Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя
(повне найменування вищого навчального закладу)
Факультет комп'ютерно-інформаційних систем і програмної інженерії
(назва факультету)
Кібербезпека
(повна назва кафедри)

ПОЯСНЮВАЛЬНА ЗАПИСКА

до дипломного проекту (роботи)

Магістр
(освітній ступінь (освітньо-кваліфікаційний рівень))

на тему: Аналіз методів розробки захищених ERP-систем

Виконав: студент (ка) 6 курсу, групи СБм-61

спеціальності (напряму підготовки)
125 Кібербезпека
(шифр і назва спеціальності (напряму підготовки))

(підпис) (прізвище та ініціали)

Керівник
(підпис) (прізвище та ініціали)

Нормоконтроль
(підпис) (прізвище та ініціали)

Рецензент
(підпис) (прізвище та ініціали)

м. Тернопіль – 2019

АНОТАЦІЯ

Дипломна робота на тему «Аналіз методів розробки захищених ERP-систем».

Об'єктом дослідження є процес проектування, розробки та підтримки ERP-системи із забезпеченням збереження даних.

Предметом дослідження є методи розробки захищених ERP-систем у вигляді веб-сайту.

Мета роботи – запропонувати та реалізувати спосіб захисту від несанкціонованого доступу.

Для досягнення мети в даній роботі нами було проаналізовано та вирішено ряд завдань:

1. Розглянуто список існуючих ERP-систем.

2. Проаналізовано можливі методи розробки ERP-системи.

3. Запропоновано метод захисту від несанкціонованого доступу на базі 1С:Бітрікс

За результатам досліджень було проведено порівняльний аналіз CMS на базі, яких можлива реалізація ERP-системи. Реалізовано скрипт захисту від несанкціонованого доступу на базі Bitrix Framework, що працює за рахунок порівняння IP-адреси користувача, та наявності привілейованого користувача в системі.

Ключові слова: РОЗРОБКА ERP-СИСТЕМИ, BITRIX FRAMEWORK, НЕСАНКЦІОНОВАНИЙ ДОСТУП, ЗАХИСТ КОНФІДЕНЦІЙНИХ ДАНИХ.

ANOTATION

Thesis on "Analysis of methods of development of secure ERP-systems".

The object of the study is the process of designing, developing and maintaining an ERP system to ensure data retention.

The subject of the study is the methods of developing secure ERP systems in the form of a website.

The purpose of the work is to offer and implement an additional method of protection against unauthorized access.

To achieve this goal in the work has been solved a number of problems:

1. The list of existing ERP-systems is considered.

2. Possible methods of development of ERP-System are analyzed.

4. The method of protection against unauthorized access based on 1C: Bitrix is proposed

According to the results of the studies, a comparative analysis of CMS on the basis of which ERP-system implementation is possible was carried out. A Bitrix Framework tamper-proof script is implemented that works by comparing the user's IP address and having a privileged user on the system.

Keywords: ERP-SYSTEM DEVELOPMENT, BITRIX FRAMEWORK, UNAUTHORIZED ACCESS, CONFIDENTIAL DATA PROTECTION.

ЗМІСТ

ВСТУП

Сьогодні кожне підприємство прагне зайняти становище на ринку, а для цього необхідно правильно і раціонально управляти своїм виробництвом. У сучасних умовах ефективне управління являє собою цінний ресурс організації (поряд з фінансовими, матеріальними, людськими та іншими ресурсами). Отже, підвищення ефективності управлінської діяльності стає одним з напрямків вдосконалення діяльності підприємства в цілому. Найбільш очевидним способом підвищення ефективності протікання трудового процесу є його автоматизація.

Автоматизація бізнес процесів допомагає створювати, описувати та управляти виконуваними бізнес-процесами в прикладних програмах.

В даний час вдосконалення корпоративного управління стає ключовим стратегічним завданням розвитку будь-якого підприємства.

На даний момент системи ERP знаходяться на верхньому рівні в ієрархії систем управління підприємством. Крім того, деякі підприємства поєднують в своїй роботі як безперервне виробництво, так і елементи дискретного виробництва.

Значною перевагою ERP-системи є високий коефіцієнт прискорення обробки різних фінансових обрахунків, швидкості виконання всіх замовлень, створення та формування звітів щодо прибутків та зведення фінального балансу.

Основним ефектом є оперативне прийняття рішень щодо управління на основі достовірної інформації. Завдяки тому, що всі дані лежать у єдиній базі даних.

Ядро кожної компанії - це ERP система, в ній проходять всі основні для бізнесу процеси, починаючи від закупівлі, оплати і доставки і закінчуючи управлінням людськими ресурсами, продуктами і фінансовим плануванням. Вся інформація, що зберігається в ERP-системах, має

найважливіше значення, і будь-який неправомірний доступ до неї може понести за собою величезні втрати включно аж до зупинки бізнесу.

Важливою стадією розробки ERP-системи є комплексний підхід, щодо реалізації систем захисту від несанкціонованого доступу, які будуть відповідати актуальним стандартам.

В даній дипломній роботі пропонується реалізація ERP-системи на базі 1С:Бітрікс. Дана CMS містить великий функціонал, який спрямований на захист від різноманітних атак та вразливостей.

Для захисту від несанкціонованого доступу засобами Bitrix Framework нами було реалізовано функціонал, який унеможливлює доступ сторонніх осіб до ERP-системи у позаробочий час.

1 АНАЛІЗ МЕТОДІВ РОЗРОБКИ ЗАХИЩЕНИХ ERP - СИСТЕМ

1.1 Аналіз поняття ERP-системи

У сучасному бізнесі необхідність автоматизації різних процесів стала вже звичним явищем. Вже стає складно уявити собі складський або бухгалтерський облік без застосування спеціалізованого програмного забезпечення, торгові представники використовують спеціальні програми для оформлення та відправки замовлення в офіс прямо з комунікаційних пристроїв, таких, як смартфон чи планшет.

Спеціалісти з Seosreda[1] пояснюють, що ERP-система допомагає інтегрувати всі структурні підрозділи (відділи, підрозділи, дивізіони) і їх функції в єдину систему. Важливо відзначити, що всі структурні підрозділи працюють з єдиною базою даних, що істотно спрощує процес обміну інформацією між структурними одиницями.

Для компаній та фірм, які володіють великими потужностями необхідно чітко керувати ресурсами підприємства, саме це і дозволяє реалізувати різноманітні ERP системи.

Основна задача ERP систем:

- ведення конструкторських і технологічних специфікацій, що визначають склад виробів, а також матеріальні ресурси і операції, необхідні для його виготовлення;
- формування планів продажів і виробництва; планування потреб в матеріалах і комплектуючих, термінів і обсягів поставок для виконання плану виробництва продукції;
- управління запасами і закупівлями: ведення договорів, реалізація централізованих закупівель, забезпечення обліку і оптимізації складських і цехових запасів;

- планування виробничих потужностей від укрупненого планування до використання окремих верстатів і устаткування;
- оперативне управління фінансами, включаючи складання фінансового плану і здійснення контролю його виконання, фінансовий і управлінський облік;
- управління проектами, включаючи планування етапів і ресурсів, необхідних для їх реалізації.

Отже системи планування ресурсів підприємства – служать для інтеграції всіх даних і процесів організації в єдину систему. Для цього типова ERP - система застосовує багато різних апаратних і програмних компонентів.

1.2 Акутальність ERP систем

У роботі “Аналіз сучасних систем управління ресурсами підприємства” говориться, про актуальність та переваги впровадження ERP-системи [5].

Впровадження ERP-системи дає можливість автоматизації управління дебіторською і кредиторською заборгованостями, зменшення складських запасів, калькуляції всіх видів продукції, статистичної обробки архівних даних, а також оптимізації внутрішніх бізнес-процесів, звільненні менеджерів від рутинної роботи і, як наслідок, поліпшення ефективності діяльності підприємства та підвищення конкурентоспроможності.

З метою економії матеріальних ресурсів, підвищення ефективності роботи транспорту, технологічного обладнання і устаткування можна упорядкувати завантаженість виробничих потужностей, згладити обсяги виробництва у часі, створити єдину базу даних для планування. При потребі в зовнішніх інвестиціях, приватизації, об'єднанні або поглинанні

підприємств можна підвищити прозорість фінансово-господарської діяльності та контрольованість підприємства для інвесторів або власників.

Перевагою ERP-системи є не тільки прискорення виконання певних видів робіт, наприклад, обробка замовлень, розрахунок фінансових показників, формування звіту з прибутків, зведення балансу. Основним ефектом є можливість прийняття оперативних управлінських рішень на основі повної, достовірної інформації завдяки створеній єдиній базі даних. При цьому скорочується час на виконання рутинних робіт і збільшується відповідно для аналітичної роботи. Існує можливість скорочення кількості працівників низької кваліфікації та фіксації їх дій в системі. Фінансовий ефект полягає в якісному управлінні закупками сировини і відвантаженням готової продукції, а також у зменшенні виробничих запасів відповідно до реальних потреб і вивільненні оборотних коштів.

1.2.1 Огляд популярних ERP- систем

За даними веб-сайту livebusiness[2] можна виділити основні ERP системи, яким надають перевагу користувачі.

Найбільш популярним продуктом є розробка німецької компанії SAP AG, а саме - покоління S. Самі розробники позиціонують свій проект, як ERP-система орієнтована на великі і середні підприємства. Перевагами даної системи є велика база модулів, які призначені для максимального розширення спектру можливого функціоналу програмного продукту.

Для розробки і підтримки SAP S використовують вискорівневу мову програмування ABAP/4 [3].

ABAP / 4 - пропріетарна мова програмування високого рівня, розроблена німецькою компанією SAP і призначений виключно для роботи з ПЗ, яке розробляється у межах компанії. У рейтингу популярності мов ABAP / 4 займає 22 місце прямо перед Scratch.

Мова програмування була розроблена ще в 1983 році для роботи з внутрішньою системою SAP R / 2. Внаслідок того, що всі ці роки система доопрацьовувалась, а її функціонал постійно збільшувався разом з ним і збільшувалась кількість вразливостей та слабких місць у програмних продуктах, написаних на цій мові програмування.

Другою по популярності ERP системою є Oracle NetSuite ERP.

Netsuite ERP - це програмне забезпечення ERP, яке дозволяє користувачам оптимізувати бізнес-процеси, такі як управління фінансами, закупівлі і багато іншого. Його функції також включають в себе бізнес-аналітику і масштабовану систему управління бізнесом.

Основний функціонал та усі модулі даної системи написані на Java.

Основні переваги Java:

- Робота з багатопотоковістю - велика продуктивність з високонавантаженими проектами.

- Кросплатформеність - можливість повторного використання вже існуючих фрагментів коду, для розробки мобільних додатків.

Ці фактори дозволяють розробляти гнучкі програмні продукти з великою клієнтською базою, яка не обмежується одними тільки комп'ютерами на ОС Windows чи Linux.

Dynamics 365 є серед найбільш популярних ERP систем. Даний продукт був створений у 2016 році компанією Microsoft.

Весь функціонал систем розподілений між основними модулями:

- Finance and Operations
- Retail
- Talent
- Sales
- Customer Service
- Project Service Automation
- Field Service

Великою перевагою Dynamics 365 є те, що розширювати його функціонал можна за допомогою плагінів написаних на JavaScript, який з кожним роком стає все популярнішим, та розширює сферу свого використання.

У підсумку можна сказати, що більшість популярних ERP систем є справді потужними інструментами, але через велику кількість функціоналу та сторонніх модулів система стає вразливою до хакерських атак.

1.3.2 Огляд недоліків, що становлять вразливості безпеки, притаманних ERP системам

ERP системи в більшості випадків - це високонавантажені програмні продукти, для роботи яких необхідні великі потужності. Часто для роботи таких систем використовують місця у дата-центрах, де виключається можливість недостатньої потужності машин, на яких і працюють програмні продукти.

Окрім великих вимог до характеристик комп'ютерів, ERP системи повинні завжди бути під наглядом інженерів чи програмістів. Оскільки під час збою роботи певного компонента системи можливі великі втрати фінансових ресурсів підприємства під час простою.

До недоліків можна віднести такі пункти:

- Велика кількість додаткових модулів у самій системі суттєво збільшує кількість вразливих місць. А саме, програмні інтерфейси для передачі даних між компонентами самої ERP системи.
- Висока вартість впровадження і володіння. До сих пір традиційна схема впровадження ERP систем має на увазі великі початкові витрати при впровадженні. Причому гроші треба витратити ще до того, як система запрацює, а бізнес отримає свої переваги.

- Високі ризики впровадження. Існує велика кількість складнощів при впровадженні, тут і важку спадщину у вигляді особливостей роботи старого програмного забезпечення, які треба врахувати при переході, і опір персоналу змінам, і відсутність кваліфікованих кадрів всередині підприємства, здатних організувати процес переходу і подальшої підтримки, і багато іншого. До сих пір проекти впровадження ERP систем на підприємствах залишаються одними з найбільш ризикованих для бізнесу.

