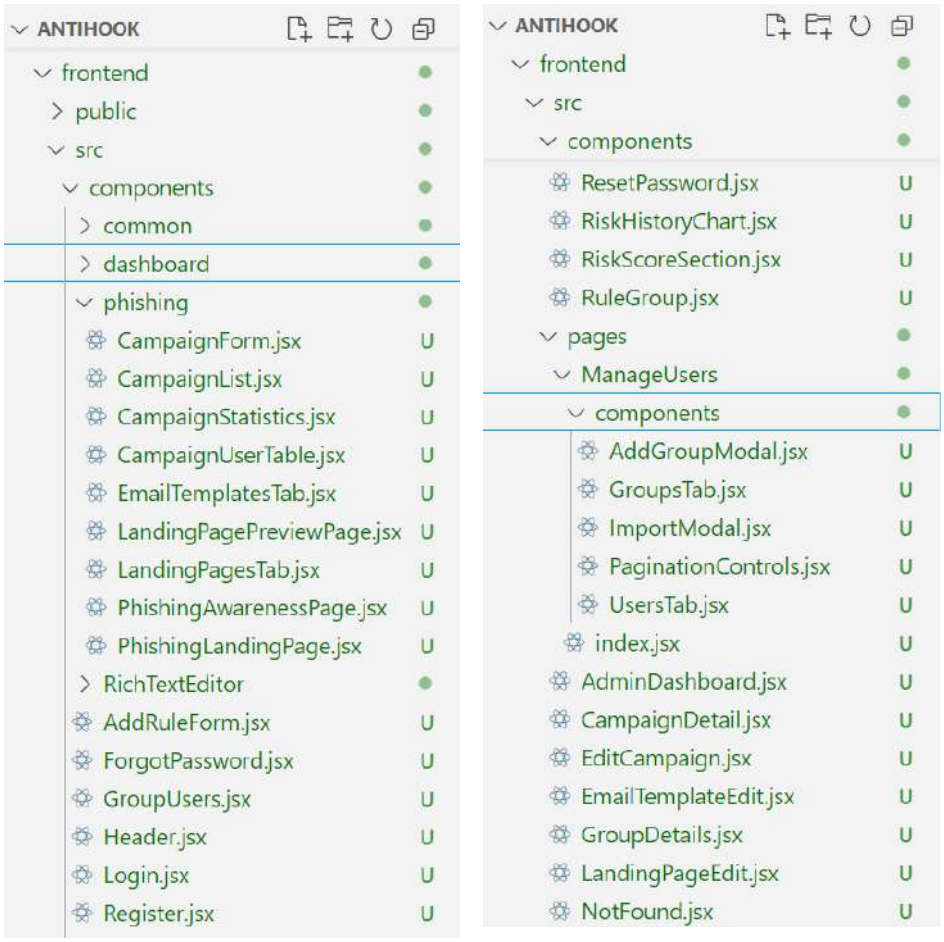
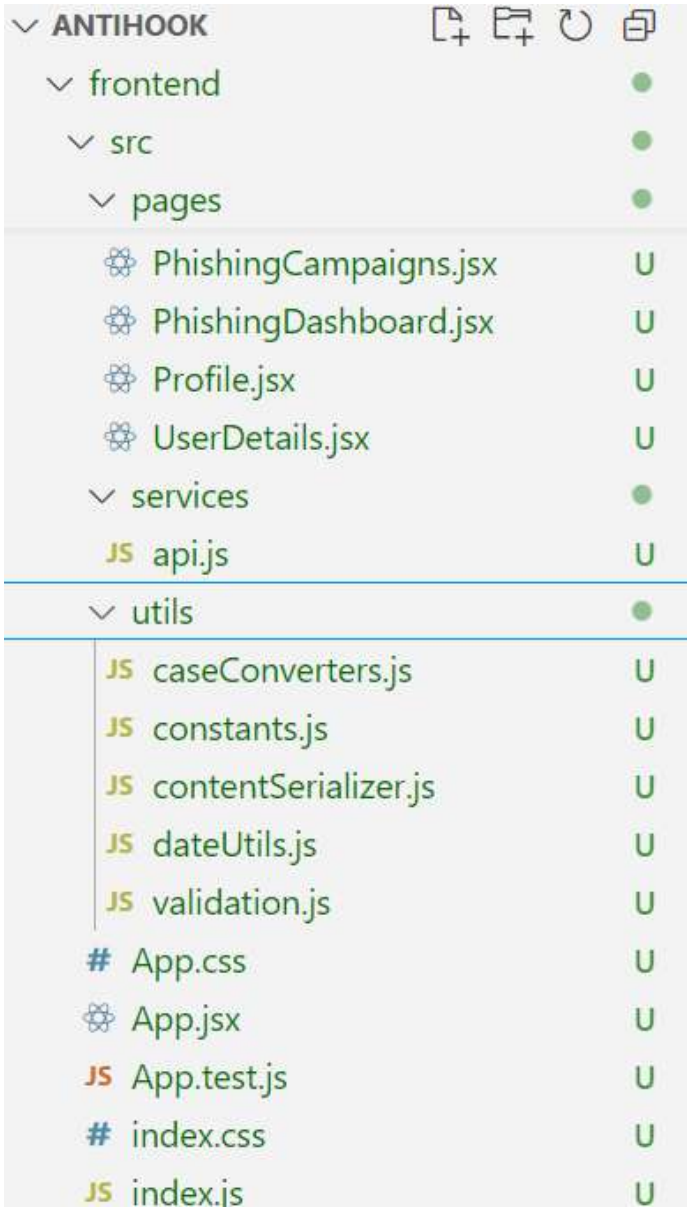


Рисунок 2.3-2.4 - Структура проєкту (фронтенд)



Інтерфейс користувача організовано в компоненти багаторазового використання відповідно до практик React. Наприклад, є компоненти для переліку кампаній, компоненти для відображення оцінки ризику користувача, форми для створення/редагування кампаній тощо. Варто зазначити групу користувацьких компонентів – це редактор форматованого тексту, який використовується для створення шаблонів фішингових електронних листів та цільових сторінок.

Рисунок 2.5 - Структура проєкту (фронтенд)

Ще одним важливим аспектом фронтенду є інтерфейси інформаційної панелі та аналітики. AntiHook надає аналітику того, як користувачі реагують на фішингові симуляції (хто натиснув, хто повідомив про електронний лист, хто його проігнорував тощо) та загальні тенденції організаційних ризиків. Вони представлені у вигляді діаграм і таблиць.

Фронтенд також реалізує обробку автентифікації на стороні клієнта. Під час початкового завантаження, якщо користувач не ввійшов у систему, йому відображається екран входу. Після автентифікації отриманий токен JWT зберігається та використовується для наступних викликів API. React

Router захищає певні маршрути (наприклад, /dashboard або /campaigns), щоб лише автентифіковані користувачі (з дійсним токеном) могли отримати до них доступ – в іншому випадку відбувається перенаправлення на вхід.

Дизайн (з Tailwind) гарантує, що інтерфейс є адаптивним та доступним, адаптуючись до різних розмірів екрана, щоб адміністратори могли зручно відстежувати результати навіть на планшеті або ноутбуці.

Під час розробки фронтенд працює на локальному сервері (порт 3000) з перезавантаженням у реальному часі, але у продакшені процес збірки компілює додаток React у статичний HTML/CSS/JS, який може обслуговуватися через простий веб-сервер або через проміжне програмне забезпечення для статичних файлів FastAPI.

Огляд схеми БД: База даних використовує схему, яка фіксує користувачів, фішингові кампанії та пов'язані сутності. На вищому рівні основні таблиці/сутності в схемі включають:

- **Користувачі (users)**: Зберігає облікові записи користувачів (співробітників або цілей фішингових симуляцій). Поля включають ідентифікатор користувача, ім'я, електронну пошту, хешований пароль (якщо використовується локальна автентифікація), роль/дозвіл (наприклад, адміністратор або звичайний користувач) та зв'язок з організацією. Користувачі мають зв'язки з іншими таблицями: кожен користувач може належати до однієї організації та бути частиною кількох груп.
- **Організації (organisation)**: Представляє окремі організації, які використовують платформу. Це дозволяє розділяти дані, якщо система використовується кількома компаніями. Містить такі поля, як ідентифікатор організації та назва. Користувачі та кампанії пов'язані з організацією.

- **Групи (groups)**: Дозволяє групувати користувачів (наприклад, за відділом або рівнем ризику). Кожна група має ідентифікатор, назву, та може мати визначені JSON правила, за якими вона формується (бути статичною або динамічною). Таблиця об'єднання *user_groups* пов'язує користувачів із групами (зв'язок «багато до багатьох» між користувачами та групами). Групи використовуються під час запуску кампаній для таргетування певних підмножин користувачів.

