https://doi.org/10.31891/2219-9365-2026-85-35
УДК 004.056.5 : 004.738.5

КОРОБЕЙНІКОВА Тетяна
Національний університет «Львівська політехніка»
https://orcid.org/0000-0003-2487-8742
e-mail: tetianakorobeinikova@gmail.com
КУРИЛЯК Андрій
Національний університет «Львівська політехніка»
https://orcid.org/0009-0002-5289-6470
e-mail: andrii.kuryliak.kb.2024@lpnu.ua
ЖУРАВЕЛЬ Ігор
Національний університет «Львівська політехніка»
https://orcid.org/0000-0003-1114-0124
e-mail: Ihor.m.zhuravel@lpnu.ua

СИСТЕМА РОЗРОБКИ БЕЗПЕЧНИХ ВЕБДОДАТКІВ НА ОСНОВІ ПІДСИСТЕМ НАВЧАННЯ, ПЛАНУВАННЯ ТА ВИБОРУ СТЕКУ ТЕХНОЛОГІЙ

У статті розглянуто проблему забезпечення кібербезпеки вебдодатків класів SPA та PWA в умовах зростання складності атак і залежностей. Запропоновано авторську систему розробки безпечних вебдодатків, що поєднує тренінгову та технічну частини і реалізується через взаємопов'язані підсистеми: навчання персоналу, планування, вибір стеку технологій, прототипування, розробку, тестування та супровід. Кожна підсистема містить сегменти і контрольні точки, які формалізують переходи між етапами та забезпечують принципи «security by design» і «security by default». Описано методики постановки SMART-цілей, формування матриці загроз і сценаріїв реагування, а також критерії оцінювання технологій за ознаками безпеки, підтримуваних стандартів і життєвого циклу оновлень. Демонструється інтеграція засобів безпеки у конвеєр розробки: SAST/DAST, аудит залежностей, WAF з правилами OWASP CRS, політики CSP, контроль доступу до БД (TLS, RLS, шифрування полів), журналювання та моніторинг. Запропоновано процедури верифікації готовності стеку перед прототипуванням і метрики для оцінювання ефективності навчання та впроваджених контрзаходів. Окрему увагу приділено специфіці SPA/PWA: керуванню service worker, безпечному кешуванню офлайн-даних, валідації на клієнті й сервері, захисту токенів і міждоменній взаємодії. Представлено підходи до управління політиками, розподілу відповідальностей, пріоритизації за ризиком і безперервного вдосконалення в парадигмі DevSecOps. Узагальнено практики відтворюваності, документування контрольних точок і використання дашбордів для прозорості рішень. Запропонована система придатна до адаптації під ресурсні обмеження і масштаби організації та різні моделі розгортання.

Ключові слова: вебдодатки, SPA, PWA, безпека, система, підсистема, навчання, планування, стек технологій, SMART-цілі.

KOROBEINIKOVA Tetiana, KURYLIAK Andrii, ZHURAVEL Ihor
Lviv Polytechnic National University

SYSTEM FOR DEVELOPING SECURE WEB APPLICATIONS BASED ON SUBSYSTEMS FOR TRAINING, PLANNING, AND SELECTING A TECHNOLOGY STACK

The article discusses the problem of ensuring the cybersecurity of SPA and PWA web applications in the context of the increasing complexity of attacks and dependencies. The author proposes a system for developing secure web applications that combines training and technical components and is implemented through interconnected subsystems: staff training, planning, technology stack selection, prototyping, development, testing, and support. Each subsystem contains segments and control points that formalize transitions between stages and ensure the principles of "security by design" and "security by default." The methodology for setting SMART goals, forming a matrix of threats and response scenarios, as well as criteria for evaluating technologies based on security features, supported standards, and update lifecycle, is described. The integration of security measures into the development pipeline is demonstrated: SAST/DAST, dependency auditing, WAF with OWASP CRS rules, CSP policies, database access control (TLS, RLS, field encryption), logging, and monitoring. Procedures for verifying stack readiness before prototyping and metrics for evaluating the effectiveness of training and implemented countermeasures are proposed.

Special attention is paid to the specifics of SPA/PWA: service worker management, secure offline data caching, client- and server-side validation, token protection, and cross-domain interaction. Approaches to policy management, responsibility distribution, risk prioritization, and continuous improvement in the DevSecOps paradigm are presented. Practices of reproducibility, documentation of control points, and the use of dashboards for decision transparency are summarized. The proposed system can be adapted to resource constraints and the scale of the organization, as well as to different deployment models.

Keywords: information security; web applications, SPA, PWA, security, system, subsystem, training, planning, technology stack, SMART goals.

Стаття надійшла до редакції / Received 12.12.2025
Прийнята до друку / Accepted 17.01.2026
Опубліковано / Published 05.03.2026

ПОСТАНОВКА ПРОБЛЕМИ У ЗАГАЛЬНОМУ ВИГЛЯДІ ТА ЇЇ ЗВ'ЯЗОК ІЗ ВАЖЛИВИМИ НАУКОВИМИ ЧИ ПРАКТИЧНИМИ ЗАВДАННЯМИ

Інформаційний простір дозволяє майже всьому світу отримувати доступ до ресурсів, які потрібні в освіті, роботі, створення контенту чи ведення соціальних мереж – всі ці напрями використання є частинами особливого світового досягнення під назвою Інтернет. Повсюди створюються великі MAN (Metropolitan Area Networks), LAN (Local Area Networks) та навіть PAN (Personal Area Networks) які дають людям можливість отримувати зв'язок з іншими людьми на відстані без "пересилання голубів з листами" чи SMS. Але окрім цього, творці контенту в Інтернеті можуть розширювати доступ для інших людей не тільки через статичні вебсайти, а і через створення багатофункціональних і неймовірно масштабованих вебдодатків, які вже отримали величезну популярність після входження Інтернету в епоху Web 2.0 [1–3].