- Недостатня універсальність ERP рішень. Незважаючи на те, що провідні виробники намагаються зробити свої рішення максимально гнучкими і придатними під будь-які вимоги бізнесу, зрозуміло, що практика далека від теорії. На ринку може просто бути відсутнім повністю відповідне рішення, тому часто відбувається адаптація продукту під конкретну організацію, що значно збільшує вартість проекту.

Отже до найбільших недоліків ERP систем можна віднести необхідність у потужному обладнанні, постійну присутність технічного спеціаліста для тех-підтримки продукту та суттєвому зростанню кількості можливих вразливостей зі збільшенням функціоналу продукту.

1.2 Фукнції ERP систем

Опис основного функціоналу ERP систем був описаний у роботі “Аналіз сучасних erp-систем” А.П. Власовим, С.П. Бобковим та Б.Я. Солоним [6].

ERP - «Планування ресурсів підприємства». Системи цього класу орієнтовані на роботу з фінансовою інформацією для вирішення завдань управління великими корпораціями з територіально віддаленими ресурсами. Сюди включається все, що необхідно для отримання ресурсів,

виготовлення продукції, та транспортування і розрахунків на замовлення клієнтів.

Крім названих функціональних вимог, до систем ERP пред'являються і нові вимоги по застосуванню графіки, використання реляційних баз даних, CASE технологій для їх розвитку, архітектури обчислювальних систем типу "кліент-сервер" і реалізації їх як відкритих систем. Системи цього класу активно розвиваються з кінця 80-х років;

APS (Advanced Planning / Scheduling) - "Розвинені системи планування".

З ростом потужностей обчислювальних систем, впровадженням MRPII / ERP, пошуком нових більш ефективних методів управління в умовах конкуренції з середини 90-х років на базі систем MRPII / ERP з'являються системи класу APS.

Для цих систем характерне застосування економіко-математичних методів для вирішення завдань планування з поступовим зниженням ролі календарно-планових нормативів на виробничі цикли.

У нашій країні аналогічні системи (які почали створюватися з кінця 60-х років) отримали назву типових проектних рішень (ТПР). Це АСУКунцево, «Плутон», «Сатурн», «Юпітер» та ін.

Віднесення реально створених систем до того чи іншого покоління дещо умовне. В якості критеріїв розробниками використовується міра наступних показників:

1. інтегрованість;
2. гнучкі налаштування;
3. наявність технології впровадження.

В [1] розглядаються тільки системи «вищого класу, які відрізняються високим рівнем деталізації господарської діяльності підприємства».

Зокрема дається опис 16 груп функцій системи:

1) Планування продажів і виробництва (Sales and Operation Planning).

2) Управління попитом (Demand Management).

3) Складання плану виробництва (Master Production Scheduling).

4) Планування матеріальних потреб (Material Requirement Planning).

5) Специфікації продуктів (Bill of Materials).

6) Управління складом (Inventory Transaction Subsystem).

7) Планові поставки (Scheduled Receipts Subsystem).

8) Управління на рівні виробничого цеху (Shop Flow Control).

9) Планування потреб в потужностях (Capacity Requirement Planning).

10) Контроль входу / виходу (Input / output control).

11) Матеріально-технічне постачання (Purchasing).

12) Планування ресурсів розподілу (Distribution Resource Planning).

13) Планування і управління інструментальними засобами (Tooling Planning and Control).

14) Управління фінансами (Financial Planning).

15) Моделювання (Simulation).

16) Оцінка результатів діяльності (Performance Measurement).

Декомпозиції, наведеної в роботі [7], автори роботи присвоїли статус стандарту, хоча інші розробники подібних систем далеко не завжди дотримуються подібного підходу.

Розглянемо інші ERP-системи, які є лідерами продажів. За оцінками фірми ARC Advisory Group [8] визнаним лідером є ERP-система «SAP Business All-in-One», в десятку найсильніших входить також «Microsoft Business Solutions». Основні функціональні можливості ТПР «SAP Business

All-in-One» [9] для машинобудування включають в себе:

1) управління життєвим циклом продукту;

2) підготовка виробництва;

3) управління виробництвом;

4) управління взаємовідносинами з клієнтами;

5) підтримка клієнтів;

6) бізнес-аналітика;

7) управління фінансами. Основними функціональними областями системи Microsoft Business Solutions - Navision [10] є:

1) Управління Фінансами;

2) Управління Взаємовідносинами

з Клієнтами (CRM - Customer Relationships Management);

3) Співпраця в ланцюжках поставок (SCC - Supply Chain Collaboration);

4) Персонал і зарплата;

5) Електронна Комерція.

З продуктів СНД найбільш відомо «Галузеве рішення Галактика Машинобудування» [11], яке вирішує наступні завдання:

1) Управління роботами по конструкторської та технологічної підготовки виробництва.

2) Ведення нормативної бази за складом продукції та технології виготовлення.

3) Інтеграція з PDM-системою.

4) Управління договірною діяльністю.

5) Формування виробничої програми.

6) Оцінка потреб в ресурсах (Матеріалах, обладнанні, трудових ресурсах).

7) Управління виробничими завданнями цехом.

8) Управління змінно-добовими завданнями.

9) Управління запасами.

10) Управління якістю робіт і продукції.

11) Управління технічним обслуговуванням і ремонтом обладнання.

12) Планування і облік витрат на виробництво, калькулювання собівартості продукції.

13) Підтримка прийняття рішень керівництвом. Моніторинг економічних і фінансових показників діяльності підприємства.

Для повноти уявлення наведемо опис Російської системи «Компас» [12], яка включає в себе наступні основні підсистеми:

1) Управління фінансами.

2) Документообіг.

3) Система менеджменту якості (WorkFlow).

4) Бюджетування.

5) Управлінський облік.

6) Управління закупівлями, запасами і продажами.

7) Основні фонди.

8) Облік спеціальних активів.

9) Управління персоналом (HRMсістема).

10) Кадровий облік.

11) Розрахунок заробітної плати.

12) Управління виробництвом (MRP-II).

13) Управління витратами.

14) Маркетинг і менеджмент.

Системи, представлені в [7], [8], 11], [12], істотно відрізняються один від друга і кожна не відповідає типовому уявленню про ERP-системах, даному в [1]. Хоча в рекламних заявах всіх систем, представлених в [9], [10], [11], [12], йдеться про те, що система відповідає стандарту MRP / ERP.

Перший рівень декомпозиції в системах [7], [11], [12], являє собою занадто довгий список, який важко сприймається. В [13] відзначається, що оптимальним для сприйняття вважається число 7 на кожному рівні ієрархії. Відсутня строгість у викладі таких понять як планування і управління, функції і цілі. У фундаментальній праці з менеджменту [8] з позицій системного підходу дається чітке уявлення, що планування - це одна з фаз управління. Тобто управління це більш широке поняття, яке включає в себе такі фази як планування, контроль (облік) і ін.

Оцінюючи поточний ринок ERP інструментів можна з впевненістю підвести підсумок, що у найкоротшій перспективі значних змін у ньому не передбачається. Наявне програмне забезпечення справляється практично з усіма покладеними на нього завданнями і робить це ефективно.

1.3 Огляд рішень для реалізації ERP - системи

Будь-який проект під час розробки зустрічається з складностями та проблеми, пов'язані з технологічними та організаційними питаннями.
Однак існують певні вимоги, які є важливими для успіху реалізації будь-якого проекту впровадження.

Хоч би яким було програмне забезпечення: системне, прикладне, веб-додаток або додаток для мобільних, - загальна схема розробки та її принципи однакові.

Залежно від виду, масштабів і потреб проекту визначається порядок розробки. Він буде дещо відрізнятися для розробки мобільних додатків, системного ПО, рішень для автоматизації та БД, але загальна послідовність дій для створення ПО універсальна.

Проектування. Визначивши вимоги до програмного забезпечення, розробник отримує узгоджений чіткий план дій, графік і терміни, скорочує

час розробки та підвищує її якість, а також дозволяє передбачити будь-які інші нюанси розробки, наприклад, юридичні.

Проектуючи ПО заздалегідь, розробник отримує можливість:

- оцінити вартість і час розробки програмного продукту,
- виключити втрати часу і грошей на непотрібні дії, вимушені доопрацювання, тривалий узгодження,
- уникнути розбіжностей і незадоволеності клієнта і виконавця.

Результатом проектування повинно бути готове технічне завдання у я кому будуть чітко визначені пункти щодо деталей проекту.

1.3.1 Web – застосунок та варіанти його розробки

Під web-застосунком розуміється прикладне програмне забезпечення, яке буде працювати на виділеному сервері, у якості сховища даних використовується база даних, зазвичай MySQL чи PostgreSQL.

Лідерами серед розробки web-проєків є PHP та Python.

PHP - це широко використовувана, вільно розповсюджувана і ефективна мова програмування .

Python - Це один з найшвидших мов програмування, так як він вимагає дуже мало рядків коду. Акцент робиться на зручність і простоту.

Основні причини, за якими PHP є популярною мовою програмування:

1. PHP працює на різних платформах, таких як Windows, Unix, Linux, Mac OS X і т. Д.
2. PHP сумісний практично з усіма Apache, IIS серверами
3. PHP легко вивчити, він працює ефективно на стороні сервера
4. PHP можна завантажити безкоштовно з офіційного сайту

В свою чергу Python отримує переваги у наступних пунктах:

1. У порівнянні з кодом іншої мови код Python легко писати і налагоджувати. Тому його вихідний код відносно простий в обслуговуванні.

2. Python поставляється з безліччю вбудованих бібліотек, що полегшує завдання розробки.

3. Python надає інтерактивну оболонку, яка допомагає тестувати речі до їх фактичної реалізації.

4. Python підтримує програми з графічним інтерфейсом і фреймворки для Web. Приклад: tkmter, WXPython, Django.

Підбивши підсумки порівняння PHP та Python [27], можна зробити наступний висновок: PHP і Python є потужними інструментами, які здатні виконувати одні й ті ж самі функції. Python в основному використовується для машинного навчання. В свою чергу, PHP використовується для серверних скриптів і веб-розробки [38].

1.3.2 Методи захисту ERP - систем на етапі розробки

З розвитком технологій середовище додатків стає складнішим, а безпека розробки додатків стає складнішою. Програми, системи та мережі постійно знаходяться під різними атаками безпеки, такими як шкідливий код або відмова в обслуговуванні. Деякі проблеми з точки зору безпеки розробки додатків включають віруси, трояни, логічні бомби, черв'яки, агентів та апплетів.

Програми можуть містити вразливості безпеки, які інженери програмного забезпечення спричинили навмисно або необережно.

Потрібно керувати програмним, та апаратним забезпеченням, хоча вони не можуть запобігти проблемам, що виникають через неякісний код програми.

Використання перевірок обмежень та послідовності для перевірки введення даних користувачів поліпшить якість даних. Навіть незважаючи на те, що програмісти можуть дотримуватися кращих практик, програма все одно може вийти з ладу через непередбачувані умови, і тому вона повинна успішно впоратися з несподіваними помилками, спочатку записуючи всю інформацію, яку вона може зібрати під час підготовки до аудиту.

Зазвичай програми розробляються з використанням мов програмування високого рівня, які самі по собі можуть мати вразливості безпеки. Основні заходи, важливі для процесу розробки програмного забезпечення, для створення захищених додатків і систем, включають:

- концептуальне визначення;
- функціональні вимоги;
- специфікацію управління;
- огляд дизайну;
- огляд коду та детальну інформацію;
- огляд тесту на систему та управління технічним обслуговуванням та змінами.

Побудова захищеного програмного забезпечення - це не тільки відповідальність інженера, але й відповідальність зацікавлених сторін, які включають:

1. керівництво;
2. керівників проектів;
3. бізнес-аналітиків;
4. менеджерів із забезпечення якості;
5. технічних архітекторів, спеціалістів із безпеки;
6. власників програм;
7. розробників;

1.3.3 Вимоги інформаційної безпеки робочого проекту розробки ERP - системи.

1. Механізм аутентифікації та авторизації. Цей процес включає добре розроблену систему, яка не дозволяє користувачеві змінювати ідентифікацію без повторної аутентифікації, багатофакторної автентифікації, механізму управління безпекою, авторизації ресурсів, дозволів на файли та бази даних тощо, перевірку, яка захищає будь-яке програмне забезпечення від проблем, пов’язаних з автентифікацією.

2. Перевірка даних: У життєвому циклі розробки увага завжди зосереджується на процесі перевірки даних, який включає централізовані механізми перевірки, перетворення даних у канонічну форму, використання загальних бібліотек примітивів перевірки та реалізацію типів мовного рівня для збору припущень даних. . тощо.

3. Криптографія: Криптографія є одним з найважливіших інструментів побудови захищених систем. При правильному використанні криптографії, забезпечується конфіденційність даних, захищеність даних та будь-якої інформації від несанкціонованих змін та автентифікує джерело даних.

4. Ідентифікація та обробка конфіденційних даних. Одне з найважливіших завдань - це визначити конфіденційні дані та визначити, як правильно їх захистити. Чутливість даних залежить від багатьох факторів, включаючи регулювання, політику компанії, зобов'язання щодо ініціалізації та очікування користувачів тощо. Технічна чутливість даних включає механізми контролю доступу (включаючи механізми захисту файлів, механізми захисту пам'яті та механізми захисту баз даних), криптографію для збереження конфіденційності або цілісності даних, резервні копії та резервні копії для підтримки доступності даних тощо.