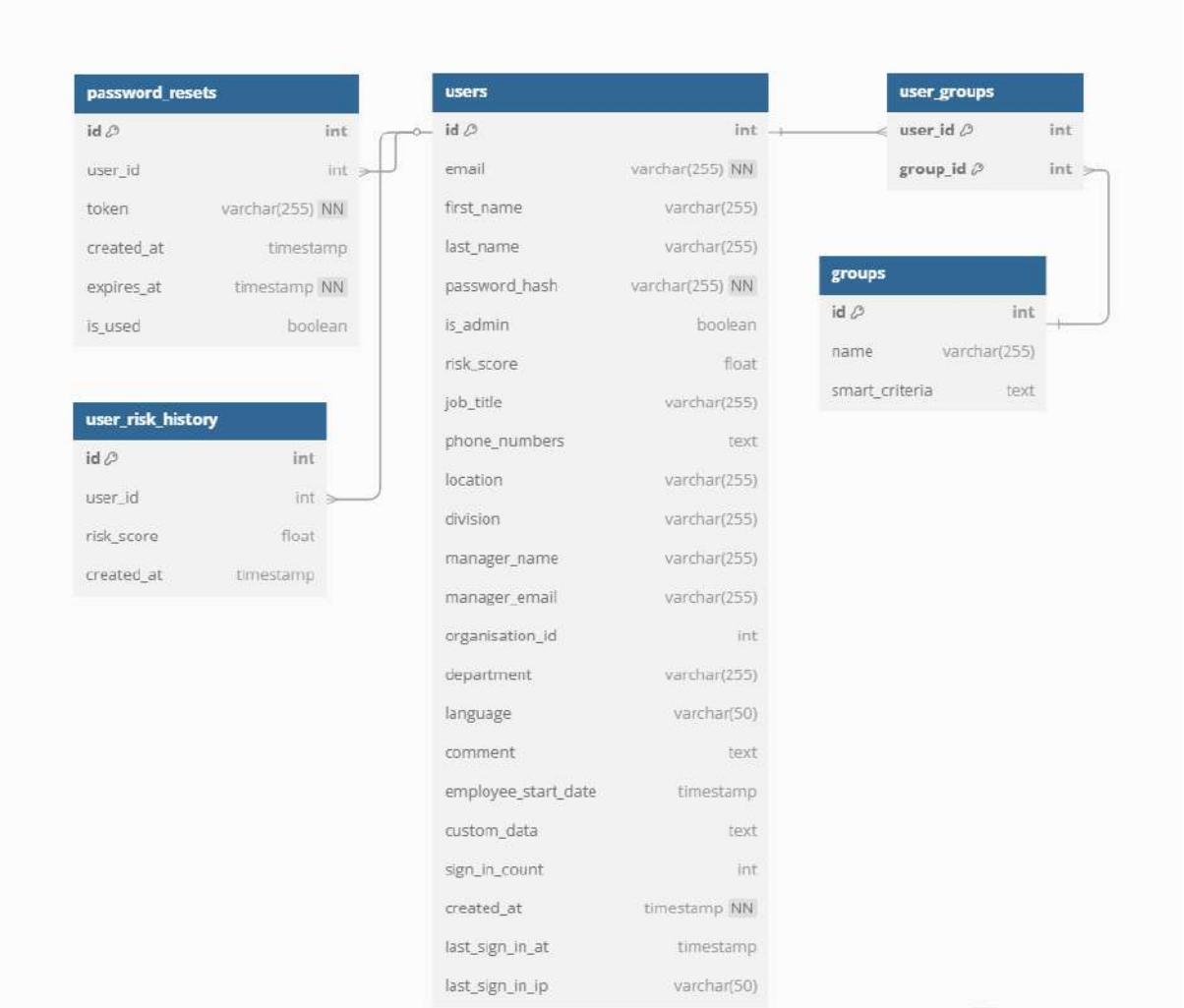


Рисунок 2.6 - Структура бази даних (Користувач і Група)

- **Кампанії (campaigns)**: Представляє кампанію симуляції фішингу. Ключові поля включають ідентифікатор кампанії, назву, опис, дату створення, дату початку (або запланований час запуску), статус

(чернетка, заплановано, у процесі, завершено) та ідентифікатор організації. Кампанія пов’язана з одним або кількома шаблонами фішингових електронних листів та з набором цільових користувачів (через групи). Кампанії також можуть мати налаштування, такі як надсилання всіх електронних листів одночасно чи поступово.

- **Шаблони електронних листів (email_templates):** Зберігає шаблони фішингових електронних листів, що використовуються в кампаніях. Поля включають ідентифікатор шаблону, ім’я, тему, тіло (HTML-контент) і метадані, такі як дата створення/останнього редагування або категорія. Якщо використовуються вкладення, може зберігатися шлях до файлу або посилання. Шаблон електронного листа також може посилатися на цільову сторінку (підроблену URL-адресу сторінки входу), на яку перенаправляються користувачі. Вміст шаблону може містити заповнювачі (наприклад, {USER_NAME}), які заповнюються для кожного одержувача під час надсилання.

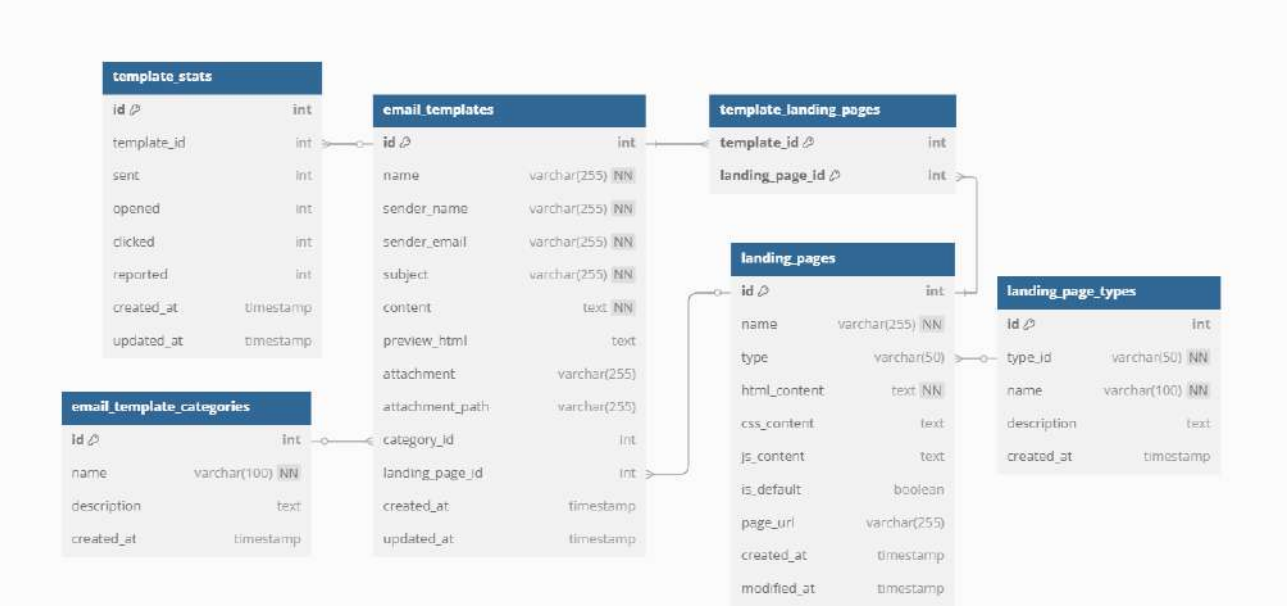


Рисунок 2.7 - Структура бази даних (Кампанії, Шаблони листів)

- **Фішингові події (phishing_events), надсилання електронних листів (email_sends)**: реєструє результати фішингових електронних листів, надісланих користувачам. Кожен запис *EmailSend*

посилається на певну кампанію, певного користувача та певний шаблон електронного листа (в одній кампанії може використовуватися кілька шаблонів). Він також зберігає унікальний токен відстеження (ідентифікатор) для електронного листа (для співвіднесення відкриттів електронних листів через піксель відстеження та кліків через унікальні URL-адреси). Якщо кампанія орієнтована на 100 користувачів, під час запуску кампанії в цю таблицю буде вставлено 100 записів (по одному на кожен надісланий електронний лист). Поля *EmailSend* включають id, user_id, template_id, campaign_id, sent_at, status (доставлено, заплановано “pending”, помилка при відправленні), scheduled_date, tracking_token. *PhishingEvent* відстежує, чи відкрив користувач електронний лист, натиснув на посилання чи надіслав дані, а також позначки часу для кожної дії. Поля *PhishingEvent* включають id, send_id (зв’язок із *EmailSend*), page_id (якщо подія пов’язана із фішинговою сторінкою, посилання на яку було в листі), event_type (відкриття листа, перехід за посиланням, введення даних на фішинговій сторінці, тощо), event_ts (часова мітка події), event_metadata (може містити додаткову інформацію, наприклад, тип пристрою та браузер, з якого надійшла фішингова подія).