Вебдодатки – це інтерактивні програми, які працюють у браузері користувача і виконують певні поставлені розробником задачі. Це можуть бути інтернет магазини, навчальні платформи для студентів, медичні платформи, фінансові сервіси, SaaS (Software as a Service), соціальні мережі і так далі. Всі вони мають перед собою завдання надати користувачу певний набір функціоналу для оптимізації роботи, досягнення потрібної цілі, вирішення побутових справ, збереження даних тощо. Кожен вебдодаток по своєму різний, не існує двох однакових – різні рішення на рівні технологій розробки, сторонні сервіси, платформи, дизайни і найголовніше підтримка безпеки. Щодо самої безпеки, то це надзвичайно вагома проблема в наш інформаційний час. Все більше і більше можна зустріти новин про проведення атак на різні вебдодатки, сайти, портали і ці атаки стають складнішими і частішими [4–6].

Через це досі актуальною проблемою буде забезпечення всеосяжної безпеки вебдодатку, звісно, отримати 100% безпеку свого вебдодатку може забезпечити не кожна команда [7–10]. Для цього потрібні гроші, ресурси, команди спеціалістів, новітні технологічні рішення і фундаментальні знання в цій сфері і для цього керівники витрачають багато часу і зусиль. Зараз з'являється все більше команд ентузіастів які бажають створити свої власні вебдодатки з високою продуктивністю і ефективністю у поставлених задачах. Частина цих розробників забувають про всеосяжну безпеку або не мають змоги її втілювати через брак кадрів, ресурсів чи коштів і через це в кінцевому результаті може вийти хороший вебдодаток але з прогалинами в безпеці. Через це з'являється потреба у створенні рішення, яке зможе зменшити вплив всіх цих факторів та підняти рівень безпеки з нахилом на її поступове збільшення. Це рішення буде сформоване у вигляді системи, яка буде складатись з восьми підсистем, кожна з яких відповідальна за важливий етап процесу створення вебдодатку. В кожній підсистемі робиться акцент на безпеці, вона задіяна в усіх кроках і відповідає найкращим практикам, правилам, інструкціям та засобам. Система поділяється на дві частини – тренінгова і технічна. Дані частини поєднуються між собою за умови того, що команда розробників чи керівник почнуть її використовувати з підсистеми "Навчання", це дасть змогу вирішити проблему обізнаності працівників про кібербезпеку, створити та інтегрувати власну системи навчання та перевірки з циклічним повторенням. Дана підсистема може включати в себе підготовку та навчання усіх інших підсистеми в технічній частині, тобто працівників можна навчити плануванню заходів безпеки, роботи з спеціалізованим стеком технологій, створювати ефективні прототипи, розробляти безпечний вебдодаток та тестувати його і підтримувати після випуску [11].

Отже, проблема полягає в відсутності уніфікованої, адаптивної та відтворюваної системи розробки безпечних вебдодатків, яка бпочиналась із підсистеми «Навчання» для підвищення обізнаності й формування практик «security by design/default», задавала формальні контрольні точки у підсистемах «Планування», «Вибір стеку», «Прототипування», «Розробка», «Тестування», «Супровід», інтегрувала засоби SAST/DAST, аудит залежностей, WAF/CRS, CSP, контроль доступу до БД (TLS, RLS, шифрування), журналювання та моніторинг у конвеєр DevSecOps, забезпечувала специфічні для SPA/PWA політики безпечного кешування, токен-менеджменту й міждоменних взаємодій. Потреба в такій системі підтверджується станом практик у веб-тестуванні/аудиті та узагальнюючими роботами щодо вразливостей вебдодатків.

АНАЛІЗ ДОСЛІДЖЕНЬ ТА ПУБЛІКАЦІЙ

Стан наукових досліджень із безпеки вебдодатків характеризується значною кількістю систематичних оглядів і емпіричних робіт, що фіксують еволюцію атак і зрілість методів тестування. Зокрема, систематичне картування Aydos та співавт. [1] структурує поле веб-тестування, вказуючи на нерівномірне охоплення життєвого циклу й брак відтворюваних процедур оцінювання, що напряму корелює з потребою у "системній" методології, описаній в авторському підході цієї роботи.

Окрема хвиля публікацій стосується SPA/PWA та клієнтських механізмів, що змінюють поверхню атаки. Праці про Service Worker XSS (SW-XSS) на матеріалах ACSAC показали, що компрометація service worker дозволяє зловмиснику персистувати в середовищі браузера, маніпулювати кешем та HTTP-потоком, а вразливі сайти можуть залишатися під контролем десятки днів [3].

Щодо політик браузера, дослідження ефективності Content Security Policy (CSP) у провідних ACM-виданнях виявили проблеми реальної дієвості: неповну підтримку, низький рівень правильного налаштування

та необхідність постійного супроводу політик. CSP слід розглядати як компонент комбінованого захисту, а не самодостатній бар'єр, що узгоджується з ідеєю багатошарових підсистем авторської системи [4].

DevSecOps фокусується на організаційно-процесних бар'єрах та пропонує карти рішень, однак підкреслює дефіцит девелопер-центричних засобів і узгоджених метрик безпеки в конвеєрах CI/CD. Найчастіше це такі рішення, як автоматизація тестів, безперервна оцінка стану захисту й уніфікація політик. Для нас ці елементи лежать в основі пропонованих у роботі підсистем «Навчання», «Планування» та «Вибір стеку» [5, 7].

Емпіричні дослідження SAST/DAST/IAST вказують на обмеження одиничних інструментів і кращі результати при комбінуванні технік (напр., IAST з DAST або SAST), а також на потребу тонкого тюнінгу правил. Це підтверджує доцільність контрольних точок для валідації якості конфігурації інструментів і агрегації звітів у метриці готовності [6, 12].

На рівні периметра дедалі більше праць аналізують WAF: від класичних наборів правил (OWASP CRS) до ML-підходів. Публікації у PeerJ Computer Science та суміжних виданнях демонструють, що базові конфігурації CRS можуть не виявляти частину ін'єкцій, тоді як адаптивні/ML-моделі підвищують виявлення, але потребують суворих процедур валідації та моніторингу, щоб уникати помилкових спрацювань. Це узгоджується з пропозицією інтегрувати WAF (CRS) у DevSecOps-ланцюг з телеметрією та періодичною переоцінкою якості [13].