5. Аналіз впливу на безпеку інтеграції зовнішніх компонентів: при інтеграції будь-яких сторонніх програм у будь-яке програмне забезпечення існує значний ризик залучення певних загроз, що супроводжують сторонні інтеграції. Тому необхідно аналізувати помилки сторонніх додатків, які можуть бути замасковані як програмні помилки, проблеми доступу між сторонніми додатками та конкретним програмним забезпеченням, несумісність між сторонніми програмами та програмними інтерфейсами тощо, щоб гарантувати, що будь-яка зовнішня інтеграція працює, як і очікувалося і не впливає на існуючі функціональні можливості програмного забезпечення.

1.3.4 Огляд вразливостей ERP - систем

Сучасний світ несе в собі тисячі загроз і потенційних небезпек буквально на кожному кроці і в кожен момент часу. Інтернет, що став невід'ємною частиною нашого життя, не є винятком.

Кіберзлочинність зараз розвинена як ніколи - адже майже кожна компанія має свій сайт в інтернеті, а зловмисник у мережі може легко залишатися анонімним.

Кількість загроз зростає пропорційно зростанню бізнесу, проте, як показує багаторічна статистика, 99% атак відбуваються через десяток стандартних помилок валідації даних, або виявлені вразливості у встановлених компонентах програмного забезпечення сторонніх виробників, або банально, через недбальство системних адміністраторів, які використовують налаштування і паролі, встановлені за замовчуванням.

Класифікацією векторів атак і вразливостей займається спільнота OWASP. Це міжнародна некомерційна організація, зосереджена на аналізі та поліпшення безпеки програмного забезпечення.

OWASP[16] створив список з 10-и найбільш небезпечних векторів атак на Web-додатки. Клієнтські компоненти ERP - систем часто працюють через web-інтернейс, які зазвичай містять у собі критичні вразливості, такі я веб форми і поля для завантаження файлів на сервер. Даний список вразливостей отримав назву OWASP TOP-10 і в ньому зосереджені найнебезпечніші уразливості, які можуть коштувати деяким людям великих грошей, або підриву ділової репутації, аж до втрати бізнесу.

1.3.4.1 Ін'єкції – Injections

Всі дані, як правило, зберігаються в спеціальних базах, звернення до яких будуються у вигляді запитів, найчастіше написаних на спеціальній мові запитів SQL[37] (Structured Query Language - структурована мова запитів).

Додатки використовують SQL-запити для того, щоб отримувати, додавати, змінювати або видаляти дані, наприклад при редагуванні користувачем своїх особистих даних або заповненні анкети на сайті. При недостатній перевірці даних від користувача, зловмисник може впровадити в форму Web-інтерфейсу додатку спеціальний код, що містить фрагмент SQL-запиту.

Такий вид атаки називається ін'єкція[17], в даному випадку найпоширеніший - SQL-ін'єкція. Це найнебезпечніша вразливість, що дозволяє зловмисникові отримати доступ до бази даних і можливість читати / змінювати / видаляти інформацію, яка для нього не призначена.

Наприклад, змінити разом з ім'ям і прізвищем баланс свого рахунку, подивитися баланс чужого рахунку, або ж, викрасти конфіденційні особисті дані.

Ця вразливість є наслідком недостатньої перевірки даних, що надходять від користувача. Це дозволяє зловмисникові «підсунути»,

наприклад, в веб-форми, спеціально підготовлені запити, які «обдурять» додаток і дозволять прочитати або записати нелегітимні дані.

В цілому цей різновид атак має загальну назву «Помилки валідації», до неї відносяться далеко не тільки SQL-ін'єкції і ми будемо згадувати цей вектор ще не раз.

1.3.4.2 Недоліки системи аутентифікації і зберігання сесій

Для того, щоб відрізняти одного користувача від іншого, web-додаток використовує так звані сесійні куки. Після того, як Ви ввели логін і пароль і додаток вас авторизуватися, в сховище браузера зберігається спеціальний ідентифікатор, який браузер надалі пред'являє серверу при кожному запиті сторінки вашого web-додатку. Саме так web-додаток розуміє, що Ви - це саме Ви.

У разі, якщо ваш ідентифікатор вкраде зловмисник, а в системі не були реалізовані перевірки, скажімо IP-адреси сесії, або перевірки наявності більш одного з'єднання в одній сесії, зловмисник зможе отримати доступ до системи з правами вашого облікового запису. А якщо це інтернет-банк або кабінет платіжної системи, про наслідки такого несанкціонованого доступу Ви можете легко здогадатися самі.

1.3.4.3 Міжсайтовий скриптинг - XSS

Згідно статті “Cross-Site Scripting (XSS) Detection Integrating Evidences in Multiple Stages” [17] можна зробити висновок, що міжсайтовий скриптинг - ще одна помилка валідації призначених для користувача даних, яка дозволяє передати JavaScript код на виконання в браузер користувача. Атаки такого роду часто також називають HTML-ін'єкціями, адже механізм їх впровадження дуже схожий з SQL-ін'єкціями, але на відміну від останніх, впроваджуваний код виповнюється в браузері користувача. Чим це загрожує?

По-перше, зловмисник може вкрасти вашу сесійний cookie, наслідки чого були описані в другому пункті, буквально парою абзаців вище. Потрібно відзначити, що далеко не всі сервери додатків уразливі до даного типу атак, про це ми окремо поговоримо в пункті під номером 5.

По-друге, можуть бути вкрадені дані, що вводяться у форми на зараженій сторінці. А це можуть бути конфіденційні персональні дані, або, що ще гірше, дані кредитної картки разом з CVC-кодом.

По-третє, через JavaScript можна змінювати дані, розташовані на сторінці, наприклад, там можуть бути реквізити для банківського переказу, які зловмисник із задоволенням підробить і замінить підставними.

1.3.4.4 Небезпечні прямі посилання на об'єкти

Даний вид уразливості є також наслідком недостатньої перевірки призначених для користувача даних. Суть її полягає в тому, що при виведенні будь-яких конфіденційних даних, наприклад особистих повідомлень або облікових карток клієнтів, для доступу до об'єкта використовується ідентифікатор, який передається у відкритому вигляді в адресному рядку браузера, І не реалізована перевірка прав доступу до об'єктів. Наприклад, є сторінка, яка відображає приватне повідомлення і вона має адресу виду: track-in.ua/catrgory?page=12

Перебираючи число після "id =" можна буде читати чужі приватні повідомлення. Експлуатація даної уразливості дуже проста і не вимагає взагалі ніяких спеціальних навичок - достатньо лише перебирати число в адресному рядку браузера і насолоджуватися результатом. Хоч би як парадоксально, але до цієї "дитячої хвороби" , часом були схильні до досить великі європейські платіжні системи.

1.3.4.5 Небезпечна конфігурація

Безпека Web-додатки вимагає наявності безпечної конфігурації всіх компонентів інфраструктури: компонентів програми (таких як фреймворки - frameworks), веб-сервера, сервера баз даних і самої платформи. Налаштування компонентів сервера за замовчуванням найчастіше небезпечні і відкривають можливості до атак. Наприклад, крадіжка сесійного cookie через JavaScript при XSS-атаки стає можливою завдяки виключеною за замовчуванням налаштування cookie_http only.

При правильному налаштуванні сервера і включеній опції cookie_httponly, отримати сесійний cookie через JavaScript неможливо, але часто ця проста і важлива настройка була відсутня в таких критично важливих місцях, як особисті кабінети платіжних систем.

Ще один приклад "дитячої" вразливості - використання налаштувань за замовчуванням в серверах баз даних, таких як Redis, Memcached і інших - закрита служба може бути доступна на публічній IP-адресі сервера, і / або використовувалися паролі, встановлені виробником за замовчуванням. Це дозволяє зловмисникові запросто читати і змінювати дані, серед яких, нерідко бувають і сесійні cookies (які можуть бути наслідки - ми вже знаємо) і виводяться користувачам в браузер дані (що дозволяє ще й XSS-атаку застосувати).

Крім того, програмне забезпечення повинно бути в актуальному стані: уразливості знаходять кожен день в самих різних програмних компонентах - операційній системі, web-серверах, серверах баз даних, поштових серверах і т.д. І навіть якщо ваш додаток правильно написано і ретельно перевіряє всі вхідні дані, і взагалі, добре захищене, це не означає що в один прекрасний момент не знайдеться вразливість у вашій ОС або Web-сервері.

1.3.4.6 Незахищеність критичних даних

Багато веб-додатки не захищають конфіденційні дані, такі як кредитні карти і облікові дані для аутентифікації. Зловмисники можуть вкрасти або модифікувати такі слабо захищені дані для використання в своїх корисливих цілях.

Найпростіший приклад - передача даних по протоколу HTTP. Справа в тому, що дані передаються по протоколу HTTP ніяк не зашифровано, а при проходженні даних від комп'ютера користувача до Web-сервера, дані пройдуть досить багато різних вузлів: маршрутизатор офісу або домашній роутер, маршрутизатор провайдера, маршрутизатор на каналі, маршрутизатор в дата- центрі хостинг-провайдера сервера і так далі. На кожному з цих вузлів може зачаїтися так званий сніффер, програма, яка зчитує весь трафік і передає зловмисникові. А останній переглядає отримані дані на предмет персональних даних та даних кредитних карт.

Такі дані повинні передаватися виключно за протоколом HTTPS, про що повинен свідчити відповідний напис в адресному рядку браузера.

Ще одне завдання SSL-сертифіката (а саме так називається спеціальний ключ, за допомогою якого здійснюється перевірка справжності та шифрування в HTTPS) - підтвердити, що він виданий саме для даного сайту. У разі, якщо сертифікат прострочений або підроблений, побачите наступну картину, рисунок 1.1

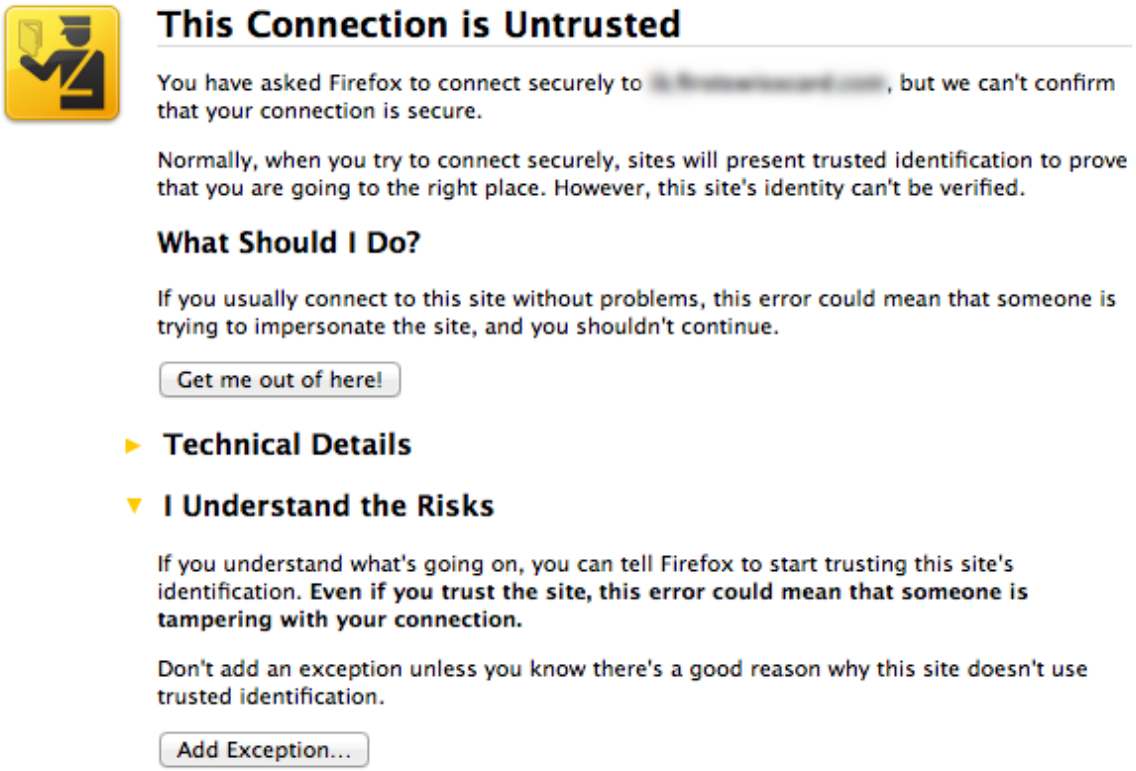


Рисунок 1.1 – Відсутність SSL на сайті

Інший приклад - відсутність шифрування критичних даних, таких як паролі або номери кредитних карт. У разі, якщо дані зашифровані, то навіть у разі отримання несанкціонованого доступу на сервер, зловмисник не зможе вкрасти критичні дані. До паролів, зокрема, повинна застосовуватися необоротна хеш-функція - розшифрувати шифрограму при цьому не можливо і перевірка пароля відбувається шляхом формування шифрограми введеного пароля і порівняння її з наявною в базі.