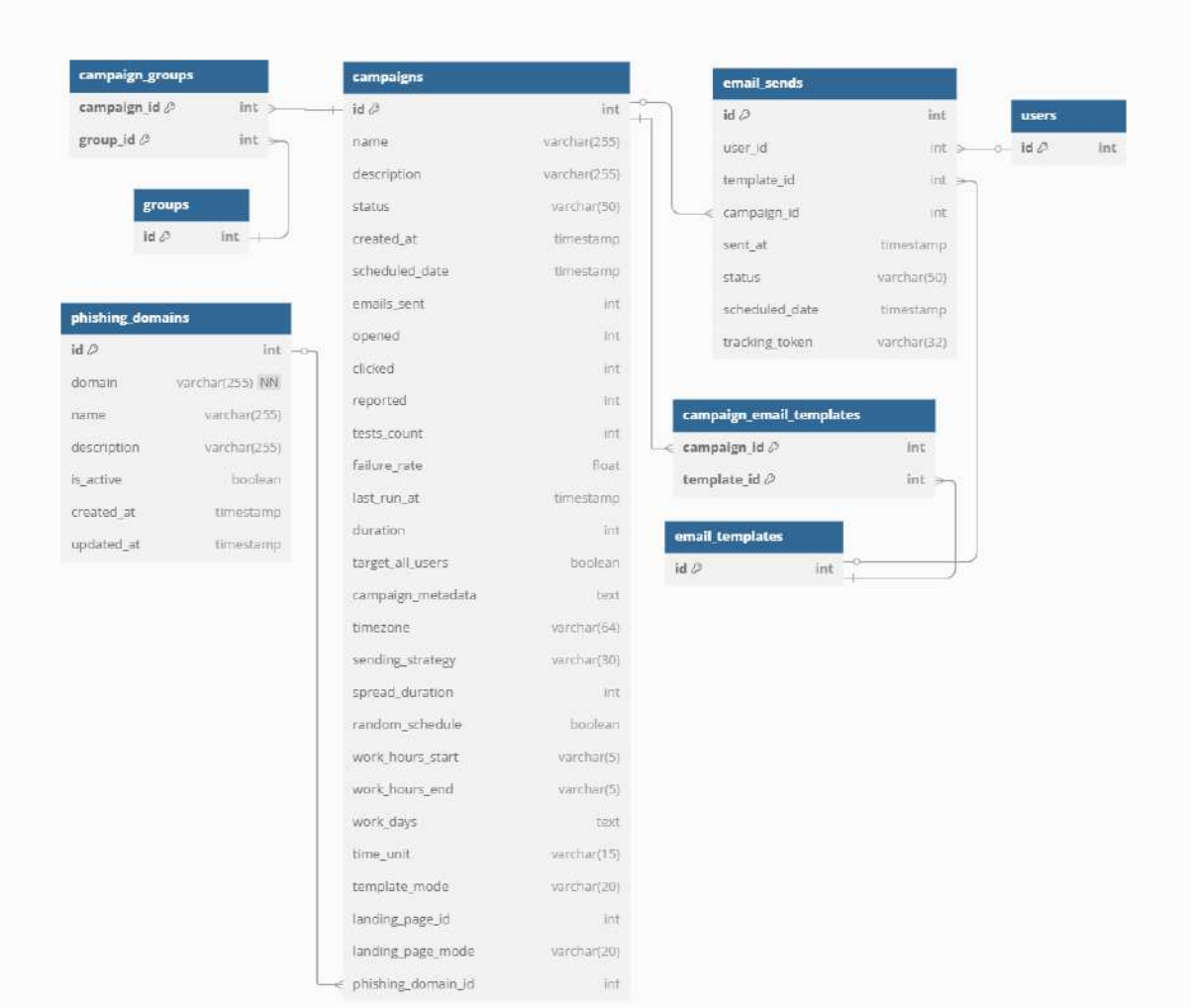


Рисунок 2.8 - Структура бази даних (Кампанії, Посилання листів)

- **Ризики (user_risk_history, organisation_risk_history, group_risk_history):** Агрегують рівень ризику користувачів/груп/організацій. Реалізовані у вигляді таблиць, де кожен запис представляє рівень ризику користувача (групи/організації) та дату створення запису. Оцінку ризику можна обчислити на основі історії фішингових подій цього користувача (наприклад, більше попадань на фішингові листи = вищий ризик). Ці таблиці дозволяють виконувати швидкі запити, такі як «показати всіх користувачів з високим рівнем ризику», або зберігати статистичну історичну інформацію про користувача/групу/організацію для інформаційних панелей.

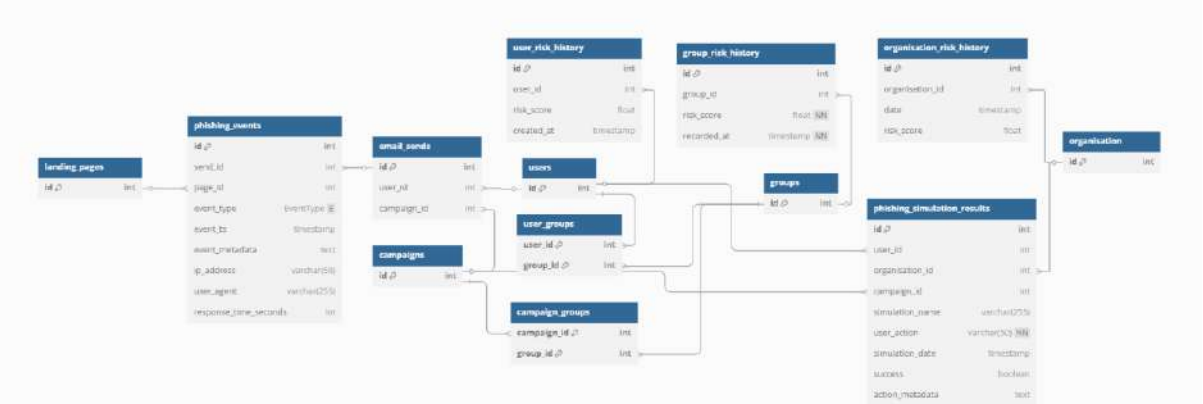


Рисунок 2.9 - Структура бази даних (зберігання результатів та статистики)

Ці таблиці з'єднані зовнішніми ключами, що встановлюють реляційну цілісність (наприклад, Кампанії мають зовнішній ключ org_id для Організацій; EmailSends посилаються на campaign_id, user_id тощо).

Операції з базою даних у застосунку проходять через сесії SQLAlchemy ORM. *models.py* визначає моделі SQLAlchemy, що відповідають наведеним вище таблицям, використовуючи класи та зв'язки. Використовуючи ORM, бекенд може виконувати такі дії, як *db_session.query(User).filter(User.email == x)*. У більшості випадків це абстрагує необроблений SQL, хоча складні аналітичні запити все ще можуть використовувати інтерфейс запитів SQLAlchemy для виконання об'єднань та агрегацій.

Концептуальна модель (користувачі, групи, кампанії, шаблони, події тощо) лежить в основі схеми, а підтримка чітких зв'язків між цими сутностями гарантує, що програма може ефективно запитувати необхідну інформацію (наприклад, «скільки користувачів натиснули на посилання в кампанії X?» або «перерахувати всі кампанії, які певний користувач не пройшов»).

Рисунок 2.10 - Структура бази даних (повна)

2.4. Реалізація основного функціоналу

2.4.1. Реалізація модуля створення фішингових сценаріїв

Цей модуль дозволяє адміністратору розробляти шаблони фішингових електронних листів, налаштовувати відповідні цільові сторінки фішингу, вибирати цільові групи користувачів, налаштовувати параметри кампанії (наприклад, розклад).

На рисунках 2.11-2.12 зображено інтерфейс редагування шаблону, який включає панель інструментів форматування (для виділення жирним шрифтом, курсивом, посиланнями тощо) та область попереднього перегляду.