ФОРМУЛЮВАННЯ ЦІЛЕЙ СТАТТІ

Мета статті полягає у розробленні системи створення безпечних вебдодатків класів SPA та PWA, яка забезпечує інтеграцію принципів security by design і security by default у всі етапи життєвого циклу програмного забезпечення, від навчання персоналу та планування до вибору стеку технологій, прототипування, тестування й супроводу.

ВИКЛАД ОСНОВНОГО МАТЕРІАЛУ

Значною особливістю даної системи є її гнучкість, а саме вона проявляється в її підсистемах. Кожен розробник чи керівник може модифікувати підсистему відповідно до власних потреб або в сторону зручності, а система яка представлена в першому варіанті основується на найбільш ефективних і загальних практиках, що і може зробити її універсальною. Фіксованим завданням залишається виконання даних підсистем, тобто, одразу ж створення прототипу без планування самого додатку є неможливим. Саме тому, кожному розробнику перед початком застосування даної системи потрібно зважити всі нюанси, проаналізувати можливий вплив системи і визначити для себе найбільш критичні місця на яких потрібно буде зосередитись [14–17]. Після аналізу за бажанням керівництва може бути виділена людина, яка буде контролювати виконання систем і збирати дані про підвищенні показники в різних підсистемах і зазначати відправні контрольні точки до переходу на інші підсистеми чи сегменти. Перехід може відбуватись за певних сприятливих факторів чи можливих позитивних результатів, звісно ж, дана система може і якоюсь мірою негативно впливати на якісь процеси, якщо їх виконує невідповідна людина, технології чи методології тощо. Перехід до наступної підсистеми чи сегмента може бути обумовленим такими показниками:

– Досягнута ціль відповідає всім очікуванням, критеріям і поставленим задачам. Дана відповідність повинна перевірятись шляхом перевірки досягнення всіх сегментів, які описані в підсистемі і тому як вони повпливали на розробку вебдодатку чи компанії. Наприклад, якщо після постановки SMART-цілей ефективність чи фокус команд зросли і приносять певні результати, то даний сегмент можна вважати виконаним, але варто перевіряти його ефективність впродовж деякого часу.

– Якщо сегмент не може бути виконаний зараз, але завдяки наступному з'являється можливість. Даний випадок може траплятись в багатьох підсистемах і в залежності від того, яка модифікована версія всієї системи була створена розробником. Інколи для досягнення однієї цілі потрібно спочатку виконати наступну.

– Вплив термінів виконання. Якщо керівник вважає, що виконання підсистеми чи її сегменту може зачекати і спершу потрібно виконати наступну, то він може поставити власну контрольну точку і створити завдання щодо повернення до виконання пропущеного завдання. Найчастіше така ситуація може виникати з некритичними завданнями чи процесам і не зможе викликати проблем в майбутньому, звісно, залишати завдання не виконаним не можна.

– Створюваний додаток не потребує виконання конкретної підсистеми чи сегменту. В системі створення безпечного вебдодатку передбачено, що всі підсистеми і сегменти повинні бути обов'язково виконані, але враховуючи, що кожна команда розробників чи керівники можуть спланувати вебдодаток який не буде відповідати деяким цілям системи, то таке правило не є обов'язковим, але може непотрібні підсистеми чи сегменти можуть замінятись іншими і це не порушить повний цикл розробки.

Додатково потрібно зауважити, що в системі використовується авторська система представлення даних, тобто, кожній підсистемі відповідає певний колір, сегменти в кожній підсистемі виділяються темнішим

кольором, а графічно показана стрілка переводить розробника до наступного сегменту, коли шляхові риски з сегменту представляють конкретні компоненти, які належать цьому сегменту.

Основною підсистемою з якої потрібно почати більшості розробників – це навчання працівників. Основними засадами даної підсистеми є плавне і ефективне введення технічних і нетехнічних команд в сферу кібербезпеки, особливо, якщо з цим виникають проблеми. Для цього в цій підсистемі створюється система персоналізованого навчання працівників, дана система не є строгою і не описує всі відомі напрямки вивчення чи які проблеми потрібно вирішувати в першу чергу. В системі створення безпечного вебдодатку вона, за рішенням розробників чи керівника, може прийматись до використання першочергово навіть перед початком етапу планування в технічній частині системи або викликатись в якійсь з частин самого процесу розробки коли в працівників виникають проблеми або немає належної підготовки. Важливо виділити ресурси для створення окремої команди, в якій будуть провідні спеціалісти з кібербезпеки які зможуть створювати дану систему, систематизувати та інтегрувати її у сам процес передумови створення вебдодатку і навчання працівників.

Така новостворена команда може оптимізувати цю систему під потреби команд розробки, акцентувати увагу на найбільших проблемах, створювати окремі поетапні ланцюжки навчання які будуть включати теоретичну і практичну частину, цим самим давати змогу стикатись з даними проблемами в реальних умов і з реалістично змодельованими діями зловмисника. Важливо також фіксувати кожен успіх працівника в тій чи іншій сфері створення безпечного рішення, за допомогою цього виникає можливість аналізувати кожного співробітника та його успіхи і за потреби повторювати цикл навчання. Звісно ж, дана система не закріплюється за технічними чи нетехнічними командами, вона також є гнучкою і може охопити всіх працівників, основний принцип полягає в інтеграції даної системи на обов'язковому рівні первинного навчання працівників щодо кібербезпеки та кібергігієни в цілому. Але, окрім цього, вже готова інтегрована система може виконуватись на постійній основі для формування великої бази знань і її повторення. Важливо також розуміти, що складові даної системи можуть мінятись дуже стрімко, основними причинами для таких змін є:

– Швидкий вплив нових загроз на бізнес та IT сегменти компанії, які можуть завдавати критичної шкоди, сповільнювати роботу чи взагалі виводити систему з ладу.

– Поява нових співробітників, які можуть бути не впевнені щодо своїх знань стосовно кібербезпеки або які потребують у перевірці знань чи додатковому вивченню.