1.3.4.7 Відсутність функцій контролю доступу

Суть уразливості, як випливає з назви, полягає у відсутності перевірки наявності належного доступу до запитуваного об'єкту.

Більшість веб-додатків перевіряють права доступу, перш ніж відобразити дані в інтерфейсі. Проте, додатки повинні виконувати ті ж

перевірки контролю доступу на сервері при запиті будь-якої функції. Адже є ще безліч допоміжних прохань про надання послуг, які, найчастіше відправляються в фоновому режимі асинхронно, за допомогою технології AJAX.

Якщо параметри запиту не досить ретельно перевіряються, зловмисники зможуть підробити запит для доступу до даних без належного дозволу.

Найпоширеніший випадок даної уразливості - відсутність перевірки користувача в особистих повідомленнях.

1.3.4.8 Міжсайтовий підробка запиту

Вектор атаки CSRF[19], також відомий як XSRF, дозволяє зловмиснику виконувати від імені жертви дії на сервері, де не реалізовані додаткові перевірки.

Наприклад, в деякій платіжній системі для переказу коштів на інший рахунок, є сторінка виду:

demobank.com/transfer_money.jsp?transfer_amount=1000&transfer_account=123456789

де transfer_amount - сума для прикладу і transfer_account - номер аккаунта, куди повинні бути переведені кошти.

Якщо жертва заходить на сайт, створений зловмисником, від її особи таємно відправляється запит на вищевказану сторінку платіжної системи. Як результат - гроші підуть на рахунок зловмисника, після чого, ймовірно, будуть оперативно обміняні на Bitcoin або переведені в іншу безповоротну платіжну систему, і отримати їх назад вже не вийде.

Передбачається, що жертва повинна була попередньо пройти аутентифікацію в платіжній системі і повинна бути відкрита активна сесія (скажімо, сторінка платіжної системи відкрита в іншій вкладці браузера).

1.3.4.9 Використання компонентів з відомими вразиливостями

Найчастіше web-додатки написані з використанням спеціальних бібліотек або «фреймворків» (англ - framework), які поставляються сторонніми компаніями. У більшості випадків ці компоненти мають відкритий вихідний код, а це означає, що вони є не тільки у вас, але і у мільйонів людей у всьому світі, які студіюють їх вихідний код, в тому числі, і на предмет вразливостей. І потрібно відзначити, що роблять вони це аж ніяк не безуспішно.

Також уразливості шукають (і знаходять) в більш низькорівневих компонентах системи, таких як сервер бази даних, web-сервер, і нарешті, компоненти операційної системи аж до її ядра.

Дуже важливо використовувати останні версії компонентів і стежити за описом нових виявлених вразливостей на сайтах типу securityfocus.com.

1.3.4.10 Неперевірені переадресації та пересилання

Web-додатки часто переадресовують користувача з однієї сторінки на іншу. В процесі можуть неналежним чином перевірятися параметри із зазначенням сторінки кінцевого призначення переадресації.

Без відповідних перевірок, атакуючий може використовувати такі сторінки для переадресації жертви на підроблений сайт, який, наприклад, може мати дуже схожий інтерфейс, але вкраде ваші дані кредитної картки або інші критичні конфіденційні дані.

Цей вид вразливостей є різновидом помилок перевірки вхідних даних (input validation).

2 МЕТОДИ РОЗРОБКИ ТА ЗАХИСТУ ERP-СИСТЕМ

За результати дослідження В.В. Голян та О.К. Кравченко було представлено декілька варіантів життєвих циклів програм[22]. Найбільш придатними до використання у нашій роботі є модель водоспаду, модель Agile та RAD.

У своїй статті О.К. Кравченко описує водоспадну модель наступним чином “Модель водоспаду розвиває програмне забезпечення поступово операційний аналіз, експлуатаційні специфікації, специфікації проектування і кодування, розробка, тестування, розгортання, оцінка”. Прикладом даної моделі життєвого циклу програмного забезпечення є рисунок 2.1.

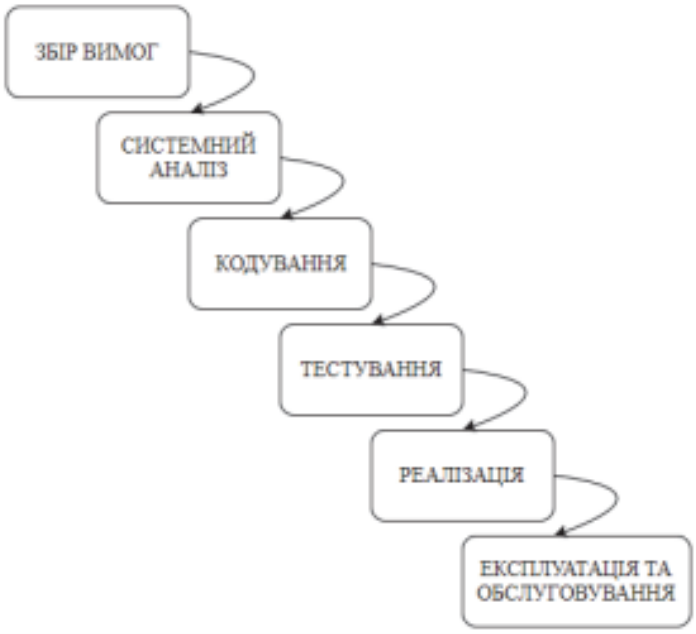


Рисунок 2.1 - Модель водоспаду. Джерело: [30].

У ітеративній моделі процес ітерацій починається з реалізації невеликого набору простих вимог і поступово покращує нові версії, поки не буде готова і реалізована повноцінна робоча система, рисунок 2.1.

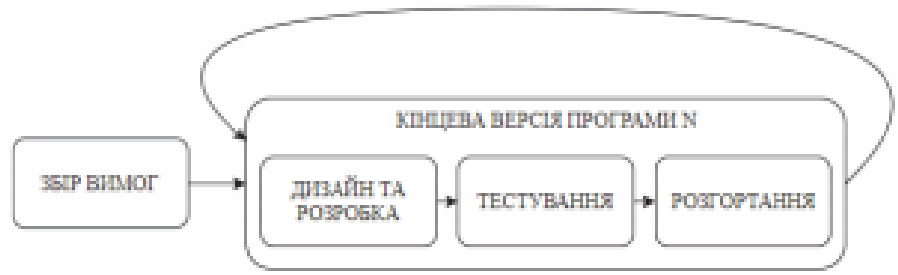


Рисунок 2.2 - Модель Agile. Джерело : розроблено В.В. Голяном.

В.В. Голян у своїй статті [21] трактує модель типу RAD наступним чином “Модель RAD – модель швидкої розробки додатків. Це тип інкрементної моделі. В моделі RAD компоненти або функції розробляються паралельно, так якщо б вони були міні-проектами. Розробки є коробкою часу, доставляються і потім збираються в робочий прототип”, дана модель представлена на рисунку 2.3

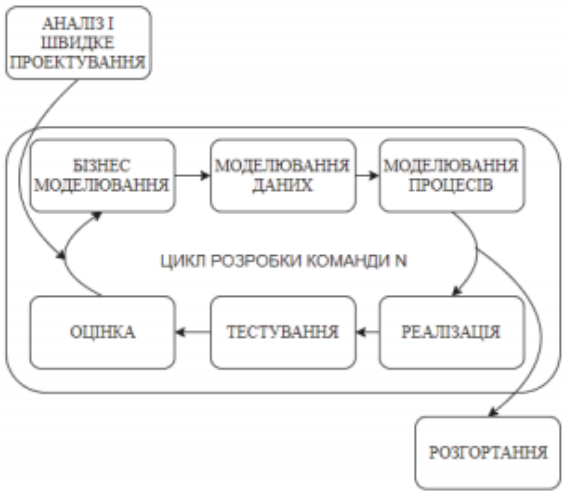


Рисунок 2.2 - Модель RED. Джерело: [21]

Для нашої роботи найбільш ефективним буде модель водоспаду, оскільки даний алгоритм розробки є ефективним та не потребує повторного аналізу та тестування функціоналу програмного забезпечення, що у свою чергу призводить до суттєвого збільшення вартості розробки проекту.

Після збору вимог необхідно спроектувати майбутній проект урахувавши усі нюанси, які пов’язані зі збереженням конфіденційної інформації та унеможливити її витоки за межі даної ERP-системи.

2.1 Написання проекту без використання зовнішніх бібліотек

Найголовнішим плюсом самописних проектів є суттєва перевага у гнучкості. Жодна CMS не дозволить створити настільки гнучку систему. Наприклад, при створенні ERP-системи потрібно реалізувати можливість розрахунку транспортних задач для оптимізації роботи логістичного центру, в CMS це зробити з коробки не вийде, а в своєму самописному проекті всі необхідні варіанти можна передбачити заздалегідь [22].
Ще одним важливим фактором на користь самостійної розробки є оптимізація швидкості роботи. В самописному проекті немає непотрібного функціоналу, а отже, скрипти будуть працювати швидше, і оптимізувати їх буде значно легше.

Мінуси у самописних движків теж присутні. Перший і основний - ціна розробки.

Для прикладу, реалізація класу з методами для роботи з базою даних може зайняти більше 30 годин роботи програміста, при тому не виключено, що при реалізації даного функціоналу не буде допущено критичної помилки у валідації даних.

Прикладом валідації даних може стати код наведений у лістингу 2.1

Лістинг 2.1 - Приклад класу для валідації даних

```
class validate{
    var $data = array();

    function naughty_checks($data){
        $this->data = $data;
    }

    function field_empty($field){
        if (!isset($this->data[$field]) || strlen(trim($this-
>data[$field])) == 0){
            return true;
        }else{
            return false;
        }
    }

    function field_numeric($field){
        if (!isset($this->data[$field]) || intval(trim($this-
>data[$field])) < 1){
            return true;
        }else{
            return false;
        }
    }


function dangerChars($field){
        if (strstr(';', $this->data[$field]) || strstr('\'',
$this->data[$field]) || strstr('"', $this->data[$field])){
            if( $this->data[$fields[strstr(';', $this-
>data[$field])-1]-1] != '\\' ||
$this->data[$fields[strstr('\'', $this->data[$field])-1]-1] !=
'\\' ||
$this->data[$fields[strstr('"', $this->data[$field])-1]-1] !=
'\\'){
return false;
}

        }else{
            return true;
        }
    }
}
```

Якщо ж допустити помилку при створенні валідатора то в подальшому за допомогою цієї вразливості зловмисник зможе реалізувати

SQL-ін’єкцію, що в майбутньому з великою ймовірністю приведе до втрати дорогоцінних даних та фінансових збитків компанії.

Також підтримка таких проектів суттєво складніша, якщо порівнювати з проектами реалізованих на популярних фреймворках чи CMS.

У більшості випадків для популярних фреймворків та CMS передбачено документації. Звідси з’являється ще один мінус, а саме те, що будь-які двигуни після своєї розробки та запуску проекту вимагають оновлень, наприклад, для додавання нових функції, не компетентні розробники, або ті, які не були залучені до розробки даного ПЗ можуть відчути складнощі при роботі з ядром[23].

Отже у підсумку можна сказати, що переваги самописної системи полягають у:

- Відсутності надлишкового коду та функціоналу;
- Оптимізованості запитів;
- До недоліків у свою чергу можна віднести:
- Складність підтримки декількома розробниками;
- Затрати часу на реалізацію;
- Зі збільшенням функціоналу зростає можливість допущення розробником критичної вразливості;

2.2 Реалізація проекту з підключенням зовнішніх бібліотек

Одним з методів розробки ERP - системи є використання зовнішніх бібліотек [45], у яких вже передбачено функціонал для роботи з певним сегментом задач. Для більшості типових задач, які притаманні веб-проектам, вже є готові рішення у вигляді бібліотек на які можна покласти роботу з БД, дії з поштою, роботу із зображеннями, графіками тощо.

Для роботи з базами даних можна виділити 3 основні бібліотеки, а саме:

1. Doctrine ORM
2. Query Builder

3. ADOdb

Приклад SQL запиту наведено у лістингу 2.2

Лістинг 2.2 - Приклад запиту на вибірку кількості товарів на складі

```
public function getCategoriesByProduct($product_id) {
  $sql = "SELECT *
  FROM `product` sba
  LEFT JOIN `product_storage_rel`
ON(product.product_id=product_storage_rel.product_storage_stoc
k_id)
  LEFT JOIN `product_storage_stock_id`
ON(product_storage_rel.product_id =
storage_stock.product_count)
  WHERE 1";
  return $query->rows;
}
```

2.2.1 Doctrine ORM

В основі Doctrine лежать патерни і абстракції, розуміння яких допоможе краще з'ясувати принципи роботи цієї ORM. Почнемо, мабуть, з найголовнішого - Data mapper, оскільки Doctrine в загальному вигляді і є реалізація цього патерну, приклад наведений на рисунку 2.3.