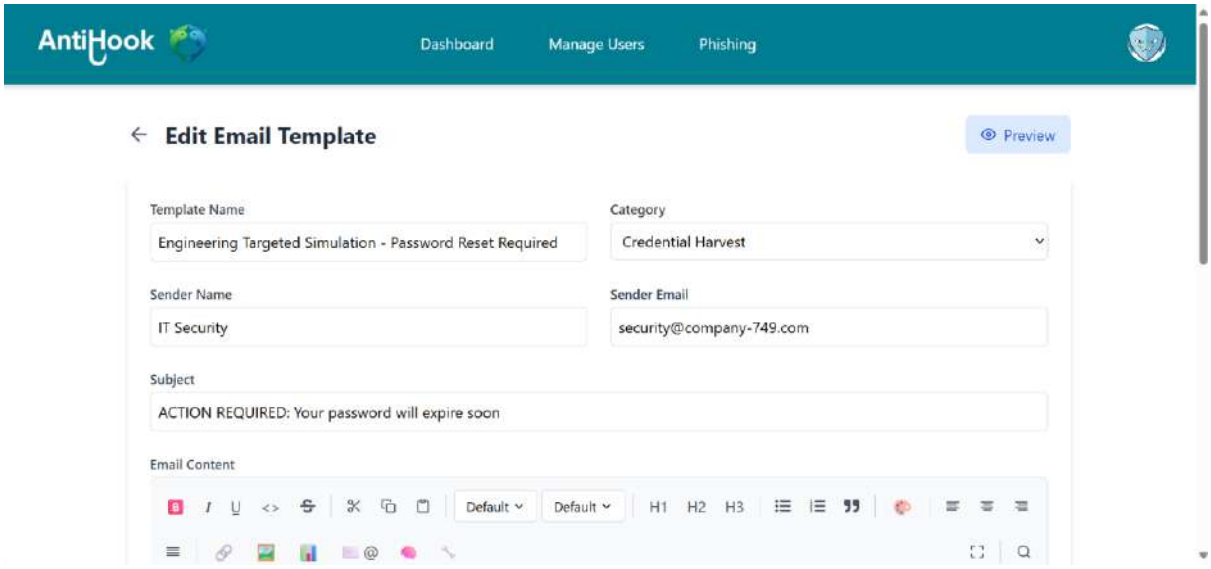


Рисунок 2.11 - Інтерфейс редагування шаблону

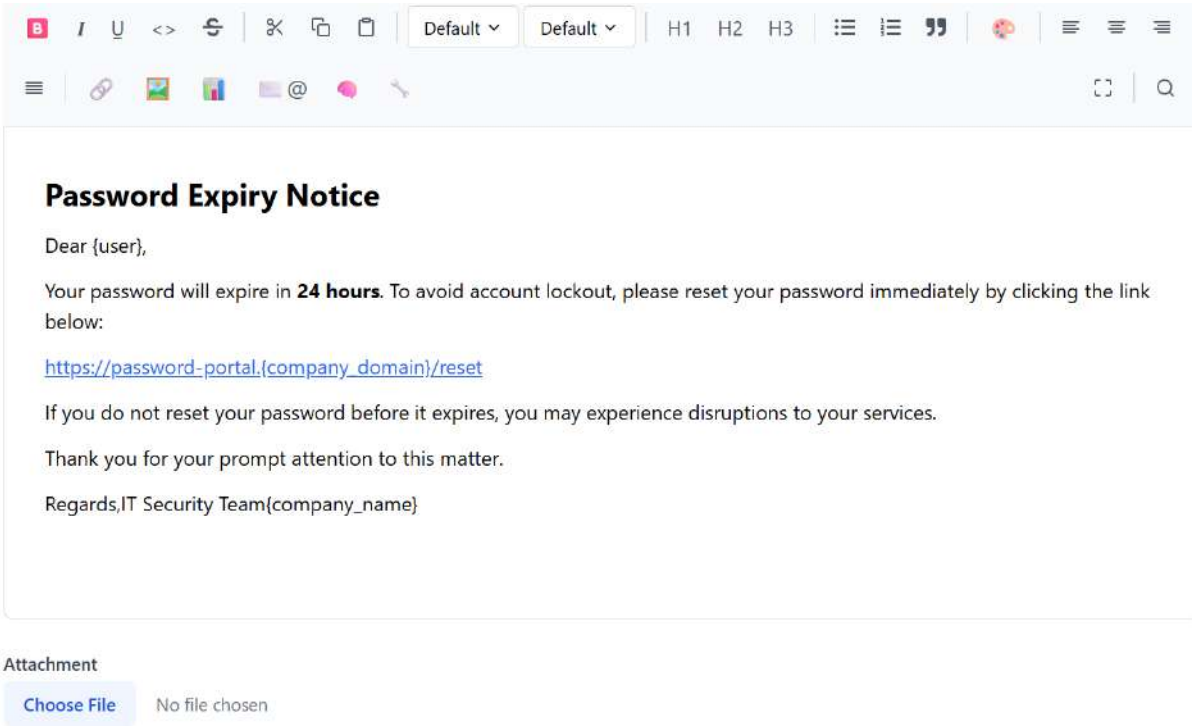


Рисунок 2.12 - Інтерфейс редагування шаблону

Компонент керування шаблонами дозволяє переглядати наявні шаблони, а також редагувати або видаляти їх. Сторінка шаблонів отримує доступні шаблони за допомогою виклику GET /api/templates, відображаючи їх у списку. Адміністратори можуть вибрати шаблон для його оновлення

(завантаживши його назад у редактор Slate) або видалити шаблони, які більше не потрібні. Шаблони також можна пов’язати з цільовими сторінками – наприклад, вміст електронного листа зазвичай містить гіперпосилання або кнопку, що спрямовує на певну URL-адресу цільової сторінки. Система може вставити спеціальне посилання-заповнювач у шаблон, яке замінюється унікальною URL-адресою відстеження для кожного одержувача під час надсилання.

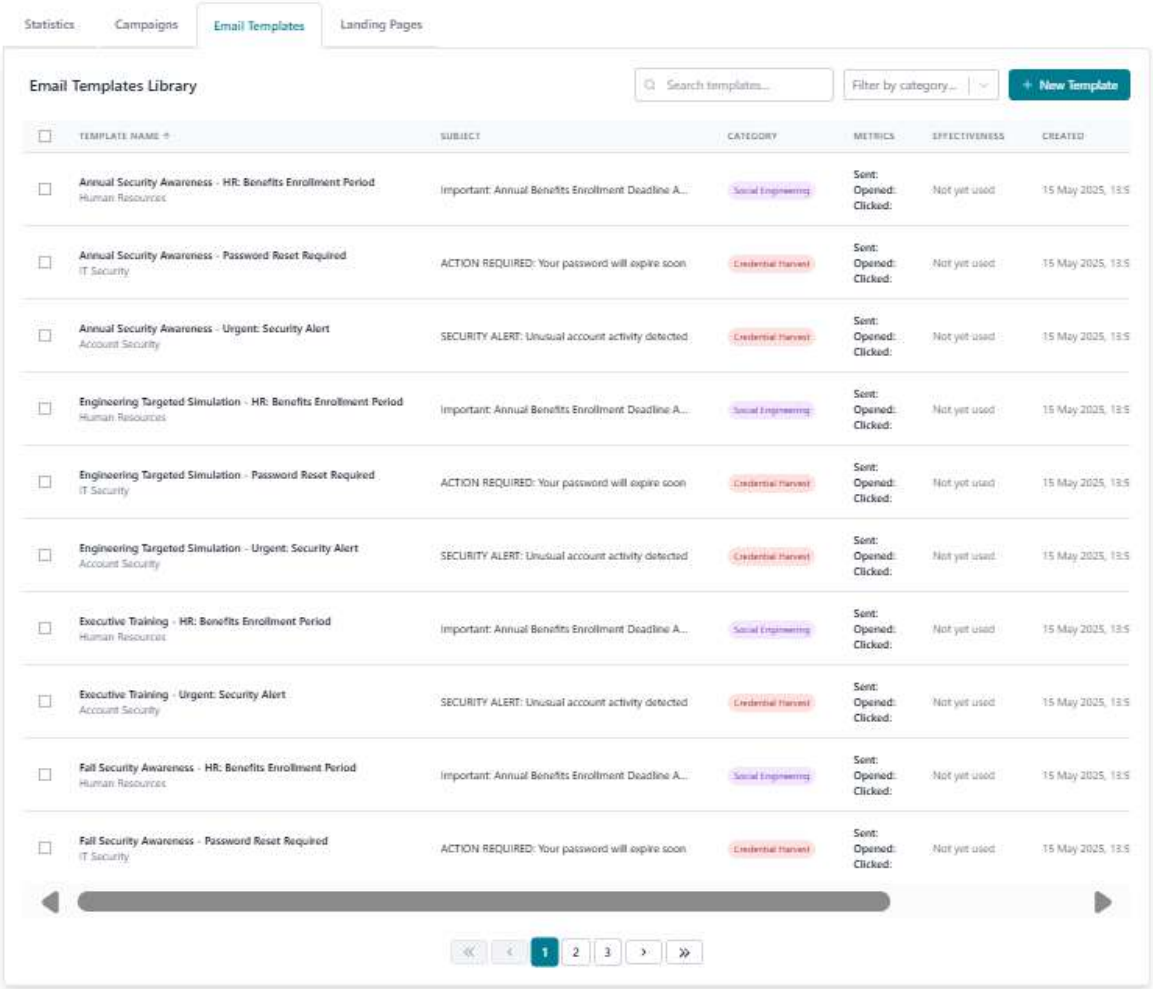


Рисунок 2.13 - Список шаблонів листів

Фронтенд також включає редактор цільових сторінок, функціонально схожий на редактор шаблонів електронних листів.

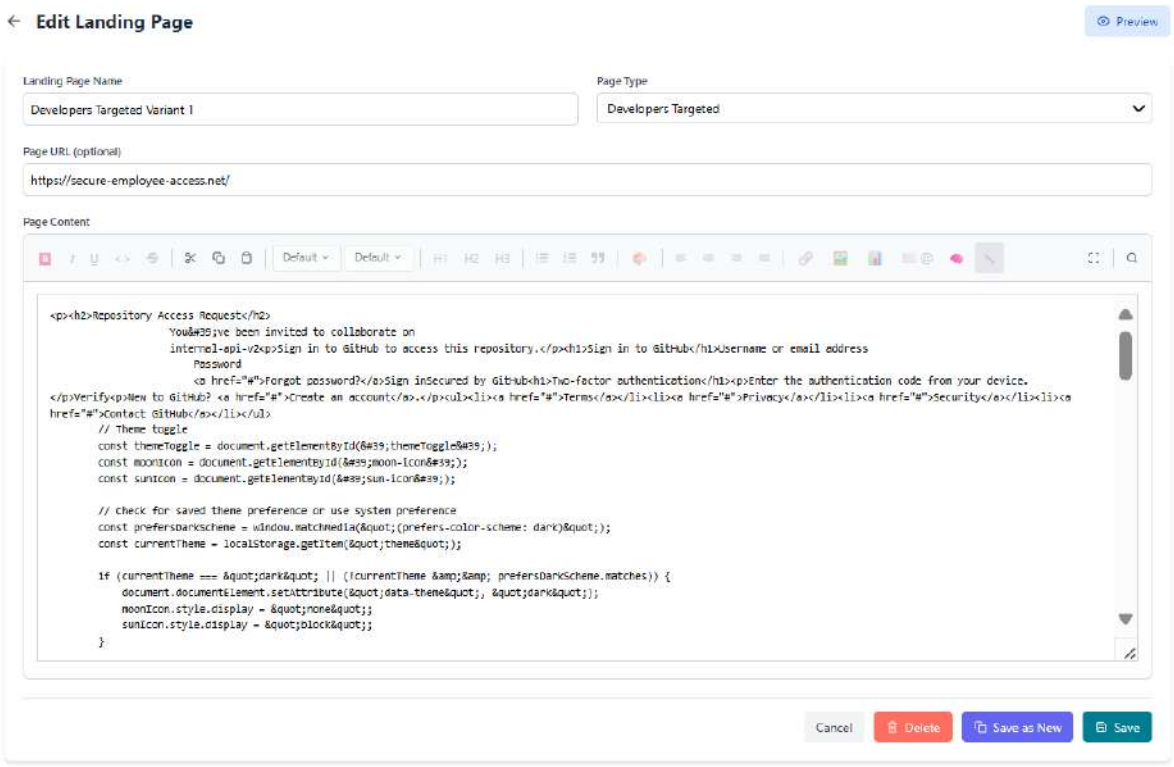


Рисунок 2.14 - Інтерфейс редагування цільової сторінки

Для фішингової кампанії потрібен цільовий список користувачів, яким будуть надсилатися фішингові електронні листи. Модуль створення сценаріїв AntiHook надає гнучку функцію вибору цільової групи, яка підтримує як **статичні групи** (фіксовані списки користувачів), так і **динамічні «розумні» групи** (членство на основі правил, яке оновлюється автоматично). Така конструкція дозволяє адміністраторам або вручну вибирати певних користувачів, або визначати критерії для включення користувачів, що корисно для таргетування певних відділів або ролей без ручного оновлення.