– Виправлення власних помилок і постійне удосконалення відповідно до якогось компоненту в системі. Якщо помилки в безпеці у якогось працівника виникають надто часто, то це може призвести до створення більш строгого персонального навчання.

– Потреба у підвищенні обізнаності всього штабу працівників чи деяких груп в певній сфері забезпечення кібербезпеки чи протидії певним атакам.

Даний список може мінятись в залежності від ситуації в компанії чи в процесах розробки вебдодатку. Саме тому, основною задачею для даної системи є формування правильного навчального плану, інтеграції в саму компанію та обов'язковому виконанню.

Підсистема навчання працівників може брати участь в кожному етапі розробки вебдодатку і допомагати створювати такі рішення, які зможуть підвищувати стан безпеки компанії і її продуктів. Правильний вибір технічної і інформаційної частин дозволить адаптувати план навчання для кожного працівника, акцентувати увагу на найбільших помилках і задати темп постійного росту для підвищення всеосяжної безпеки (рис. 1).

Важливо зауважити, що після проходження однієї з двох складових, навчальний персонал який відповідальний за навчання працівників може повторно провести повний цикл навчання або якийсь окремий сегмент. Також одразу можна побачити зв'язок даної підсистеми з іншою підсистемою в технічній частині системи – "Планування".

Планування – дуже важливий етап створення безпечного вебдодатку, його повну вимірність неможливо описати в одній системі, адже для кожної групи розробників цей етап може бути кардинально різним. Звісно, знаходження "золотої середини" це дуже реалістична ціль, яка і буде описуватись в даній підсистемі з наведеними прикладами виконання на практиці. Саме підсистема планування формує основу всього процесу розробки, якби її не було, то виконання наступних підсистем виглядало як хаотична біганина від одного сегменту до іншого. Основними цілями в даній підсистемі є:

– Сформувати SMART-цілі для розробки;

– Визначити ключові ризики та вимоги до безпеки;

– Визначити контрольні точки для відстежування прогресу у кожній підсистемі;

– Скласти матрицю загроз і сценарії реагування, які будуть актуальні на всіх етапах розробки.

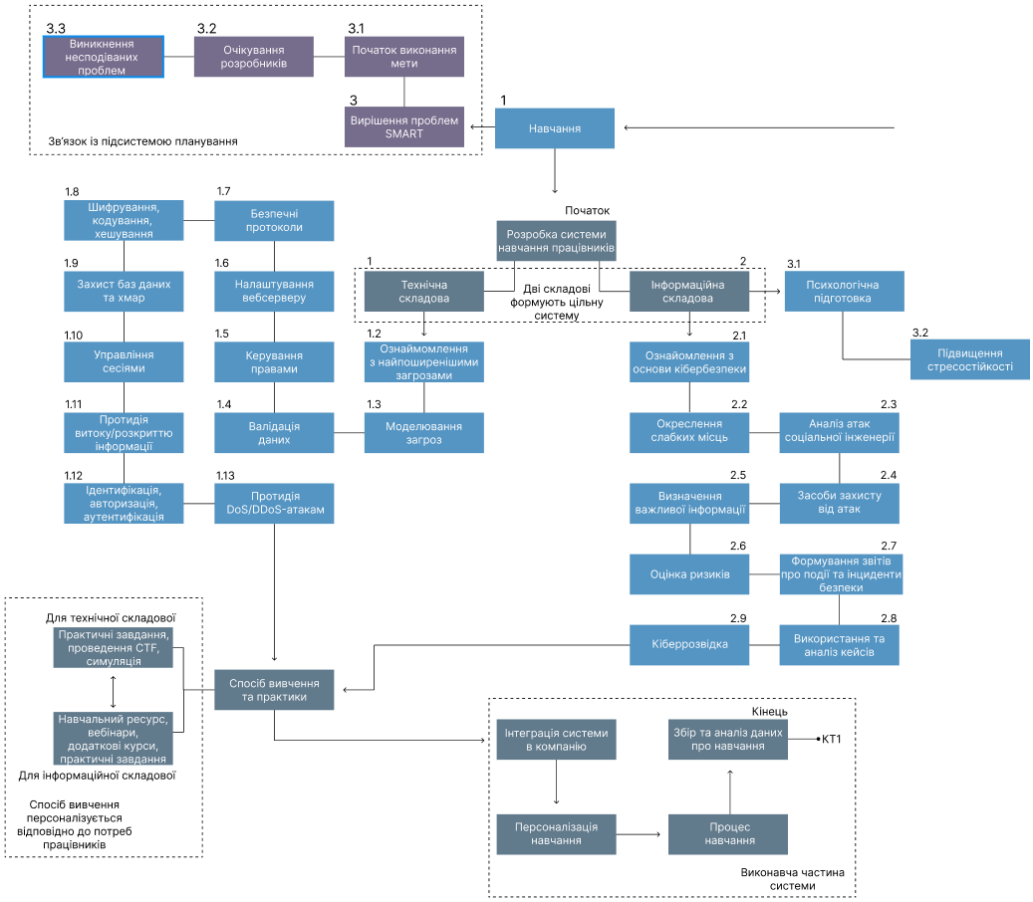

Рис. 1. Графічний вигляд підсистеми навчання працівників

Почати варто з формування SMART-цілей, які допоможуть організувати процес розробки і команду для уникнення критичних помилок. SMART цілі – це конкретний метод формування цілей, які дозволять розробнику ефективно структурувати дані, досягати певних показників і відстежувати динаміку. Починається все з сегменту 1.1 (див. рис. 2), де потрібно сформувати SMART-завдання, тобто конкретної мети, визначити терміни досягання даної мети, потрібної кількості ресурсів та вибору інструментів. Наприклад, реалізувати особистий кабінет в додатку за один спринт за допомогою певних інструментів. Саме ж слово SMART означає не тільки в перекладі з англійської “розумний”, а також акронім. Для кращого розуміння нижче наведено таблицю яка описує кожну частину SMART та пояснює її практичне застосування для розробника (табл. 1).