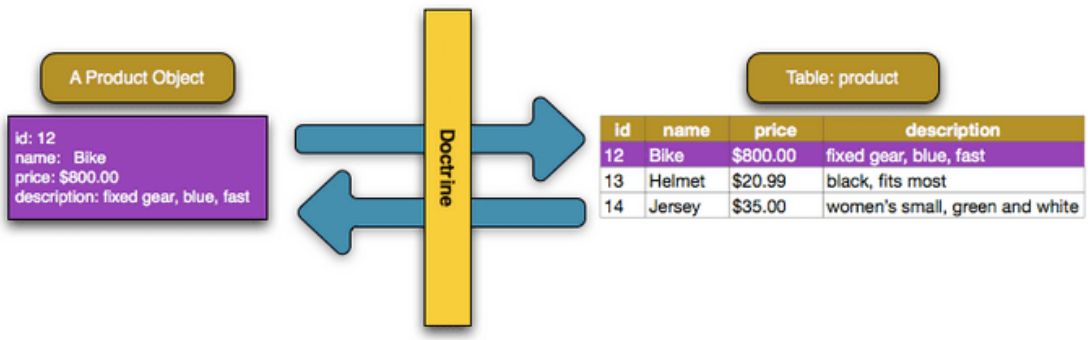


Рисунок 2.3 - Приклад роботи Doctorine ORM [32].

Зрозуміло, що відсутність необхідності вручну писати сотні SQL-запитів спрощує процес розробки[46], особливо в великих проектах. У той же час, запити, які генеруються бібліотекою, складніше оптимізувати, та й сама бібліотека додає оверхед.

У розробника без ORM просто немає іншого виходу, як писати однотипні, повторювані SQL-запити для всіх об'єктів програми. Це вкрай незручно і складно в підтримці, а значить, дорого.

Data mapper вирішує цю задачу за рахунок ізоляції об'єктів і БД відносно один одного і в якості основної, концептуальної абстракції використовує Entity[47].

За фактом, Entity{}[48] - це звичайний PHP клас, де властивості зіставлені з полями таблиці з бази даних, для якої і створювалася Entity (далі «сутність»). Всю ж роботу по Маппінг полів, обчисленню змінних тощо бере на себе mapper.

Приклад SQL Запиту з використанням Doctorine[49] ORM наведено у лістингу 2.3, який було написано нами для порівняння синтаксису, зручності, та безпеки.

Лістинг 2.3 - Варіант використання вибірки у Doctorine ORM.

```
public function getHistory($users) {
    $qb = $this->entityManager->createQueryBuilder();
    $qb
        ->select('*',)
        ->from('Credit\Entity\UserCreditHistory', 'product')
        -
>leftJoin(array('User\Entity\User','product',\Doctrine\ORM\Que
ry\Expr\Join::WITH,'product.product_id =
product_storage_rel.product_storage_stock_id'),
          array(
```

```
            'User\Entity\User',
            'product_storage_rel',
            \Doctrine\ORM\Query\Expr\Join::WITH,
            'product_storage_rel.product_storage_stock_id =
storage_stock.product_count'),

            )
        ->where('1'));
    return $qb->getQuery()->getResult();
}
```

Саме по собі використання ORM не є засобом захисту від ін'єкцій, але при правильному використанні бібліотеки надають засоби для параметризованих і підготовлених запитів.

2.2.2 Laravel Query Builder

Конструктор запитів до баз даних Laravel забезпечує зручний, вільний інтерфейс для створення та запуску запитів до бази даних. Він може використовуватися для виконання більшості операцій з базою даних у вашій програмі та працює у всіх підтримуваних системах баз даних. Прикладом вибірки за допомогою Laravel Query Builder є лістинг 2.4. Запит був написаний нами, для наведення прикладу звернення до БД

Лістинг 2.4 - Вибірка залишку товарів за допомогою Laravel Query Builder.

```
$users = DB::table('product')
            ->leftJoin('product_storage_rel',
'product.product_id', '=',
'product_storage_rel.product_storage_stock_id')
```

```
            ->leftJoin('product_storage_stock_id',
'product_storage_rel.product_id', '=',
'storage_stock.product_count')
            ->get();
```

Конструктор запитів Laravel використовує прив'язку параметрів PDO для захисту програми від атак ін'єкцій SQL. Не потрібно чистити рядки, що передаються як прив'язки.

Отже, використання Laravel Query Builder забезпечує зручність розробникам завдяки лаконічному синтаксису, та забезпечує захист від sql-ін’єкцій завдяки прив’язці параметрів PDO.

2.2.3 ADOdb

ADOdb - це швидкий, простий у користуванні, популярний шар абстракції бази даних для PHP. Це дозволяє використовувати той самий код під час доступу до широкого спектру баз даних. Він активно підтримується з 2000 року засновником проекту та численними учасниками громади. ADOdb містить компоненти для запитів та оновлення баз даних, а також об'єктно-орієнтовану бібліотеку активних записів, управління схемами та моніторинг продуктивності. Він також містить такі самостійні розширення:

1. Бібліотека дати / часу для обробки дат поза межами звичайних PHP.
2. Бібліотека управління сеансами, яка розширює звичайну функціональність PHP, щоб дозволяти зберігати дані управління сеансами в базі даних або в зашифрованих значеннях

Зауважте, що ADOdb не є заміною для рідних розширень баз даних PHP, але побудований поверх них. Це означає, що відповідні драйвери повинні бути встановлені та правильно налаштовані для роботи ADOdb.

При роботі з БД за допомогою ADOdb розробнику необхідно самому піклуватись про валідацію даних, екранацію небезпечних символів та фільтрацію даних.

Отже на прикладі бібліотек для роботи з базами даних було продемонстровано варіанти використання функціоналу. Справді ефективним можна рахувати Laravel Query Builder, оскільки крім зручності для розробника у вигляді зручного синтаксису присутні ефективні методи захисту від sql ін'єкцій.

2.3 Використання CMS/фреймворків для реалізації проекту

З CMS все доволі зрозуміло: широкий функціонал з коробки, багатий набір як платних, так і безкоштовних модулів і плагінів, швидка установка і настройка сайту.

Різниця починає з'являтись, коли виникає потреба змінити / доповнити функціонал веб-проекту. Основна перевага використання фреймворка в розробці - гнучкість і широкі можливості.

Використання фреймворка не накладаються на розробника обмежень, що існують в розробці під CMS. Фреймворки функціонують однаково, позбавляючи вас від необхідності постійно створювати одні й ті ж методи та класи.

2.3.1 Аналіз фреймворку Laravel

Laravel - це безкоштовний PHP фреймворк з відкритим вихідним кодом, створений Тейлором Отвеллом[26] для розробки веб-додатків за архітектурним шаблоном MVC.

Він був створений як альтернатива такому фреймворку, як Codeigniter, в якому було недостатньо корисних функцій для розробки веб-

додатків. В якості основи Laravel виступають компоненти іншого фреймворка - Symfony.

А тепер розглянемо можливості самого PHP-фреймворку Laravel.

Після виходу PHP7 в порівнянні з PHP5, порівняння зображено на рисунку 2.4, скрипти стали швидше і почали використовувати набагато менше оперативної пам'яті, а в зв'язці з Zend OPCache показують чудові результати. Зокрема сервіс Laravel Forge налаштовує Zend OPCache для досягнення максимальної продуктивності.

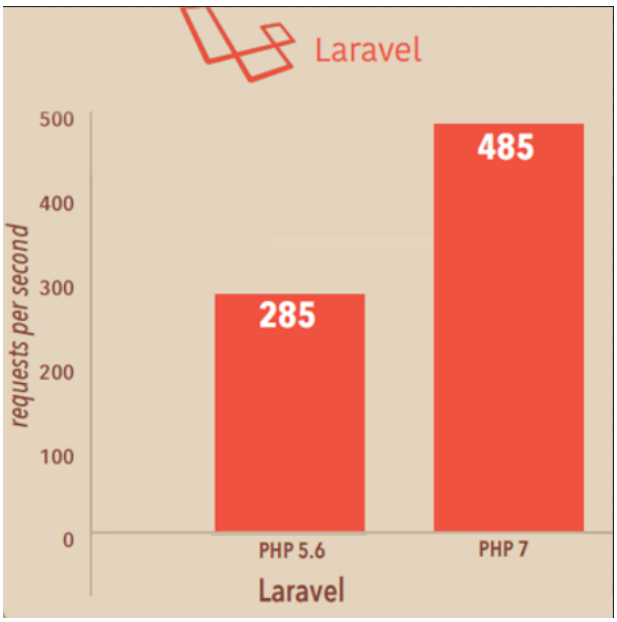


Рисунок 2.4 - Порівняння продуктивності фреймворку на різних версіях PHP

Саме тому, коли йде мова про продуктивність того чи іншого PHP-фреймворку, то завжди проводять тестування без кешування, роботи з БД або файлами, в основному роблячи мільйони викликів до звичайної PHP сторінки. В цьому плані даний PHP-фреймворк істотно нічим не відрізняється від всіх інших, але коли мова йде про масштабованості, гнучкості, універсальності вбудованих механізмів кешування і швидкості розробки, саме тоді Laravel показує всю свою гнучкість і переваги.

Згідно статті “A comparative study of laravel and symfony PHP frameworks” [27] було зроблено підсумок ключових особливостей фреймворку laravel.

PHP-фреймворк побудований за архітектурою MVC на базі відомих і надійних компонентів Symfony.

Необхідні модулі для фреймворка підключаються у вигляді пакетів-провайдерів. У версії Laravel 5.5 досить просто встановити пакет через Composer, і він відразу буде доступний, без необхідності підключення сторонніх модулів у самому коді.

Міграції для бази даних, розробник можете змінювати структуру БД і відмінювати зміни структури.

Черги завдань, планувальник завдань, консоль, робота з SSH.

Підтримка WebSockets для створення справжніх інтерактивних додатків.

Інтерфейс командного рядка artisan, який дозволяє генерувати моделі, контролери, повідомлення, запускати завдання з черги завдань і багато іншого.

Laravel Tinker - додатковий пакет, який дозволяє працювати з кодом проекту з командного рядка.

Величезні можливості для тестування веб-проекту, в тому числі заповнення бази даних тестовими даними.

Одним з найважливіших модулів даного фреймворку є масивний функціонал Eloquent ORM, який дозволяє повністю убезпечити себе від атак типу SQL Injection, а також завантажувати дані з декількох таблиць або ж обробляти дані з БД частинами.

Цей патерн є колекцією. Колекцію сутностей, які може отримувати, змінювати, зберігати, видаляти, в загальному, управляти ними в абстрактному місці зберігання. У більшості випадків, таким місцем

зберігання є різні бази даних. Але може бути і файлова система і навіть зовнішнє API.

Такий підхід повністю відповідає принципу єдиної відповідальності і підходу DDD. Крім того, завдяки цьому підходу, реалізується слабка зв'язаність - нам не важливо, яким саме чином в додатку зберігається щось, ми працюємо з Entity, коли хочемо працювати безпосередньо з об'єктним поданням даних і працюємо з Repository, коли нам потрібно взаємодіяти зі сховищем.

Підсумувавши все вище сказане, можна зробити наступний висновок. laravel є потужним фреймворком, з великою кількістю модулів для розширення штатного функціоналу, а також системою з передбаченим функціоналом для захисту від SQL ін’єкцій. Проте при роботі з великою кількістю зовнішніх колекцій чи сутностей виникає проблема для взаємодії з їхніми моделями, що суттєво зменшує продуктивність проекту.

2.3.2 Аналіз фреймворку Symfony

Symfony - другий за популярністю PHP-фреймворк. Він побудований на основі патерну проектування MVC, у вигляді штатного шаблонізатора використовується Twig.

Для роботи з БД застосовується Doctrine Object Relation Mapper, що надає потужний Dependency Injection Container, включає в себе парсер конфігурації з форматів XML і YAML, конструктор легко валідує форми, інструменти для тестування, кешування, роботи з мультимовну, а також продуману Security-компоненту для роботи з аутентифікацією і авторизацією користувачів.

Стандартна, рекомендована до встановлення збірка Symfony з коробки забезпечує захист від більшості загроз, актуальних для вебу на

сьогоднішній день. У даному розділі ми зробимо огляд основних механізмів, що роблять розробку на Symfony безпечною.

Базова задачі шаблонізації - це екранування змінних спецсимволов HTML, тобто перетворення використовуваних мовою розмітки спецсимволов в безпечні еквівалентні конструкції.

Висновок в шаблоні введених користувачем даних без правильного екранування породжує загрозу XSS-атаки. У зловмисника з'являється можливість експлуатувати шкідливий код на стороні інших клієнтів.

Прикладом небезпечного коду є лістинг 2.5.

Лістинг 2.5 - Небезпечний варіант виводу інформації

```
<?php
     echo $input
?>
```

Однією з альтернатив для є використання функції htmlspecialchars(), що є стандартним функціоналом у PHP.

Проте у Symfony передбачений власний метод, приклад якого наведений у лістингу 2.6, а саме використання шаблонізатора, який виводить повністю екранований текст.

Лістинг 2.6 - Приклад безпечного виводу інформації [35].

```
{{Input}} # виведе екрановану змінну
{{Input | raw}} # виведе неекрановану змінну
```

Для роботи з БД Symfony використовує вище згадувану бібліотеку Doctrine.