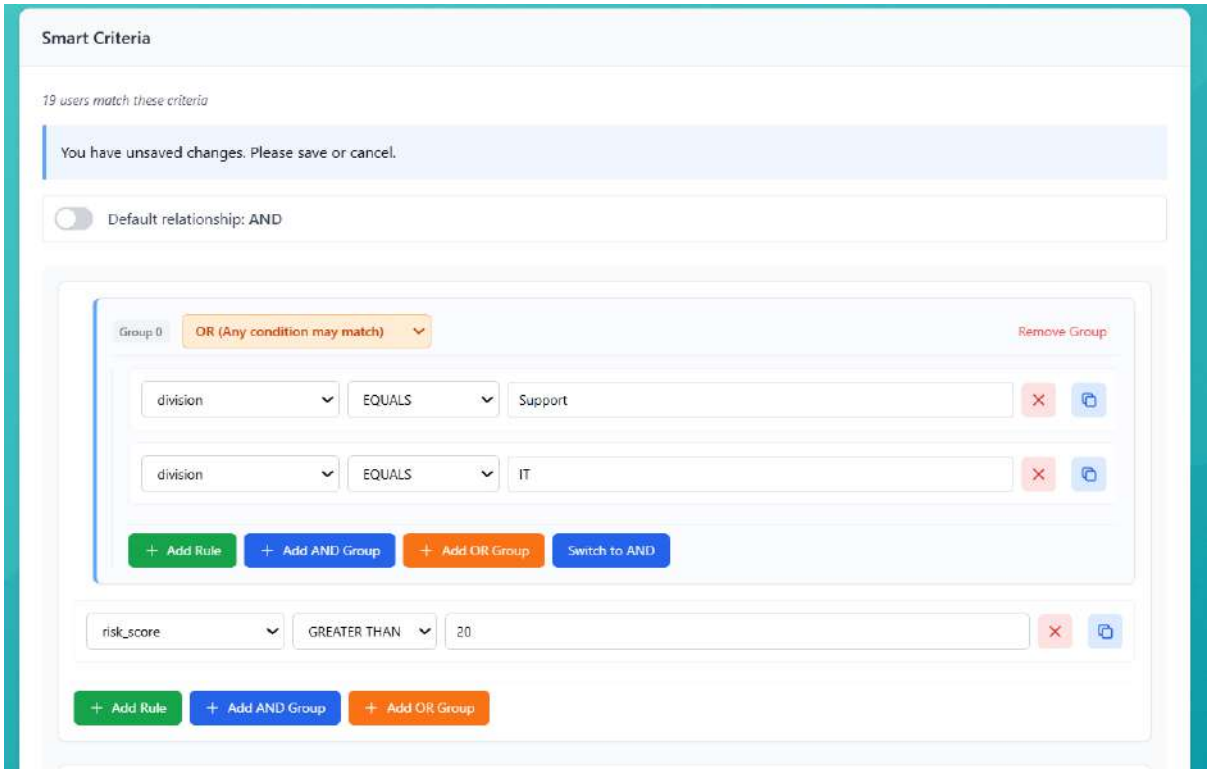


Рисунок 2.15 - Інтерфейс редагування правил динамічної групи

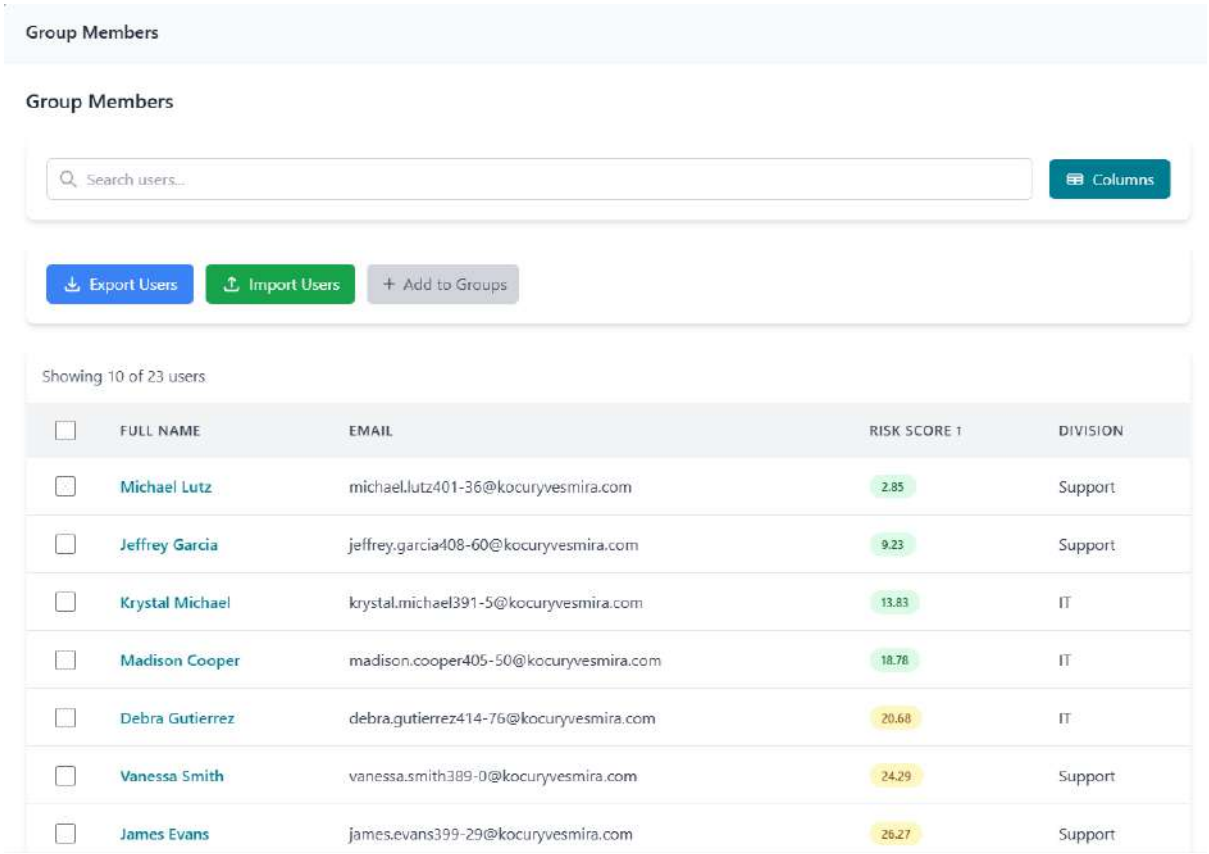


Рисунок 2.16 - Інтерфейс редагування користувачів групи

У фронтенді AntiHook є інтерфейс керування групами (доступний з панелі інструментів адміністратора), де адміністратори можуть створювати нову групу та додавати до неї користувачів. Наприклад, компонент інтерфейсу під назвою *AddGroupModal* надає форму для визначення назви групи та додавання користувачів. Як і в подібних платформах (наприклад, Gophish), адміністратор може додавати користувачів вручну, вводячи дані кожного користувача (ім'я, електронну пошту тощо), або виконувати масовий імпорт з CSV-файлу. Керування даними групи відокремлене від логіки кампанії, тобто адміністратори керують групами в розділі «Користувачі та групи» програми, але під час створення сценарію вони просто вибирають, на які існуючі групи орієнтуватися.

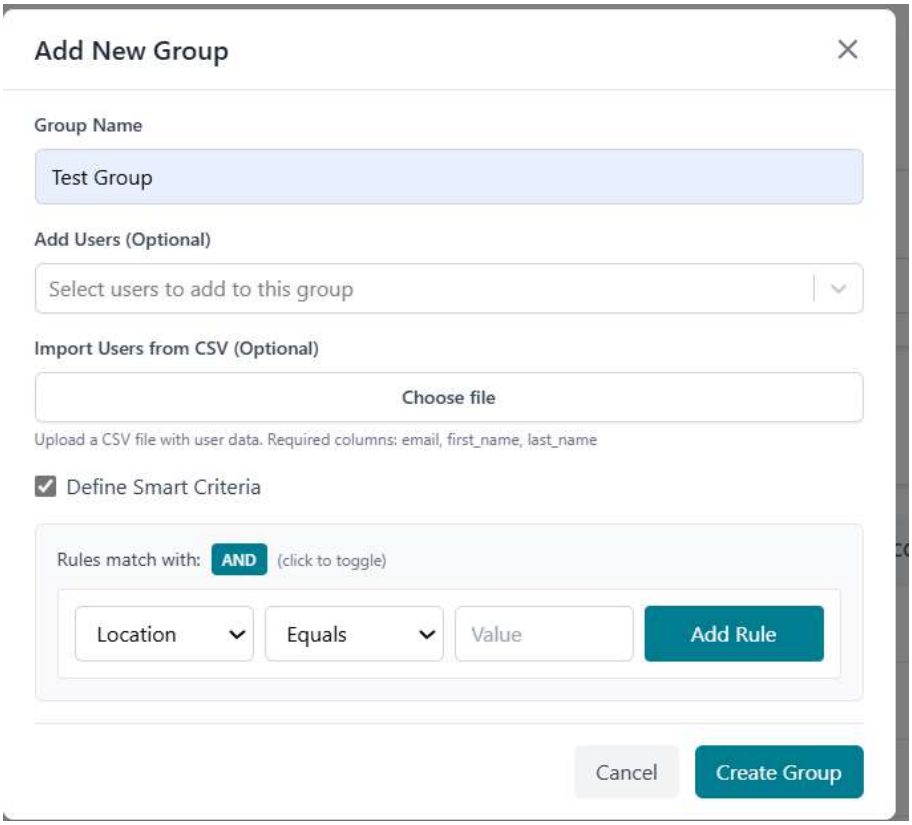


Рисунок 2.17 - Інтерфейс створення групи

Адміністратор може також визначити **розумну групу**, наприклад, для «Усі нові співробітники за останні 30 днів» або «Усі користувачі у відділі продажів». Щоразу, коли група використовується, система перераховує учасників, оцінюючи збережене правило відносно всіх профілів

користувачів. У фронтенді компонент RuleGroup дозволяє адміністратору вибрати атрибут (наприклад, Відділ або ДатаНайму), оператор (дорівнює, містить, дата до/після тощо) та значення. Адміністратори можуть поєднувати кілька критеріїв (наприклад, Відділ – «Продажі» АБО Відділ – «Маркетинг»), щоб розширити групу (бекенд зберігає правило у форматі JSON).