Таблиця 1

Пояснення SMART

Критерій	Пояснення	Використання на практиці	Вплив
S-Specific Конкретна	Завдання повинне бути чітким і зрозумілим.	Розподілити 25% ресурсів на тестування додатку.	Конкретна мета і створення завдань для її досягнення полегшує роботу працівників і наштовхує їх на правильний шлях вирішення.
M-Measurable Вимірна	Який показник показує прогрес до досягання поставленої мети	Створити команду для тестування додатку шляхом пентесту.	Вибір правильного показника дозволить оптимізувати його виконання для швидшого вирішення задач.
A-Achievable Досяжна	Формувати тільки досяжну мету, яка буде реалістичною	Підвищити ефективність тестування SAST на 200% за два спринта (неможлива мета)	Неможлива мета може демотивувати команду та не дати можливості прогресувати у власних знаннях.
R-Relevant Значуща	Мета повинна бути актуальною та релевантним запитом для розробників і відповідати планам розвитку	Задіяти нові інструменти DAST для підвищення ефективності SAST	При формуванні актуальної і релевантної мети потрібно вирішити чи не конфліктує вона з іншими метами і чи є в ній цінність для розробників.
T-Time bound Обмежена в часі	Мета та ціль повинна мати термін для своєї реалізації	Повністю застосувати стандарт ASVS у тестуванні за один квартал (3 місяці)	Терміни не повинні бути занадто довгими, адже це може зменшити фокус команди. Якщо час довший то рекомендовано ділити її на етапи.

SMART-цілі формуються в даній підсистемі для продуктивної комунікації команди розробників, дизайнерів, тестувальників з підвищенням кращої роботи над поставленими задачами, підвищенням відсотку кращої успішності, ефективному відстеженню прогресу та покращенню фокусу. Але як і кожна система,

технологія чи інше рішення SMART має свої переваги та недоліки, і саме в підсистемі навчання працівників цьому приділяється окрема увага як підготовка до першого етапу розробки. Вирішенню цих проблем і навчанню правильності постановки SMART-цілей буде виконувати лідер у розробці або відповідальна за ці цілі людина, яка зможе грамотно їх сформувати, розприділити і поставити часові рамки. Перелік переваг та недоліків наведені в таблиці 2.

Таблиця 2

Переваги та недоліки SMART

Переваги	Недоліки
Прозорість результатів запланованої роботи на старті	При постановці певної мети команда може не зрозуміти з чого варто починати
Універсальність SMART	Не всі поставлені плани можуть відповідати очікуванні розробників
Ефективна практичність цілей яка базується на певному плані з точно описаними кроками дій	Несподівані проблеми які можуть виникнути в процесі досягання поставленої цілі можуть відволікти від вже сформованого шляху і зірвати терміни реалізації
При правильному формуванні SMART-цілей зростає економія ресурсів – розрахунок часу та бюджету	

Такі проблеми які стають недоліками даного методу можуть потребувати додаткових рішень, але не кожна команда готова їх застосовувати вже в запущеному процесі розробки. Саме тому наявні проблеми потрібно вирішувати в підсистемі навчання, і тому для прикладу далі описано рішення для наявних в таблиці недоліків:

– Проблема початку виконання поставленої мети. Для вирішення її після вже сформованої мети керівник проєкту повинен описати точку входу в процес розробки, поставити перші кроки для досягнення точки закріплення і продовженню роботи. Такий тренінг може забрати трохи більше часу, але тоді розробники зможуть правильно почати шлях виконання.

– Проблема відповідністю до очікування розробників. Якась мета чи цілі можуть не відповідати очікуванням розробників і це нормально. Для вирішення такого керівник повинен формувати команди для реалізації не тільки із закріплених за посадами працівників, але і опиратись на навички та запал кожного розробника. Крім того, мета і ціль має частково або повністю зачіпати весь штаб розробників які працюють над додатком і задовільняти вимоги більшості чи підлаштовувати хід розробки.

– Виникнення несподіваних проблем. Неможливо уникнути усіх проблем які можуть виникнути на шляху, адже не про всі є можливість дізнатись. Тому, слід виділити тільки ті які можуть створити найбільшу проблему під час розробки і відволікти від виконання роботи. Список таких проблем можна оновлювати після закінчення кожного спринта і створювати власний сигнатурний список проблем для ефективнішого вирішення в майбутньому.

Таким чином, система одразу показує свою ефективність у вирішенні проблем які можуть виникнути, а також пов’язаність підсистем між собою для отримання всеохоплюючого рішення для кожного розробника який бере участь у процесі створення безпечного вебдодатку.

Передумовою для переходу до наступної підсистеми “стек технологій” є виконання набору контрольних точок, які готують розробників не тільки до наступної підсистеми але і до інших, які будуть виконуватись в уже прямому циклі розробки. Тому, серед таких контрольних точок (КТ) є:

КТ1 – Визначення функціональних і безпекових вимог. Команда проводить збір і систематизація функціональних вимог вебдодатку, що визначають її майбутні складові (фронтенд, бекенд, бази даних, інші інтеграції). Щодо безпекових вимог, то для них створюється спеціальний документ, який буде описувати потенційні ризики, ймовірність виникнення цих ризиків та рівень впливу;

КТ 2 – Побудова логічної архітектури додатку. Полягає у створенні логічної схеми компонентів вебдодатку, тобто взаємодії між всіма складовими. Важливим пунктом в цій контрольній точці буде зона довіри, яка поділятиметься на користувацьку зону (фронтенд), прикладну зону (бекенд), зону зберігання даних (бази даних) та всеосяжний периметр безпеки (вебсервери, проксі тощо);

КТ3 – Критерії вибору технологій. Фокусується на протидії вибору технологій тільки через особисті вимоги (зручність роботи, простота, зрозумілий інтерфейс, доступність і т.п) і впроваджує об’єктивні критерії оцінювання певної технологій. До таких критерії входять: відповідність вимогам безпеки, підтримка стандартів, активність спільноти і частота оновлень, сумісність з наявною інфраструктурою та інструментами, вимірювані показники продуктивності, наявність готових механізмів резервування або моніторингу;

КТ4 – Планування політики безпеки та відповідальності. Звісно ж, до початку вибору самого стеку потрібно розподілити ролі у сфері безпеки. Цей розподіл обов’язково повинен фіксуватись у внутрішній політиці безпеки, щоб під час вибору технологій одразу враховувати, які засоби автоматизації потрібно інтегрувати. Ролі можуть відрізнятись від команди до команди, кожен формує їх по своїм власним потребам, далі наведено деякі актуальні ролі:

– Хто буде відповідати за політику резервного копіювання та тестові відновлення;

– Хто адмініструє користувацькі ролі та доступ до бази даних;

– Хто контролює стан залежностей;
– Хто відповідає за аудит і моніторинг подій безпеки.