Такий підхід виключає появу в запиті несподіваних конструкцій. У разі порушення будь-яких структурних правил побудови запиту бібліотека

видає помилку на рівні PHP, а не на рівні синтаксичного аналізатора SQL. Це ізолює дані від помилкових запитів.

У стандартну збірку Symfony також інтегрована бібліотека Doctrine ORM, яка дозволяє працювати не з самим SQL-запитом або його конструктором, а безпосередньо з PHP-об'єктами. Класи моделей розмічаються певним чином (наприклад, за допомогою анотацій в phpDoc), в результаті чого властивості об'єктів проектуються на колонки таблиць в бд. Взаємодія між об'єктом і рядком в бд (створення, редагування, видалення) відбувається автоматично через налагоджені механізми, що крім підвищення швидкості розробки зводить до мінімуму можливість помилки.

Невід'ємною частиною веб-додатки є форми. PHP є слабо універсальна мова, тому серверна валідація та приведення вхідних даних до типів є особливо гострим питанням.

Валідація в Symfony інтегрована в саме ядро механізму конструювання форм. Наприклад, при оголошенні поля типу (або групи радіокнопок) обов'язковим є вказівка списку опцій. Отримавши заповнену форму, фреймворк автоматично перевіряє дані на відповідність пропонувалися опцій, прикладом реалізації функціоналу є лістинг 2.7.

Крім того, Symfony дозволяє розмічати моделі (властивості яких населяються входять даними після обробки запиту) різними валідаторами, а потім за пару рядків коду отримувати масив помилок.

Лістинг 2.7 - Механізм валідації даних з веб-форм [34].

```
$builder->add('gender', ChoiceType::class, array(
    'choices'  => array(
        'Мужской' => 'male',
        'Женский' => 'female',
    ),
));
```

У Symfony за замовчуванням включена захист від CSRF-атак. У всі форми автоматично додається, унікальний для кожного користувача ключ. При отриманні відповіді, фреймворк насамперед звіряє отриманий токен з токеном з сесії.

Ми розглянули основні інструменти фреймворка Symfony, що дозволяють вирішити головні питання безпеки веб-додатки. Стандартна збірка Symfony передбачає захист від XSS- і CSRF-атак, SQL-ін'єкцій, включає інструменти швидкої і зрозумілою валідації форм, механізми аутентифікації і авторизації.

2.3.3 Аналіз фреймворку YII 2

Фреймворк реалізує парадигму MVC[51] і підходить для розробки додатків будь-якої складності, особливо якщо мова йде про великий проект: форуми, інтернет-магазини, портали чи не стандартні веб-проекти.

Yii має хорошу документацію. Вона довгий час була англомовна, але на разі ведеться переклад документації на українську.

Yii можна сміливо назвати одним з лідируючих за популярністю фреймворком.

Переваги Yii - це висока продуктивність та швидкість роботи і хороша підтримка ООП[52]. Yii включає в себе велику кількість бібліотек. Завдяки їм можна без підключення зовнішніх сервісів створити веб-додаток, яке відповідатиме всім сучасним стандартам. Вбудовані методи дозволяють значно скорочувати кількість коду.

Фреймворк Yii відомий великою кількістю співтовариством розробників, в тому числі українською мовою. Великі спільноти розробників - це можливість оперативно отримати допомогу і обговорити важливі теми.

Проте Yii не можна назвати проектом однієї людини[53], так як зараз фреймворк підтримує і розвиває велика команда. Вона стежить за основними IT-тенденціями і впроваджує їх у проект.

Yii має хороший генератор коду. Він згенерує вихідний код, або ж структуру програми, в залежності від параметрів, які вкаже розробник.

Важливим моментом є те, що Yii сприяє швидкому прототипуванню проекту[54]. Це суттєво пришвидшує розробку на ранніх стадіях, та дає змогу зрозуміти, що певний компонент системи справді може бути реалізованим і виконувати свої функції. Після прототипування настає стадія робочої реалізації.

Що до безпеки продукту, як і для будь-якого проекту необхідно приділяти увагу до фільтрації введення та екранування виводу.

Фільтрація введення означає, що вхідні дані ніколи не повинні вважатися безпечними і ви завжди повинні перевіряти, чи є отримані дані допустимими[55]. Наприклад, якщо ми знаємо, що сортування може бути здійснена тільки за трьома полями title, created_at і status, і поле може передаватися через введення користувачем, краще перевірити значення там, де ми його отримали, прикладом реалізації даної валідації є лістинг 2.8.

Лістинг 2.8 - Перевірка валідності фільтра [35].

```
$sortBy = $_GET['sort'];
if (!in_array($sortBy, ['title', 'created_at', 'status'])) {
    throw new Exception('Invalid sort value.');
}
```

Екранування виведення означає, що дані в залежності від контексту повинні екрануватися, наприклад в контексті HTML ви повинні екранувати “<”, ”>” і схожі спеціальні символи. В контексті JavaScript або SQL буде

інший набір символів. Так як ручне екранування значною кількістю помилок, Yii надає різні утиліти для екранування в різних контекстах.

SQL-ін'єкції відбуваються, коли текст запиту формується склеюванням НЕ екранованих рядків, як показано у лістингу 2.9

Лістинг 2.9 приклад SQL-ін’єкції

```
$username = $_GET['username'];
$sql = "SELECT * FROM user WHERE username = '$username'";
```

Це валідний запит, який спочатку буде шукати користувачів з порожнім ім'ям, а потім видалить таблицю user. Швидше за все буде зламано додаток і будуть втрачені дані.

Більшість запитів до бази даних в Yii відбувається через Active Record, який правильно використовує підготовлені запити PDO всередині. При використанні підготовлених запитів неможливо маніпулювати запитом як це показано вище.

Проте, іноді потрібні сирі запити або генератор запитів[56]. В цьому випадку ви повинні використовувати безпечні способи передачі даних. Якщо дані використовуються для порівняння зі значенням стовпців краще використовувати підготовлені запити, які продемонстровані у лістингу 2.10, код побудовано на основі документації YII [39].

Лістинг 2.10 - Звертання до БД через генератор запитів

```
// query builder
$userIDs = (new Query())
    ->select('id')
    ->from('user')
    ->where('status=:status', [':status' => $status])
    ->all();
$userIDs = $connection
```

```
    ->createCommand('SELECT id FROM user where
status=:status')
    ->bindValues([':status' => $status])
    ->queryColumn();
```

XSS або крос-сайтінговий скриптинг стає можливий, коли НЕ екранований вихідний HTML потрапляє в браузер. Наприклад, якщо користувач повинен ввести своє ім'я, але замість Alexander він вводить <script> alert ('Hello!'); </ Script>, то всі сторінки, які його виводять без екранування, будуть виконувати JavaScript alert ('Hello!') ;, і в результаті буде виводитися вікно повідомлення в браузері. Залежно від сайту, замість невинних скриптів з висновком спливаючого hello, зловмисниками можуть бути відправлені скрипти, що викрадають особисті дані користувачів сайту, або виконують операції від їх імені.

В Yii уникнути XSS легко[57]. На місці виведення тексту необхідно вибрати один з двох варіантів:

1)вивести дані у вигляді звичайного тексту.

2)вивести дані в вигляді HTML.

Якщо потрібно вивести простий текст, то екранувати краще методом \yii\helpers\Html::encode(), якщо ж потрібно вивести HTML, то на такий випадок передбачений метод \yii\helpers\HtmlPurifier::process().

2.3.4 Аналіз фреймворку Zend

Zend Framework 3 є фреймворком з відкритим вихідним кодом для розробки web-додатків на PHP 5.3 +. Використовує тільки об'єктно - орієнтований код і всі нововведення PHP від версії 5.3 і вище,а саме: namespaces, late static binding, lambda functions and closures

Zend Framework 3 еволюціонував від Zend Framework 3 - популярного фреймворку з більш ніж 15 мільйонами сайтів використовують його.

Кожен компонент Zend Framework 3 є унікальним і розроблений з мінімальними залежностями від інших компонентів. ZF3 слід незалежного принципом створення додатків. Така слабо зв'язаної архітектура дозволяє розробникам використовувати тільки ті компоненти, які їм необходіми.Так ж використовується Pyrus і Composer для установки і відстеження залежностей як для всього проекту в цілому так і для кожного з компонентів. Для тестування коду використовуються PHPUnit і Travis CI.
Хоча кожен компонент Zend Framework 3 може бути використаний окремо, стандартний набір бібліотек робить його дуже потужним і розширюваним засобом розробки ВЕБ-додатків.

Крім того, він пропонує надійну і високопродуктивну реалізацію MVC, абстракцію бази даних, яка проста у використанні, форми, що реалізують HTML5 форми візуалізації, перевірки і фільтри, так що розробники можуть об'єднати всі ці можливості за допомогою одного простого і об'єктно-орієнтованого інтерфейсу. Інші компоненти, такі як Authentication і Acl, забезпечують аутентифікацію і авторизацію призначених для користувача облікових даних.

Також, з простором імен ZendService можливий легкий доступ до більшості найпопулярніших ВЕБ-сервісів. Незалежно від того, який проект ви хочете реалізувати, швидше за все знайдете все необхідне для швидкого і якісного створення в ядрі Zend Framework 2.

Для роботи з БД у Zend Framework передбачений цілий модуль Zend_Db, який містить у собі всі необхідні методи для вибірки чи зміни записів у БД.

Операції Select проводяться за допомогою окремого об’єкту Zend Db Sele. Клас містить у собі методи для кастомізації окремих частин запиту. Використовуючи методи PHP можна змінити стандартні частини запису і структури даних, результатом роботи класу є коректний SQL запит. Прикладом такого запиту є лістинг 2.11.

До переваг Zend_Db_Select можна віднести:

- ООП методи для фрагментованої побудови запитів SQL;
- Методи містять зарезервовані словники SQL слів і спецсимволів;
- Під час відправки запиту, він береться у лапки, та екранує спец символи, що суттєво можливість атак з типу SQL-ін'єкція.

Лістинг 2.11 - Приклад запиту до БД за допомогою Zend_Db_Select[36]

```
$minimumPrice = 100;
$maximumPrice = 500;

$select = $db->select()
             ->from('products',
                    array('product_id', 'product_name',
'price'))
             ->where('price < ?', $minimumPrice)
             ->orWhere('price > ?', $maximumPrice);
```

Використання такого метода дозволяє забезпечити високий ступінь захисту від SQL-ін’єкцій.

2.3.5 Аналіз фреймворку Bitrix Framework

На відміну від Zend Framework при розгортанні Bitrix Framework ми отримуємо не тільки набір класів, а й розвинений інтерфейс адміністрування [58].

У базовій поставці йде великий набір компонентів, і саме він забезпечує швидке розгортання і впровадження проектів.

За даними рейтингу 1С-Бітрікс займає п’яте місце серед популярних CMS в Україні. На її базі працює 9,48% від всіх україномовних сайтів. 1С-Бітрікс володіє великим числом модулів, опцій, які дозволяють істотно розширювати функціонал інтернет-магазину.

Великою перевагою Bitrix є те, що розробнику не доводиться працювати з прямими запитами до БД. Замість цього у даній CMS реалізована система Інформаційних блоків.

Інформаційні блоки - ключовий момент Bitrix Framework. Практично все, що робиться в системі в тій чи іншій мірі зав'язано на цей модуль, навіть якщо це і не відображається явно.

Інформаційні блоки представляють собою черговий рівень абстракції над звичайними таблицями СУБД, своєрідна "база даних в базі даних". Тому до них частково застосовні всі ті правила, яких дотримуються при проектуванні БД [59].

Інфоблоки - сутність, яка в фізичну структуру БД створює 4 таблиці, не змінюються при зміні структури даних: типи об'єктів, екземпляри об'єктів, властивості об'єктів і значення властивостей об'єктів.

Плюси такого підходу:

- зручний контроль над даними такої структури зі свого додатка,
- універсальність методів,
- загальна структура даних для будь-якого проекту,
- можливість багаторазово змінювати типи даних для полів без знищення самих даних.
- Мінуси такого підходу:
- підвищені вимоги до продуктивності,
- непрозорість при прямому доступі до даних.

Замість звичайних стовпців, до яких ми звертаємось під час роботи з MySQL, у бітрікс передбачено функціонал “Властивостей”[60]. Прикладом таких властивостей є рисунок 2.5.

Саме для роботи з інфоблоками використовується Bitrix Framework.