Після вибору вмісту (шаблону електронного листа та цільової сторінки) та одержувачів (груп), адміністратор налаштовує параметри кампанії. Адміністратор зазвичай вводить назву кампанії (для цілей звітності, наприклад, «Q3 Phishing Test – Invoice Bait») та додатковий опис. Він також може налаштувати параметри, такі як надсилання кампанії негайно чи планування її на майбутню дату/час. Крім того, він може вказати поведінку надсилання – наприклад, надсилати всі електронні листи одночасно, а не розподіляти електронні листи протягом певного періоду, чи надсилати електронні листи лише в робочий час або розподіляти їх випадковим чином протягом 2-годинного вікна, щоб імітувати реалістичну схему атаки.

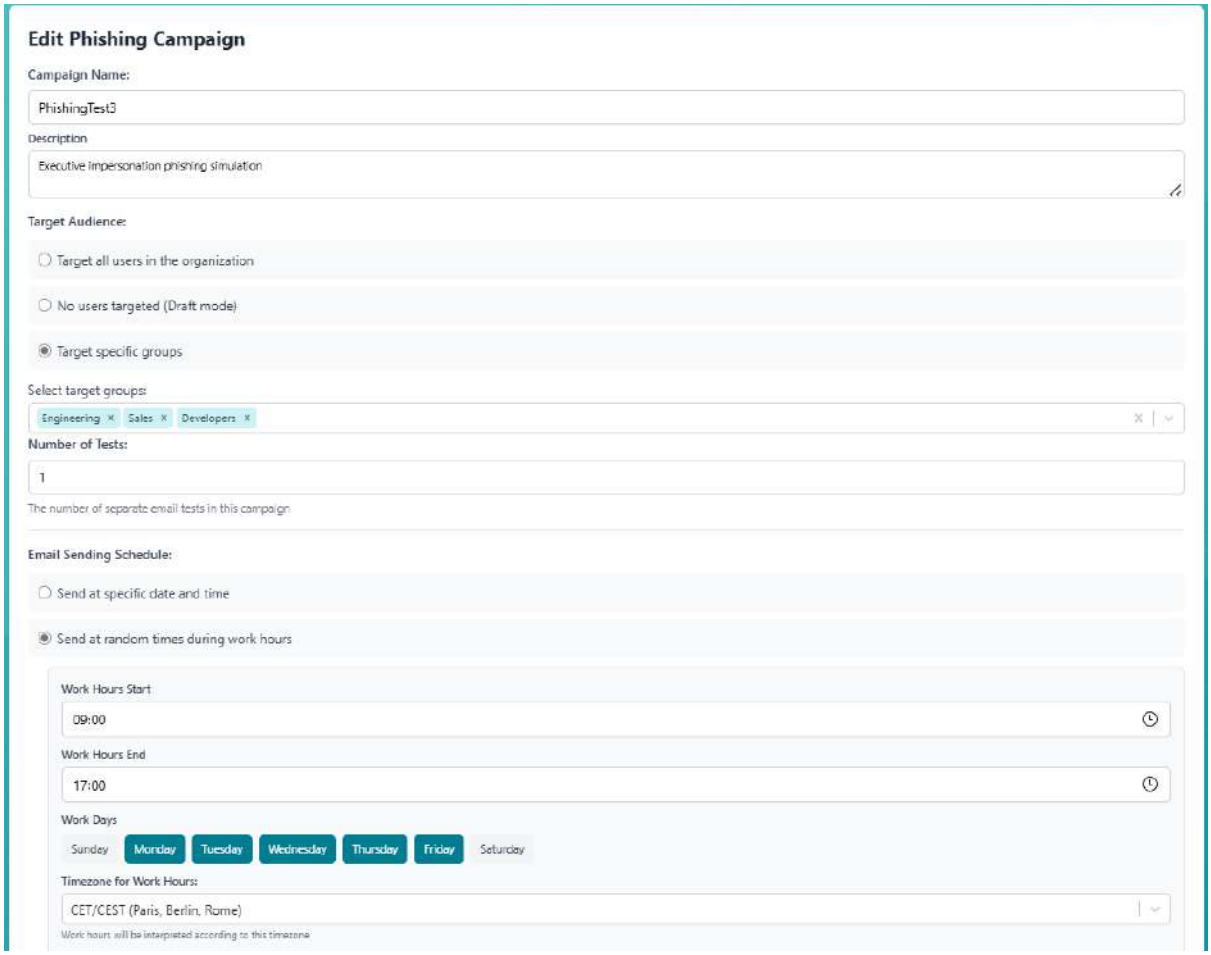


Рисунок 2.18 - Інтерфейс редагування фішингової кампанії

2.4.2. Модуль розсилки та відстеження кліків

Процес надсилання фішингових електронних листів цільовим користувачам здійснюється за допомогою комбінації допоміжних функцій та логіки планування. Основна ідея полягає у використанні попередньо розроблених шаблонів електронних листів та вставці динамічного контенту (елементів персоналізації та відстеження) у ці шаблони для кожного одержувача. Потім система планує та надсилає ці електронні листи відповідно до конфігурації кампанії.

Кожен фішинговий лист базується на HTML-шаблоні, що зберігається в базі даних (*EmailTemplate*). Шаблон містить змінні-заповнювачі (наприклад, {user}, {company_name} тощо), а також основний текст і, можливо, посилання або форму. В утиліті *planner.py*: функція

convert_html(send, db) приймає об'єкт *EmailSend* (який пов'язує користувача, кампанію та шаблон) і виконує підстановку заповнювачів та вставку трекінгових посилань. Наприклад, {random_city} або {random_device}, заповнюються випадково вибраними значеннями (наприклад, «Мюнхен» або «iPhone 14»), {{CLICK_URL}} та {{POST_CRED_URL}} замінюються URL-адресами відстеження, що генеруються під час виконання.

Коли *convert_html* обробляє шаблон, якщо href посилання ще не є URL-адресою відстеження (тобто ще не містить контрольований проєктом домен або токен), функція замінює цей href на URL-адресу виду <TRACKING_BASE_URL>/c/<tracking_token>?url=<original_url>. Тут <tracking_token> - це унікальний ідентифікатор (шістнадцятковий рядок UUID), згенерований для цього конкретного надсилання електронного листа (зберігається в EmailSend.tracking_token), а <original_url> - це легітимна URL-адреса, яка була в шаблоні (наприклад, URL-адреса фальшивої сторінки входу, залежно від сценарію). Оригінальна URL-адреса включається як параметр, щоб, за бажанням, система могла перенаправити користувача на справжній сайт після реєстрації кліку. У даній реалізації користувачі перенаправляються на внутрішню цільову сторінку, але у повноцінній реалізації, за умови контролю над доменами, можливе перенаправлення за оригінальною URL-адресою. У **додатку 1** показано фрагмент коду, який відповідає за попередню обробку електронного листа.

Логіка планування знаходиться у функції *plan_campaign_sends* (у planner.py; використовує *email.calculate_send_time*). Коли адміністратор планує кампанію, ця функція викликається для створення запису EmailSend для кожної комбінації цільового користувача та шаблону, яку потрібно надіслати.

Після створення цих записів EmailSend у базі даних, фоновий процес планування (на базі бібліотеки APScheduler у режимі AsyncIO) періодично перевіряє наявність електронних листів, які потрібно надіслати. Для кожного такого електронного листа, що має бути надісланий, контент генерується за допомогою *convert_html*, як описано, а потім фактично надсилається через SMTP. У **додатку 2** показано, як система надсилає електронний лист за допомогою функції *send_mail* та оновлює статус. *tracking_token* також передається до *send_mail*, що призводить до вбудовування цього токена в заголовок Reply-To електронного листа (у форматі r-<token>@inbound.antihook-sec.com) для перехоплення відповідей на електронні листи: якщо користувач намагається відповісти на фішинговий лист, відповідь надійде на адресу, яка містить токен, і система (через вебхук SMTP2GO) може виявити це та зареєструвати подію «відповідь».

Якщо конфігурація SMTP неповна або сервер повертає помилку, статус надсилання буде позначений як «не вдалося».

Усередині *send_mail* використовується вбудована smtplib Python. Він встановлює адресу відправника на налаштовану адресу (який у тестуванні є фіктивним доменом, яким можна керувати), а адресу одержувача - на адресу отримувача. Потім він відкриває SMTP-з'єднання з SMTP_HOST:SMTP_PORT, запускає TLS, входить у систему, використовуючи SMTP_USERNAME та SMTP_PASSWORD, і надсилає повідомлення.

Для надсилання електронних листів використаний SMTP2GO, хмарний сервіс доставки електронної пошти. У цій реалізації фактичні доменні імена та поштові адреси відправників не є налаштовуваними через обмеження бюджету та ресурсів, тому використовуються заповнювачі за замовчуванням (наприклад, налаштовано адресу відправника

noreply@antihook-sec.com). Однак за допомогою конфігурації (придбання) доменних імен система могла б надсилати користувачам електронні листи з імітованими адресами відправника та посилань.

Ще одна ключова частина полягає в тому, як реалізовано відстеження дій користувачів: модуль використовує спеціально створені посилання для відстеження та піксельні зображення, вбудовані в електронні листи. Посилання для відстеження - це унікальні URL-адреси, що вказують назад на додаток, та містять токен, що ідентифікує електронну адресу та користувача. У розробці використовується базова URL-адреса-заповнювач (URL-адреса ngrok для локального тестування, api.antihook-sec.com для онлайн версії). У повній реалізації адресу мало би бути замінено спеціалізованим доменом відстеження (наприклад, доменом, що імітує легітимний сайт або піддомен домену організації), щоб фішингові електронні листи та посилання виглядали переконливо. Аналогічно, крихітний піксель відстеження (зображення розміром 1×1 піксель, прозоре) вставляється в електронний лист. Коли електронний лист відкривається та зображення завантажуються поштовим клієнтом користувача, цей піксель ініціює запит до сервера (GET `/o/<token>`), тим самим реєструючи подію «відкриття». Цей метод використання невидимого зображення для відстеження відкриттів є стандартним у маркетингових та фішингових кампаніях.