КТ5 – Підготовка сценаріїв загроз і тестів на проникнення. Затвердження стеку для розробки вебдодатку дуже важливий етап в процесі його створення і тому перед цим потрібно перевірити архітектурну модель на стійкість до типових і актуальних атак. Така перевірка здійснюється за рахунок формування покрокових сценаріїв тестування цих атак. Також визначається на якому рівні можна зупинити ту чи інші загрозу, наприклад: у браузері, на вебсервері, у бекенді чи на рівні бази даних. Результатом шляху досягнення даної контрольної точки є окремий документ “Threat-Mitigation Matrix”, який буде інформувати розробників, які саме засоби захисту повинні бути реалізовані.

КТ6 – Підготовка бюджетних і часовим рамок. Етап планування завершується оцінкою ресурсів і розрахунком часу, а також вартості ресурсів для впровадження безпекових заходів. На шляху досягнення даної контрольної точки (далі – КТ) потрібно сформувати:

– Таблицю чи лістинг витрат на впровадження інструментів безпеки;
– Часовий графік для повного впровадження безпекових заходів у вебдодаток (наприклад, запуск WAF у тиждень 4);
– Резерв часу на тестування відновлення бази даних та аудит конфігурацій.

Контрольні точки зустрічаються у кожній підсистемі, їх виконання все ближче наближує команду розробки до повного виконання даної підсистеми і переходу до наступної. Самі КТ можливо виконувати не почергово, тобто спочатку досягти КТ1, а тоді перейти до КТ3 чи КТ4, такий спосіб не вплине на сам процес розробки якщо деякі КТ не пов’язані з наступними, якщо ж пов’язані то тоді їх потрібно буде виконувати почергово. На графічній частині системи дані КТ зображені рискою з точкою на кінці і підписані відповідно до свого номера, нижче показаний рисунок підсистеми планування і її контрольних точок:

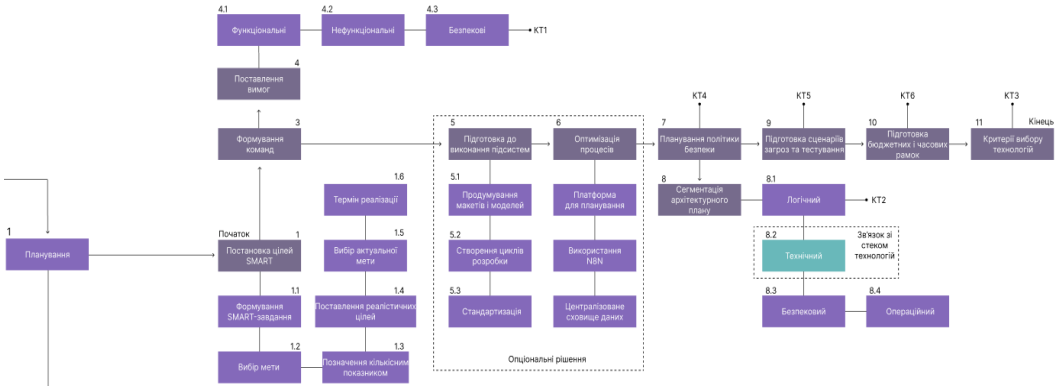

Рис. 2. Графічне представлення підсистеми планування

Відповідно після досягнення всіх КТ на підсистемі планування відбувається перехід до підсистеми вибору стеку технологій (рис. 3). Саме ця підсистема є критичною частиною архітектурного формування системи, адже саме на цьому етапі закладається технічний фундамент, який визначає рівень безпеки, масштабованості, стабільності та гнучкості.