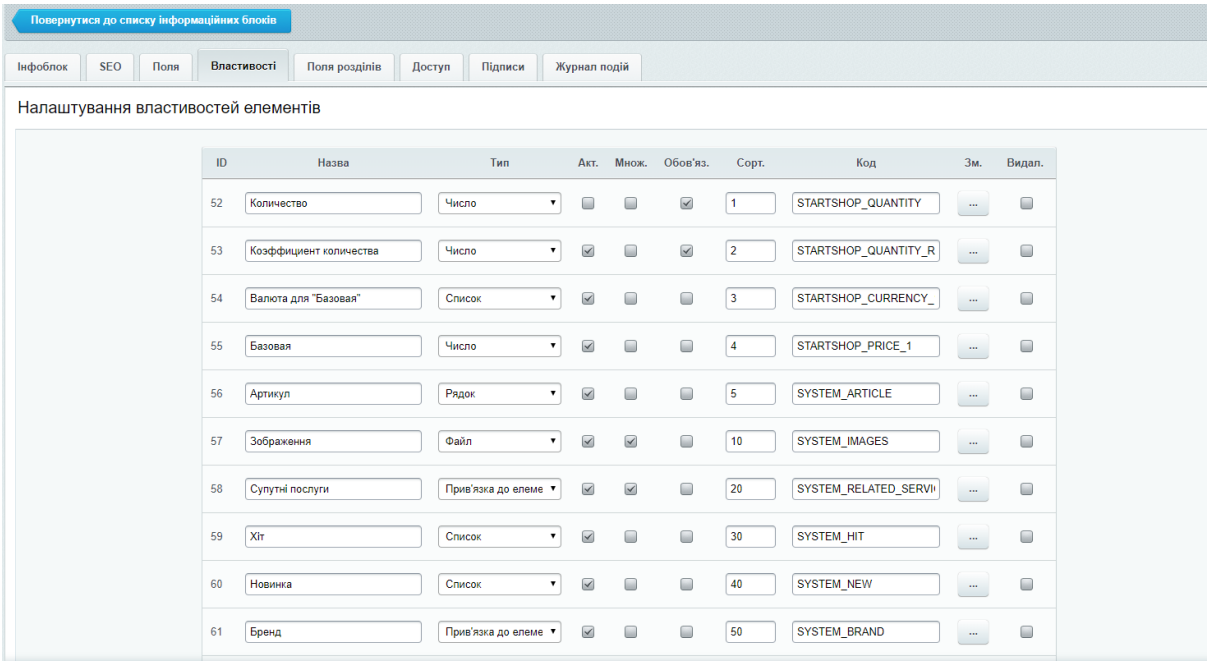


Рисунок 2.5 - Система властивостей

Демонстрацією вибірки записів з інформаційного блоку є лістинг 2.12, написаний нами для реалізації прикладу і порівняння. Після виконання даної вибірки на виході ми отримуємо асоціативний масив з елементами інформаційного блоку та його значеннями властивостей:ID, NAME, DATE_ACTIVE_FROM.

Лістинг 2.12 - Використання методу GetList

```
$res = CIBlockElement::GetList(
      Array(),
      Array("
            IBLOCK_ID"=>IntVal($IBLOCK_ID),
            "ACTIVE_DATE"=>"Y",
            "ACTIVE"=>"Y"
            ),
      false,
      Array("nPageSize"=>20),
      Array("ID", "NAME", "DATE_ACTIVE_FROM")
      );
```

```
while($ob = $res->GetNextElement()){
     $arFields[] = $ob->GetFields();
}
```

Перевагою використання методів є те, що в разі допущення помилки в синтаксисі, повідомлення про неї буде згенеровано не самою БД[61], а ще до неї, при виконанні php скрипта.

Для запису та зміни властивостей елементу інформаційного блоку використовується метод SetPropertyValuesEx, приклад використання наведено у лістингу 2.13.

Лістинг 2.13 - метод SetPropertyValuesEx класу CIBlockElement

```
$ELEMENT_ID = 18;
$PROPERTY_CODE = "PROP1";
$PROPERTY_VALUE = "Синий";
CIBlockElement::SetPropertyValuesEx($ELEMENT_ID, false,
array($PROPERTY_CODE => $PROPERTY_VALUE));
```

Безпека продукту 1С:Бітрікс - це цілий багатофункціональний комплекс технічних рішень які суттєво підвищують рівень захисту ERP-системи і її компонентыв. Проактивний захист - це багаторівнений захист від більшості атак на веб-додатки.

Модулю проактивний захист містить у собі великий функціонал:

- Підлкючення захисту DDoS;
- Панель рівнівзахисту;
- Модуль проактивного фільтру;
- Аудиту безпеки PHP коду;
- Веб-антивірус;
- Технологію генерації одноразових паролів для авторизації;
- Захист сесій під час авторизації;

- Контроль активності користувачів під час роботи з інфомраційними блоками;
- Захист авторизації без наявного SSL сертифікату;
- Журнал вторгнень;
- Контроль цілісності скриптів ядра;
- Монітор оновлень системи;

Проактивний фільтр забезпечує повноцінний потужний захист від усіх найбільш популряних видів атак на веб-додатки та веб-сайти. У звонішніх запитах до системи фільтр розпізнає практично усі загрози і при необхідності блокує запит.

При правильному налаштуванні у журнал вторгнень заносяться усі події, які система оприділяє, як небезпечні, незвичайні чи зловмисні. оперативна реєстрація даних подій дозволяє список відповідей в журналі одразу після їх генерації. Даний функціонал дозволяє миттєво реагувати не проведені атаки, та захистити ERP-систему від них.

В разі реєстрації проактивним фільтром небезпеки у журнал вноситься зпис і відмічається в одній з категорій:

1. спроба ін’єкції SQL запиту;
2. спроба атаки з використанням XSS;
3. спроба атаки з використанням завантаженого PHP-скрипта.

Використовуючи модуль контроль цілісності файлів ми зможемо у будь-який моментвідслідкувати зміни у ядрі, системних областях чи публічній частині продукту.

Захист адміністративного розділу, даний модуль дозволяє власниками програмного продукуту строго регламентувати та керувати трафіком, який можна вважати. Основний ефект від використання даного модулю - Будь-які XSS / CSS атаки на користувачів erp-системи стають неможливими.

Після проведення аналізу методів розробки захищеного програмного забезпечення, було описано переваги різних методів.

Написання проекту без використання зовнішніх бібліотек є самим повільним методом створення готового проекту, під час розробки всі обов`язки по передбаченню можливих вразливостей лежать на плечах розробника. Багато часу витрачається на реалізацію всього функціоналу.

Написання проекту з використанням сторонніх бібліотек суттєво знижує час реалізації різного функціоналу, такого, як робота з БД, з поштою чи зображеннями. При тому, валідація даних також залишається у повній мірі за розробником.

Реалізація проекту на фреймворку є найбільш швидким та безпечним методом реалізації. Для роботи з БД різні фреймворки використовують різні бібліотеки у зв’язці власними класами для валідація та перевірки даних.

В даному проекті найкращим фреймворком Ми вважаємо Bitrix, оскільки вже при початку роботі у розробників є величезний функціонал з повноцінною адміністраторською частиною. також при роботі відсутня необхідність у використанні прямих запитів до БД, замість цього розробник використовує API Bitrix Framework для вибірки елементів інформаційного блоку, що суттєво полегшує реалізацію певного функціоналу та збільшує захищеність системи.

3. ПРАКТИЧНЕ ЗАСТОСУВАННЯ BITRIX FRAMEWORK ДЛЯ РЕАЛІЗАЦІЇ ЗАХИЩЕНОЇ ERP-СИСТЕМИ

CMS 1-С Бітрікс, яка побудована на базі Bitrix Framework вже з коробки містить у собі величезний функціонал призначений для запобігання можливих вразливостей самої системи.

3.1 Встановлення та налаштування платформи для початку розробки

При установці проекту була використана віртуальна машина VM Bitrix. Вона побудована на базі Centos 7 і зразу адаптована для роботи CMS Bitrix.

Перший крок установки, який зображено на рисунку 3.1, на ньому першому необхідно вибрати реакцію самої CMS

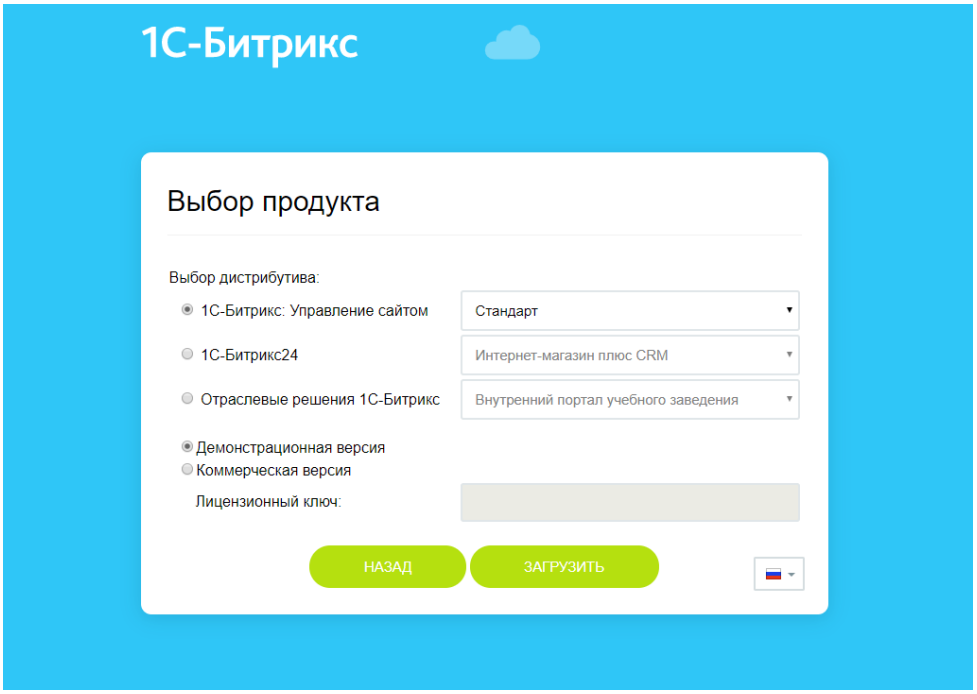


Рисунок 3.1 Перший крок установки

Після підтвердження погодження з ліцензією створюється користувач з надійним 16-ти значним паролем, згенерованим

спеціалізованою утилітою. Приклад даного етапу установки зображено на рисунку 3.2

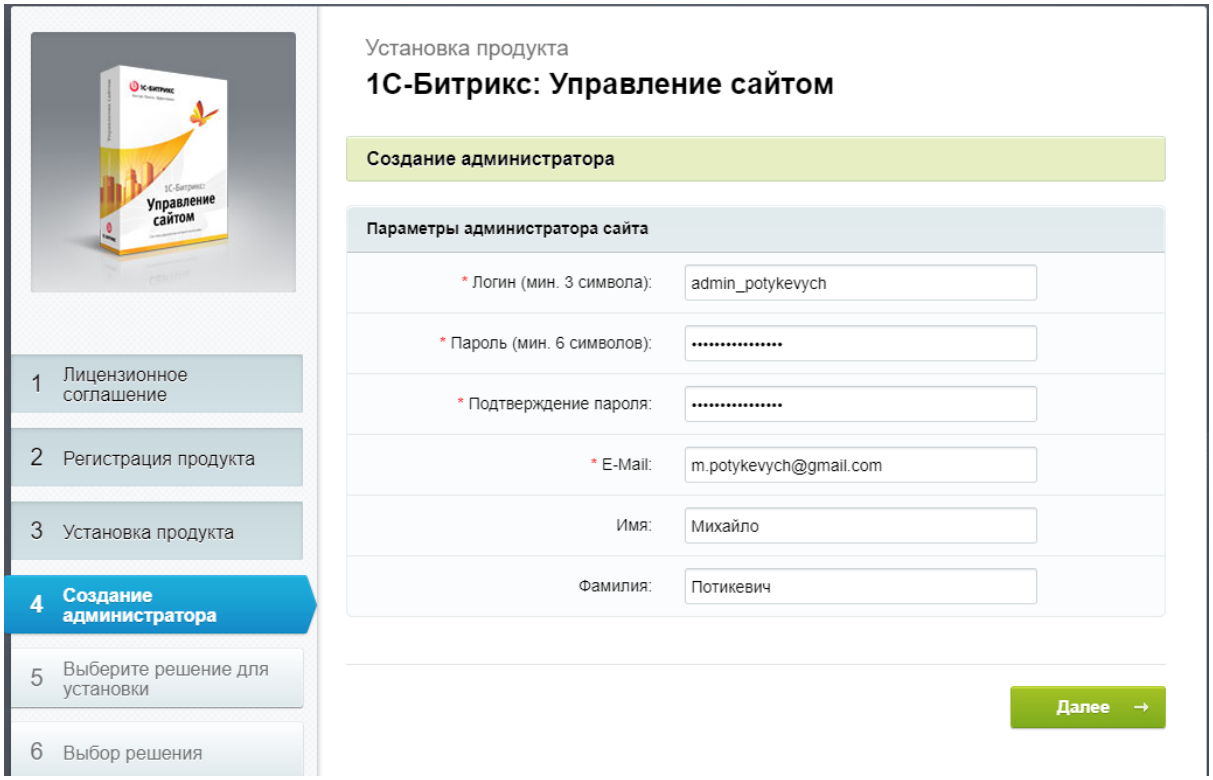


Рисунок 3.2 - Етап установки зі створенням користувача з правами адміністратора

Після завантаження самої CMS, створення БД та користувача необхідно закрити сайт від сторонніх користувачів. Самий надійний спосіб реалізувати дану функцію це через правила роботи веб-сервера. У файлі .htaccess було створено запис, який наведено у лістингу 3.1. В рядку AuthUserFile задається шлях до файла .htpassword, код даного файла наведено в лістингу 3.2.