При відкритті фішингового посилання користувач, на сервер надсилається запит GET `/c/<token>`, цільові сторінки отримуються за шляхом `/l/{page_id}/{token}` (обробник кліків створює перенаправлення на цю URL-адресу).

Якщо цільова сторінка фішингу містить форму (наприклад, із запитом на ім’я користувача та пароль), і користувач її заповнює та надсилає, форма налаштовується на надсилання POST-запиту на сервер за адресою `/p/<token>`. Для ілюстрації кінцевих точок відстеження в коді, у

додатку 3 показано реалізацію кінцевої точки відстеження відкриття електронної пошти, а в **додатку 4** показано кінцеву точку відстеження надсилання облікових даних. Сервер, отримавши трекінговий запит, додає до бази даних запис PhishingEvent відповідного типу та оновлює статистичні дані.

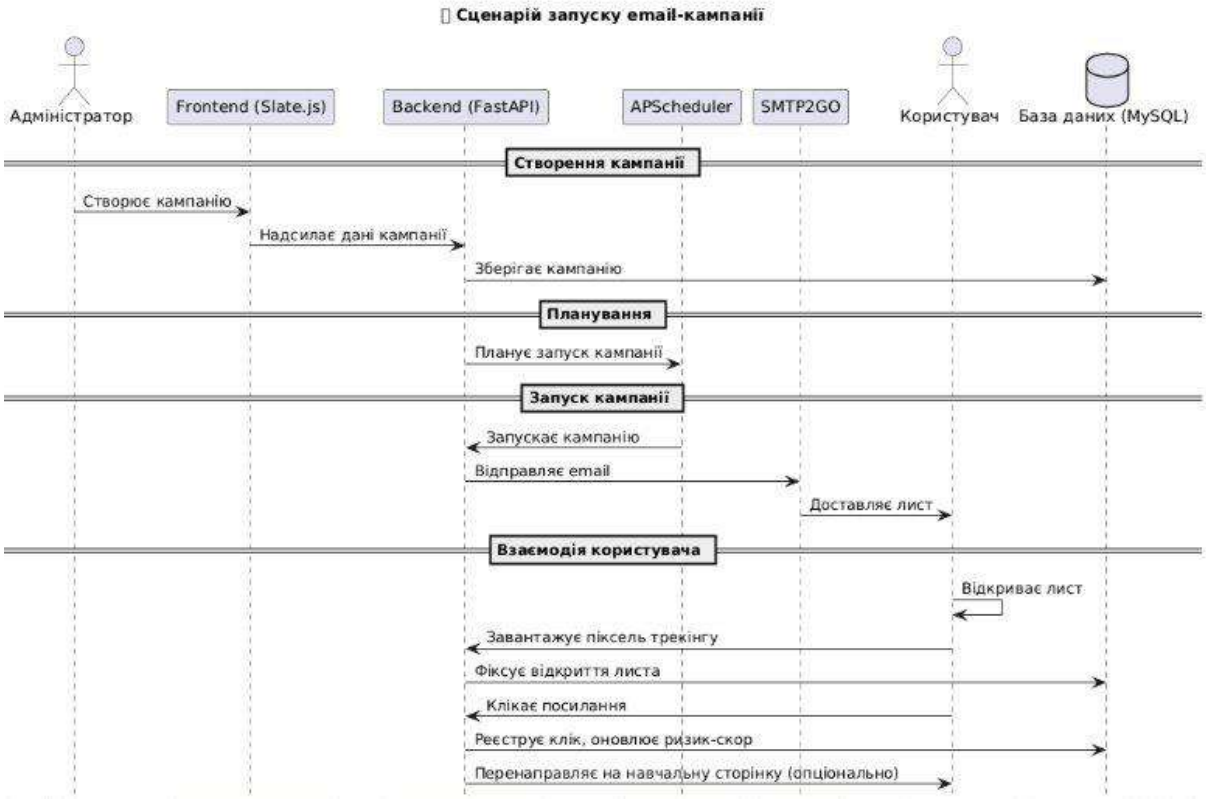


Рисунок 2.19 - Схема роботи фішингової кампанії

На завершення, модуль надсилання та відстеження кліків досягає своєї мети – симуляції фішингових атак та моніторингу відповідей користувачів у контрольованому середовищі, незважаючи на використання деяких заповнювачів та відсутність повноцінної конфігурації (домени тощо). Подальша робота значною мірою передбачає інтеграцію реальних сервісів (доменних імен), забезпечення безпеки та конфіденційності, а також

розширення діапазону симуляцій. Робота, виконана в цьому проекті, закладає міцну основу для такої еволюції.

2.5. Механізм аналізу помилок клікерів

Після збору даних про поведінку користувачів у кампанії, серверна частина оцінює ризики та продуктивність як на рівні користувача, так і на рівні кампанії. Ключова аналітика та логіка включають:

- **Оцінка ризику користувача:** Програма обчислює оцінку ризику для кожного користувача на основі зареєстрованих помилок фішингу. Оцінка ризику – це кількісна міра схильності користувача до фішингу. Кожна з фішингових подій має свою вагу і сукупна оцінка ризику користувача є сумою балів за всі його події. За задумом, вища оцінка вказує на користувача, який більш схильний попастися на спроби фішингу, що допомагає адміністраторам визначити, хто потребує додаткового навчання або підвищення обізнаності.
- **Аналіз результатів кампанії:** Для кожної фішингової кампанії серверна частина обчислює сукупні результати (коефіцієнт відкриття (який відсоток цільових користувачів відкрив електронний лист), коефіцієнт кліків (відсоток тих, хто натиснув на посилання), тощо). Додаток також може реєструвати час кожної події, що дозволяє аналізувати, як швидко користувачі, як правило, клікали після отримання електронного листа (наприклад, щоб побачити, чи кліки відбуваються в основному в перші кілька годин).
- **Аналітика на рівні групи:** Обчислюючи середній бал ризику для кожної групи, система допомагає виявити команди з високим рівнем ризику в організації, щоб можна було зосередитись на навчанні слабких груп.

Проаналізовані дані представлені у додатку у вигляді зрозумілих графіків та діаграм, побудованих за допомогою бібліотеки Recharts.

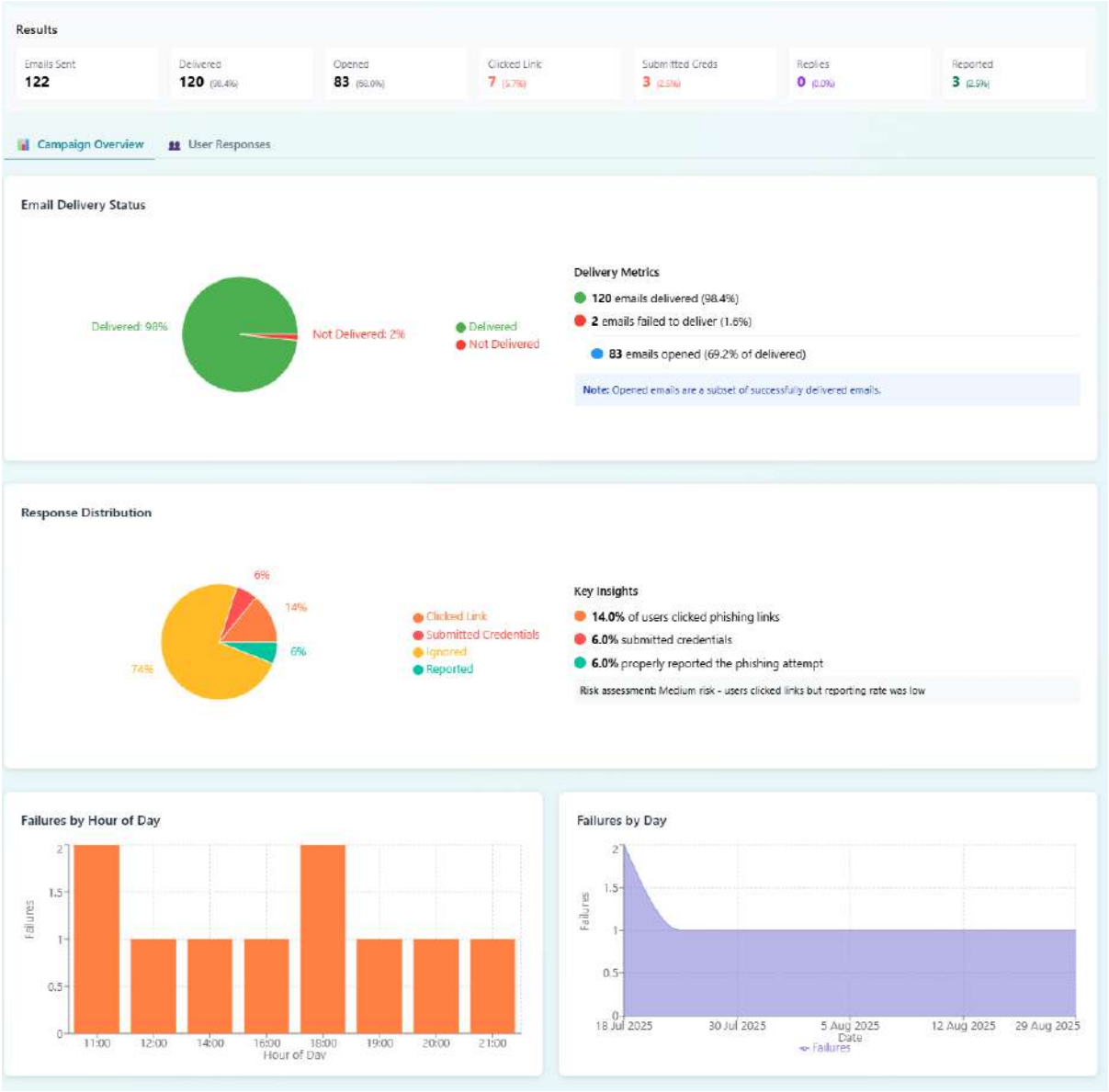


Рисунок 2.20 - Виведення результатів фішингової кампанії

ВИСНОВКИ

Підсумовуючи, у цій роботі було проведено теоретичний огляд тактик фішингу, стратегій захисту та існуючих інструментів симуляції фішингових атак, що створило міцну концептуальну основу для подальшого проєкту. Спираючись на цю основу, було розроблено та впроваджено веб-додаток відповідно до визначених вимог, котрий забезпечує платформу для запуску фішингових кампаній електронної пошти, запису взаємодії співробітників та аналізу результатів.