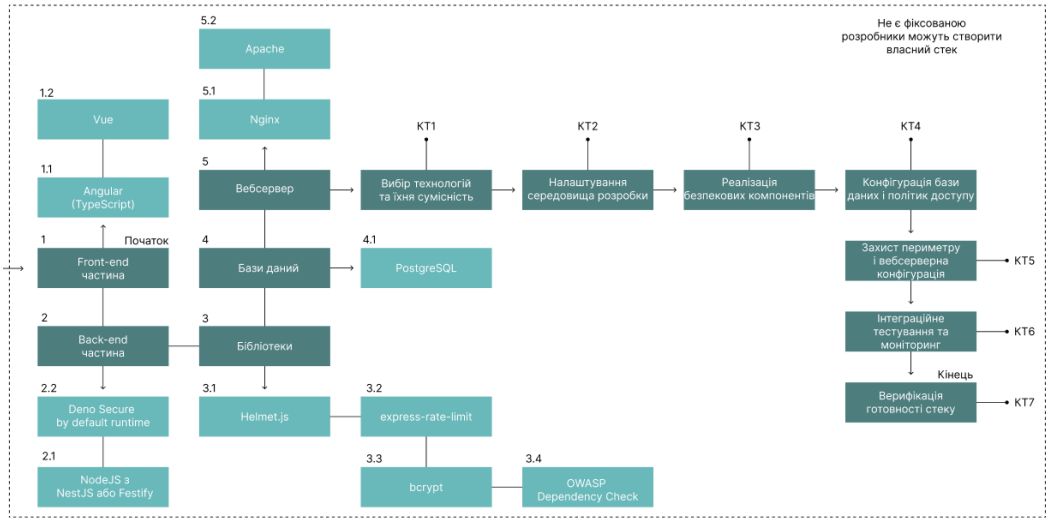

Рис. 3. Графічне представлення підсистеми вибору стеку технологій

Основною ідеєю даної підсистеми є створення єдиної екосистеми технологій у якій усі компоненти – від інтерфейсу користувача до вебсерверу працюють узгоджено та за принципом “Security by default (безпека за замовчуванням). Саме на даній підсистемі етап планування перетворюється на технічний, де ідеї створені в минулому, моделі загроз і вимоги безпеки отримують практичні рішення. Контрольні точки відіграють не менш важливу роль на етапі вибору стеку технологій, вони створюють чітке розуміння кроків поєднання різних рішень, налаштування середовищ для тестування працездатності, реалізації безпекових компонентів, конфігурацій баз даних та політик доступу, захисту периметру і вебсерверної конфігурації, інтеграційному тестуванні та моніторингу і верифікації готовності стеку до прототипування. В цьому випадку кожна КТ дає окреме фундаментальне рішення для підготовки до створення прототипу і роботи в цілому(див. рис. 3). Далі детально будуть розглядатись дані КТ.

КТ1 – Вибір технологій та їхня сумісність. Першочерговою задачею даної підсистеми є формування стабільного та ефективного складу технологій для кожного рівня системи, тобто на шляху до досягнення даної КТ відбувається аналіз взаємної сумісності компонентів (версій, ORM драйверів бази даних, бібліотек тощо). Для команди рекомендовано підготувати окремий файл з описом причин вибору кожної технології і бібліотеки, критерії оцінки та ризик заміни;

КТ2 – Налаштування середовища розробки. Після затвердження стеку формується базове середовище, тобто репозиторії із політиками безпеки та скриптами для статичного аналізу (SAST) та автоматичного тестування. Цим самим розробники можуть запускати систему локально в ізольованому режимі, де всі рішення будуть взаємодіяти через захищені канали, а будь-яке порушення політик чи інші проблеми будуть фіксуватись лінтерами і CI-перевірками;

КТ3 – Реалізація безпекових компонентів. Дана КТ може відігравати центральним ядром усієї підсистеми, через “повне шліфування” безпеки із додаванням додаткових заходів у вигляді бібліотек, скриптів та правил. Крім того, у CI/CD інтегрується OWASP Dependency Check, який виконує аналіз усіх бібліотек для знаходження критичних CVE-вразливостей, а результати звітів аналізу фіксуються у репозиторії;

КТ4 – Конфігурація бази даних і політик доступу. Шлях виконання цієї КТ основується на додаткових технічних рішеннях стосовно бази даних на PostgreSQL, адже Oracle можуть собі дозволити тільки великі команди з великою кількістю ресурсів. Тобто для PostgreSQL активовується TLS-шифрування для всіх з’єднань, вмикається автентифікація SCRAM-SHA-256, та грамотно створюються окремі ролі для читання, запису та адміністрування. Також додатково використовується політика RLS (Row Level Security) – завдяки їй дані будуть відфільтровуватись на рівні SQL відповідно до ролі користувача. Окрім цього, модуль pgcrypto забезпечить шифрування полів з персональними даними, а логування буде проводитись через pgAudit із поділом рівнів критичності полів. Обов’язковим до виконання також є розгортання системи резервного копіювання pgBackRest або WAL-G, з виконанням щоденних інкрементних та щотижневих повних бекапів;

КТ5 – Захист периметру і вебсерверна конфігурація. Після налаштування та “поставлення на ноги” внутрішніх компонентів, необхідно забезпечити безпечний периметр системи. Для цього на рівні Nginx (або Apache) впроваджується TLS 1.3 та політика HSTS з поступовим збільшенням часу дії, також вебсервери налаштовуються на обмеження розміру запиту й кількості з’єднань на клієнта. Додатково інтегрується “ModSecurity” із набором правил OWASP Core Rule Set (CRS), це забезпечує базовий WAF-захист від SQL-ін’єкцій, XSS, CSRF та LFI/RFI-атак. Критеріями для досягнення даної КТ є: сервер приймає лише зашифровані з’єднання, WAF працює в активному режимі без фатальних помилок, а всі сканування повертають рейтинг безпеки A або вище;

КТ6 – Інтеграційне тестування та моніторинг. Після вже фінального вибору потрібних рішень необхідно провести комплексне тестування взаємодії цих самих рішень між собою. Для цього Фронтенд тестується на відповідність CSP-політиці, бекенд на правильність обробки валідаційних помилок і “rate-limits”, а база даних на коректність RLS-фільтрів і аудит-логів. Паралельно цьому впроваджується одна із систем моніторингу - Prometheus, Grafana, ELK Stack або Sentry, які будуть збирати метрики й події безпеки. Критеріями для досягнення даної КТ є: система пройшла інтеграційні тести, метрики доступні для аналізу, журнали безпеки фіксують усі спроби доступу, а команда має дашборд для спостереження;

КТ7 – Верифікація готовності стеку до прототипування. Дана КТ являє собою формальну перевірку стабільної взаємодії всіх обраних технологій і їх відповідність вимогам планування. Для цього створюється звіт про готовність стеку, в якому будуть: перелік усіх версій компонентів і бібліотек; результати аудиту залежностей і безпекових сканувань; докази виконання резервного копіювання; звіт тестування TLS, CSP і RLS-політик; журнал пройдених тестів на навантаження та проникнення.

Лише після затвердження цього звіту підсистема вважається завершеною, і команда може переходити до етапу прототипування функціональних модулів. Відповідно після виконання КТ7, команда розробників зможе успішно почати створювати перший прототип свого вебдодатку в наступній підсистемі. Вплив обраного стеку технологій є величезним, він задає темп для команда, фокусує увагу на прийманнях несподіваних і планованих рішень, зазначає які кроки у розробці та підтримки безпеки потрібно робити під час усього циклу створення безпечного вебдодатку.

Кожен розробник повинен особисто оцінити вплив окремої технології на майбутнє додатку, зважити всі позитивні та негативні сторони і тоді зробити вибір, або ж доповнювати вже існуючий і підвищувати його стан. Дана підсистема також вважається гнучкою, адже ні один сегмент не є фіксованим, що вказує розробникам на можливість вибору власного стеку з вже описаними технологіями чи зовсім новими.