Основні досягнення цієї роботи включають:

- Розроблено систему управління користувачами з контролем доступу, що дозволяє адміністраторам створювати та впорядковувати облікові записи співробітників і контролювати кампанії.
- Розроблено функцію конфігурації та планування кампаній, що дозволяє адміністраторам визначати цільові фішингові кампанії з певним часом початку та тривалістю.
- Створено бібліотеку шаблонів електронних листів із можливостями налаштування (включаючи ідентифікацію відправника, тему, текст, посилання, вкладення та персоналізовані заповнювачі) для створення реалістичних та різноманітних фішингових повідомлень.
- Впроваджено автоматизований механізм відправлення та відстеження, який надсилає імітовані фішингові електронні листи та реєструє взаємодію користувачів (таку як відкриття електронних листів, кліки на посилання тощо) з часовими позначками.
- Розроблено модуль реєстрації та аналітики даних, який обчислює ключові показники (коефіцієнти кліків, коефіцієнти відкриття та схильність користувачів до фішингу) та визначає користувачів з високим рівнем ризику.

- Додано функції звітності зі зведеною статистикою та візуальними діаграмами для представлення результатів кампанії в доступному форматі та виділення закономірностей, таких як найбільш вразливі відділи.

У майбутньому кілька вдосконалень можуть підвищити ефективність та охоплення платформи. Наприклад,

- інтеграція автоматизованого навчання користувачів може забезпечити негайний зворотний зв'язок зі співробітниками, які потрапляють на симульований фішинг (шляхом пропонування навчальних курсів або тестів)
- Включення методів машинного навчання або штучного інтелекту може дозволити системі генерувати більш складні, персоналізовані сценарії фішингу та точніше прогнозувати користувачів з високим рівнем ризику.
- Розширення симуляції за межі електронної пошти (наприклад, включення SMS- або вішингу)

Додаткові покращення можуть включати інтеграцію з корпоративними службами каталогів та сервісами надання доменних імен, а також покращення інтерфейсу користувача та сумісності з мобільними пристроями для підвищення залученості. Завдяки цим розширенням, програма зможе запропонувати ще більшу цінність як адаптивний інструмент навчання з кібербезпеки та ще більше сприяти зниженню вразливості організації до фішингу.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Verizon: Nearly 80% of Data Breaches Involve Phishing and the Misuse of Credentials [Електронний ресурс] // Verizon. - Режим доступу: https://www.verizon.com/about/news/nearly-80-percent-data-breaches-involve-phishing
2. Executive Reports: Insights on Repeat Clickers in Phishing Simulations [Електронний ресурс] // Keepnet Labs. - Режим доступу: https://keepnetlabs.com/blog/executive-reports-insights-on-repeat-clickers-in-phishing-simulations
3. Phishing simulation exercise in a large hospital: A case study [Електронний ресурс] // PubMed Central. - Режим доступу: https://pmc.ncbi.nlm.nih.gov/articles/PMC8935590/
4. Phishing Attacks: A Recent Comprehensive Study and a New Anatomy [Електронний ресурс] // Frontiers in Computer Science. - Режим доступу: https://www.frontiersin.org/journals/computer-science/articles/10.3389/fcomp.2021.563060/full
5. The COVID-19 scamdemic: A survey of phishing attacks and their countermeasures during COVID-19 [Електронний ресурс] // PubMed Central. - Режим доступу: https://pmc.ncbi.nlm.nih.gov/articles/PMC9349804/
6. Gophish User Guide: Introduction [Електронний ресурс] // Gophish. - Режим доступу: https://docs.getgophish.com/user-guide
7. KnowBe4 Security Awareness Training Features [Електронний ресурс] // KnowBe4. - Режим доступу: https://www.knowbe4.com/products/security-awareness-training/features
8. PhishMe Security Awareness Training (SAT) Platform [Електронний ресурс] // Cofense. - Режим доступу: https://cofense.com/phishme-security-awareness-training-(sat)-platform
9. Phishing Trends Report [Електронний ресурс] // Hoxhunt. - Режим доступу: https://hoxhunt.com/guide/phishing-trends-report

10. Security Awareness Training Statistics [Електронний ресурс] // Keepnet Labs. - Режим доступу: https://keepnetlabs.com/blog/security-awareness-training-statistics
11. Understanding the Efficacy of Phishing Training in Practice [Електронний ресурс] // UC San Diego. - Режим доступу: https://www.sysnet.ucsd.edu/~voelker/pubs/phishtrain-oakland25.pdf
12. Phishing by Industry Benchmarking Report [Електронний ресурс] // KnowBe4. - Режим доступу: https://www.knowbe4.com/resources/whitepaper/phishing-by-industry-benchmarking-report
13. The Psychological Impact of Phishing Attacks on Employees [Електронний ресурс] // Egress. - Режим доступу: https://www.egress.com/blog/phishing/psychological-impact-of-phishing-attacks-on-employees

ДОДАТКИ

Додаток 1.

Уривок коду з planner.py, що показує, як URL-адреси відстеження та піксель відстеження вставляються в HTML-шаблон електронної пошти (за допомогою BeautifulSoup для маніпуляцій з HTML).

```
tracking_base = TRACKING_BASE_URL
click_url = f"{tracking_base}/c/{send.tracking_token}"
post_url  = f"{tracking_base}/p/{send.tracking_token}"

replacements = {
    "{user}": f"{user.first_name} {user.last_name}",
    "{company_name}": organisation_name,
    # ... (random_device, random_city, etc.) ...
    "{{CLICK_URL}}": click_url,
    "{CLICK_URL}": click_url,
    "{{POST_CRED_URL}}": post_url,
}

for a in html.find_all("a", href=True):
    original_url = a["href"]
    for placeholder, value in replacements.items():
        original_url = original_url.replace(placeholder, str(value))
    if "/c/" in original_url:
        continue
    a["href"] = f"{tracking_base}/c/{send.tracking_token}?url={quote_plus(original_url)}"
    a["data-original-url"] = original_url

random_val = hash(str(send.id) + datetime.now().isoformat()) % 1000000
pixel_url = f"{tracking_base}/o/{send.tracking_token}?r={random_val}"
pixel_tag = html.new_tag("img", src=pixel_url, width="1", height="1", alt="",
                         style="display:none;position:absolute;visibility:hidden;")
if html.body:
    html.body.append(pixel_tag)
else:
    html.append(pixel_tag)
```

Додаток 2.

Уривок коду з planner.py (ship_due_emails), що показує надсилання фішингового електронного листа через SMTP та оновлення його статусу.

```
html_content = convert_html(send, db)

if f"/o/{send.tracking_token}" not in html_content:
    html_content += f'<img src="{TRACKING_BASE_URL}/o/{send.tracking_token}" width="1" height="1" style="display:none"/>'
ok = send_mail(
    to=send.user.email,
    subject=send.template.subject,
    html=html_content,
    tags=["campaign", str(send.campaign_id)],
    tracking_token=send.tracking_token,
    attachments=[send.template.attachment_path] if send.template.attachment_path else None,
)
send.status = "sent" if ok else "failed"
db.commit()
```

Додаток 3.

Код для кінцевої точки відстеження відкриття (/o/{token}) у tracking.py. Він реєструє подію відкриття листа та повертає GIF-анімацію розміром 1×1.

```python
@router.get("/o/{token}")
async def opened(token: str, request: Request, db: Session = Depends(get_db)):
    log_event(f"email opened: {token} from {request.client.host}")
    send = db.query(EmailSend).filter_by(tracking_token=token).first()
    if not send:
        return empty_gif()

    record_event(
        db,
        email_send=send,
        event_type=EventType.OPEN_EMAIL,
        meta={
            "ip": str(request.client.host),
            "user_agent": request.headers.get("user-agent", ""),
            "tracking_method": "pixel",
            "referrer": request.headers.get("referer", "")
        },
    )
    db.commit()
    return empty_gif()
```

Додаток 4.

Код для кінцевої точки надсилання облікових даних (/p/{token}) у tracking.py. Він реєструє подію надсилання облікових даних та фіксує дані форми.

```python
@router.post("/p/{token}")
async def credentials_post(token: str, request: Request, db: Session = Depends(get_db)):
    send = db.query(EmailSend).filter_by(tracking_token=token).first()
    if not send:
        return Response(status_code=204)
    form_data = await request.form()
    record_event(
        db,
        email_send=send,
        event_type=EventType.SUBMIT_CREDENTIALS,
        meta={
            "ip": str(request.client.host),
            "user_agent": request.headers.get("user-agent", ""),
            "raw_fields": {k: form_data.get(k) for k in form_data.keys()}
        },
    )
    db.commit()
    return Response(status_code=204)
```