ВИСНОВКИ З ДАНОГО ДОСЛІДЖЕННЯ І ПЕРСПЕКТИВИ ПОДАЛЬШИХ РОЗВІДОК У ДАНОМУ НАПРЯМІ

Система розробки безпечного вебдодатку позиціонує себе як цілісний організм, який поєднує в собі всі найкращі практики планування, вибору стеку технологій, прототипування, розробки, тестування та підтримки. Система відбирає актуальні засоби та методи для досягнення всеосяжної безпеки і підтримки її у всьому життєвому циклі розробки і після випуску, даючи розробникам вільний вибір рішень в деяких підсистемах. Рішення про реалізацію певної частини системи приймаються на основі повного виконання контрольних точок та перевірки працездатності поставлених рішень. При цьому, підготовкою до виконання підсистем в технічній частині є правильне навчання працівників, не тільки кібербезпеці але і роботі з системою в цілому, усуваючи можливі характерні проблеми при першому проходженні. Система впевнено поєднує кожен етап розробки між собою, утворюючи нерозривний ланцюг прогресу створення безпечного вебдодатку, який зможе зберегти цілісність даних користувачів та компанії-розробника і виконати поставлену перед собою задачу. Окремою її цінністю є актуальність – всі розібрані в ній засоби та методу впровадження безпеки є фундаментальним та прогресивними в наш час, даючи впевненість у виконаних діях.

References

1. Aydos, M., Kaya, Y., & Şen, S. (2022). *Security testing of web applications: A systematic mapping study. Journal of King Saud University – Computer and Information Sciences.* Elsevier. https://doi.org/10.1016/j.jksuci.2022.101651
2. Balsam, S., Shaqour, A., & Hasan, M. (2025). *Web application testing – Challenges and opportunities. Journal of Systems and Software, 210,* 111234. https://doi.org/10.1016/j.jss.2025.111234
3. Chinprutthiwong, P., Kapravelos, A., & Kapadia, A. (2020). *Security study of service worker cross-site scripting.* In *Proceedings of the 36th Annual Computer Security Applications Conference (ACSAC '20)* (pp. 399–410). ACM Digital Library. https://doi.org/10.1145/3427228.3427660
4. Rajapakse, R. N., Fernando, N., & Weerasinghe, D. (2022). *Challenges and solutions when adopting DevSecOps.Information and Software Technology, 146,* 106866. https://doi.org/10.1016/j.infsof.2022.106866
5. Zhao, X., Kim, T., & Hong, J. (2024). *Identifying the primary dimensions of DevSecOps: A multi-method study. Journal of Systems and Software, 197,* 111518. https://doi.org/10.1016/j.jss.2024.111518
6. Prates, L., Gonçalves, M., & Paiva, A. (2025). *DevSecOps practices and tools: A multivocal literature review.International Journal of Information Security, 24,* 77–98. Springer. https://doi.org/10.1007/s10207-025-00777-1
7. Zakharchenko, S. M., Troianovska, T. I., Boiko, O. V., & Rybachenko, V. S. (2016). *Application of single-page web interfaces in socially significant projects. Visnyk of Khmelnytskyi National University, 3,* 33–39.
8. Troianovska, T. I., Savytska, L. A., & Taranukha, V. Y. (2017). *Methods and tools for promoting commercial web resources. Information Technologies and Computer Engineering, 2,* 23–30. Vinnytsia National Technical University.
9. Kravchuk, N., & Korobeinykova, T. (2024). *A review of secure access problems to web servers. Visnyk of Lviv State University of Life Safety, 30,* 78–89. https://journal.ldubgd.edu.ua/index.php/Visnuk/article/view/2770
10. Calzavara, S., Rabitti, F., & Bugliesi, M. (2016). *Evaluating the effectiveness of Content Security Policy. Proceedings of the 25th International Conference on World Wide Web (WWW '16)* (pp. 931-941). ACM Digital Library. https://doi.org/10.1145/2872427.2883021
11. Bennett, G., Althoff, T., & Meunier, P. (2024). *Do developers use SAST tools effectively? Proceedings of the 46th International Conference on Software Engineering (ICSE '24).* ACM Digital Library. https://doi.org/10.1145/3597503.3639147
12. Durmuşkaya, M. E., & Bayraklı, N. (2025). *Web application firewall based on machine learning models. PeerJ Computer Science, 11,* e1804. https://doi.org/10.7717/peerj-cs.1804
13. Kuryljak, A. I., & Korobeinykova, T. I. (2025, August 15). *The role of HTTP and HTTPS in the context of web application models and security assurance.* In *Proceedings of the VII International Scientific Conference "Innovative Trends of Today in the Field of Natural, Humanitarian and Exact Sciences"* (pp. 183-189). Kharkiv, Ukraine.
14. Kuryljak, A. I., Zhuravel, I. M., & Korobeinykova, T. I. (2025, August 29). *The efficiency of using SPA web applications for business and ensuring user security.* In *Intellectual Resource of Today: Scientific Tasks, Development and Questions: Proceedings of the V International Scientific Conference* (pp. 253-259). Vinnytsia: UkrLogos Group.
15. Korobeinikova, T., Maidaniuk, V., Romanyuk, O., Chekhmestruk, R., Romanyuk, O., & Romanyuk, S. (2022). *Web-applications fault tolerance and autoscaling provided by the combined method of databases scaling.* In *Proceedings of the 2022 12th International Conference on Advanced Computer Information Technologies (ACIT)* (pp. 27-32). IEEE. https://doi.org/10.1109/ACIT54803.2022.9913098
16. Korobeinikova, T., Chekhmestruk, R., Mykhaylov, P., Romanyuk, O., Romanyuk, O., & Achanyar, H. (2023). *The fault-resistant web application infrastructure using autoscaling.* In *Proceedings of the 2023 13th International Conference on Advanced Computer Information Technologies (ACIT)* (pp. 479-482). Wrocław, Poland: IEEE. https://doi.org/10.1109/ACIT58437.2023.10275448