Лістинг 3.1 - Авторизація через .htaccess

```
AuthType Basic
AuthName "Password Protected Area"
AuthUserFile /home/bitrix/www/.htpasswd
Require valid-user
```

Ключі даної директиви:

- AuthType – Тип аутентифікації, яка використовується.
- AuthName - Область дії аутентифікації.
- AuthUserFile - повний шлях від кореня проекту до файлу, який містить
- AuthGroupFile - шлях до файлу груп, якщо він існує.

Лістинг 3.2 - Логін на хешований пароль .htpasswd

```
mpoty_admin:$apr1$rFa75Gpq$woo7VvAiCjXbuBd.Zmcsm.
```

Результатом виконання даної директиви буде вспливаюче вікно, як буде вимагати ввести логін на пароль користувача для доступу до веб-ресурсу. Приклад вспливаючого вікна зображено на рисунку 3.3

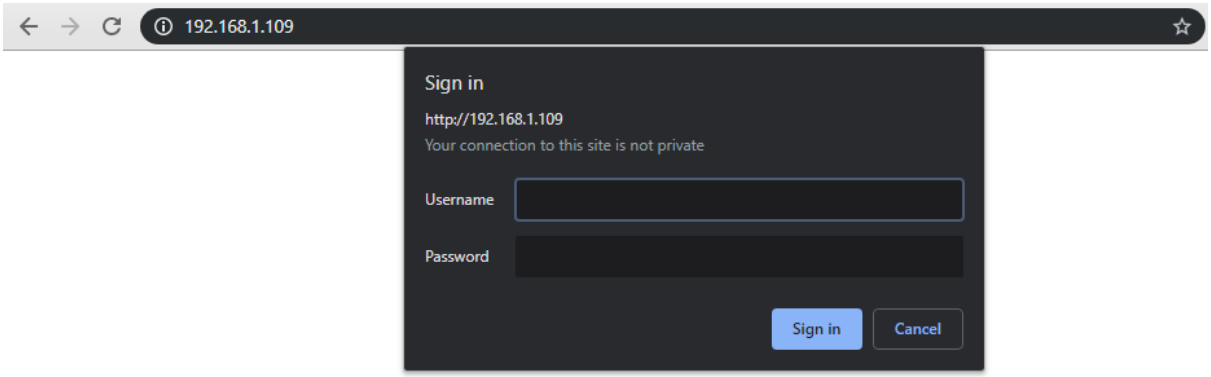


Рисунок 3.3 - Авторизація користувачів на стороні веб-сервера

Модуль проактивний захист містить у собі повний захист від відображення сайту у тезі iframe на інших ресурсах. Рисунок даного функціоналу зображено на рисунку 3.4

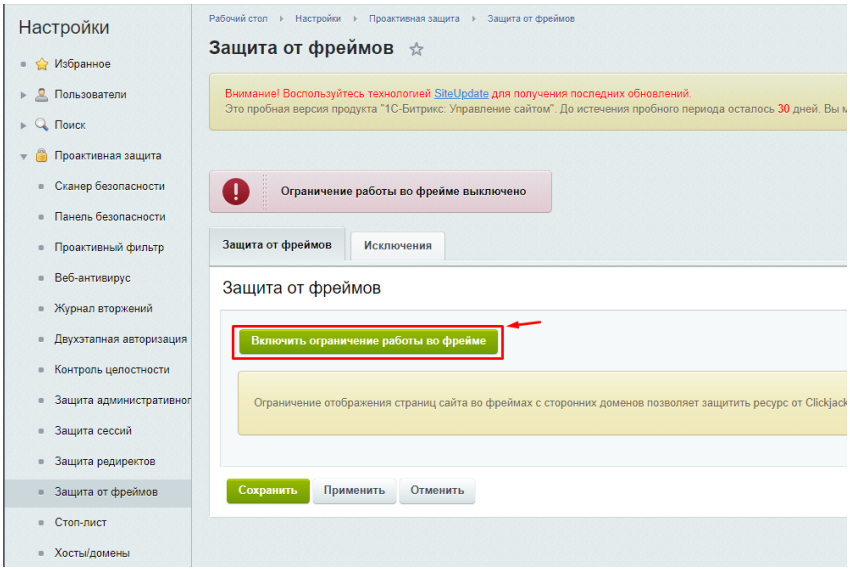


Рисунок 3.4 - Захист від фреймів

Тема X-Frame-Options з боку сервера може дозволяти або забороняти відображення сторінки всередині фрейму.

Це повинен бути саме HTTP-заголовок: браузер проігнорує його, якщо знайде в HTML-тегу <meta>. Тому при <meta http-equiv = "X-Frame-Options" ...> нічого не відбудеться.

Тема може мати 3 значення:

1. DENY - Ніколи не показувати сторінку всередині фрейму.
2. SAMEORIGIN -Дозволити відкриття сторінки всередині фрейму тільки в тому випадку, якщо батьківський документ має те ж джерело.
3. ALLOW-FROM domain - Дозволити відкриття сторінки всередині фрейму тільки в тому випадку, якщо батьківський документ знаходиться на зазначеному в заголовку домені.

Наступним кроком налаштуванн модулю проактивного захисту буде фільтрація доступу до ресурсу по ip-адресі. Даний спосіб захисту

унеможливить доступ стороннім користувачам до веб-проекту. РИсунок налаштування зображено на рисунку 3.5

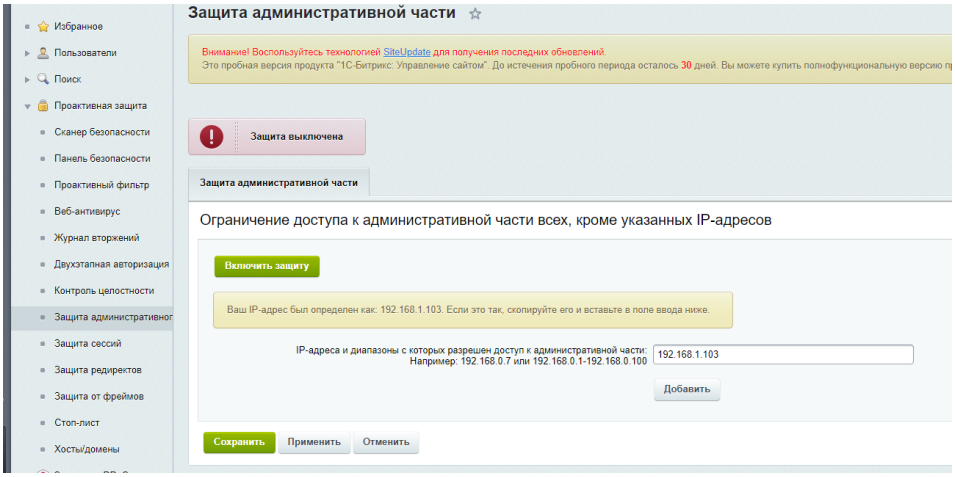


Рисунок 3.5 - Доступ тільки з дозволених IP адрес

Останнім етапом налаштування адміністраторської частини буде увімкнення двохетапної авторизації за допомогою мобільного додатку. Сторінка налаштування зображена на рисунку 3.6. Після налаштування цього кроку під час авторизації на моніторі користувача буде відображатись QR-код, після сканування якого користувач буде отримувати пароль для авторизації.

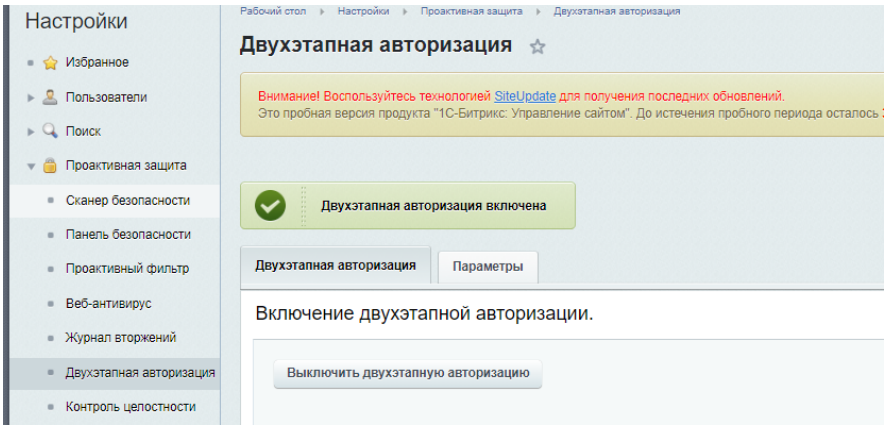


Рисунок 3.6 - Увімкнення двохетапної авторизації

Після налаштувань адміністраторської частини необхідно налаштування створення резервних копій ядра, бази даних, та користувацької частини. Етапи налаштування зображено на рисунках 3.7 та 3.8.

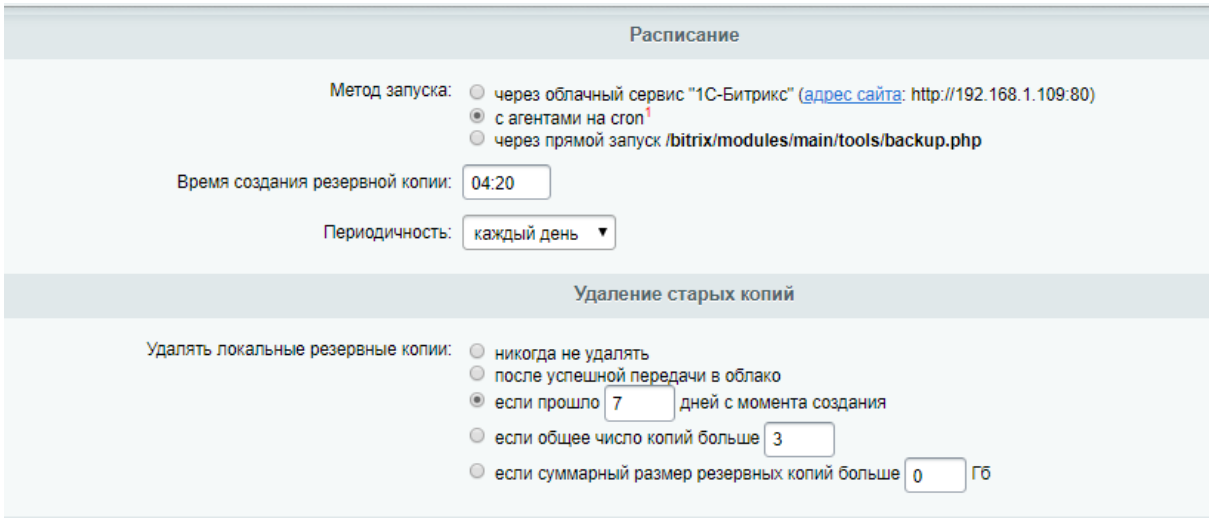


Рисунок 3.7 - Перший етап налаштування резервного копіювання

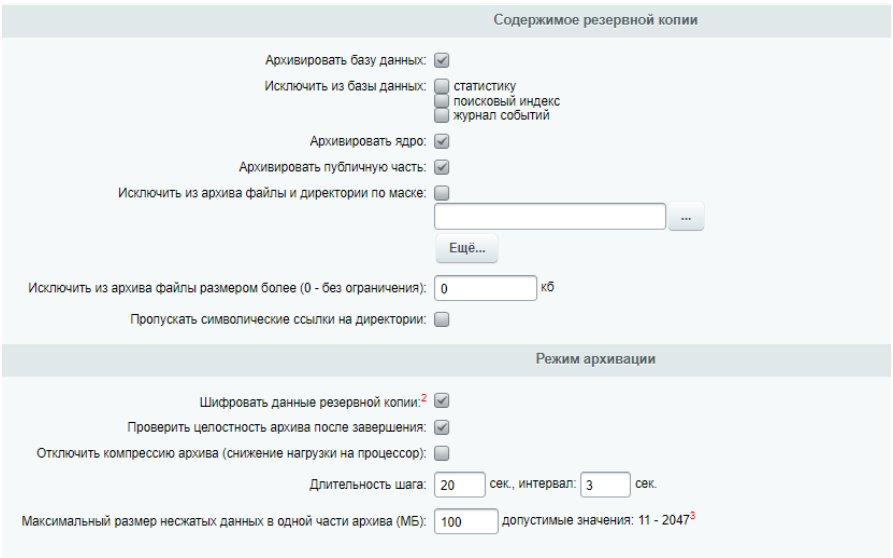


Рисунок 3.7 - Перший етап налаштування резервного копіювання

Існують загрози, коли зловмисник отримує доступ до файлів сайту через вразливості функціоналу. В такому випадку найкращим варіантом

захисту, до моменту виправлення вразливості, буде так зване цементування файлів сайту.

Під час цієї процедури через файловий менеджер FileZilla Ми змінили права доступу до файлів на 444. Це означає, що навіть при сценарії коли зловмисник може виконати фрагмент php коду, то зміни у файл внесені не будуть.

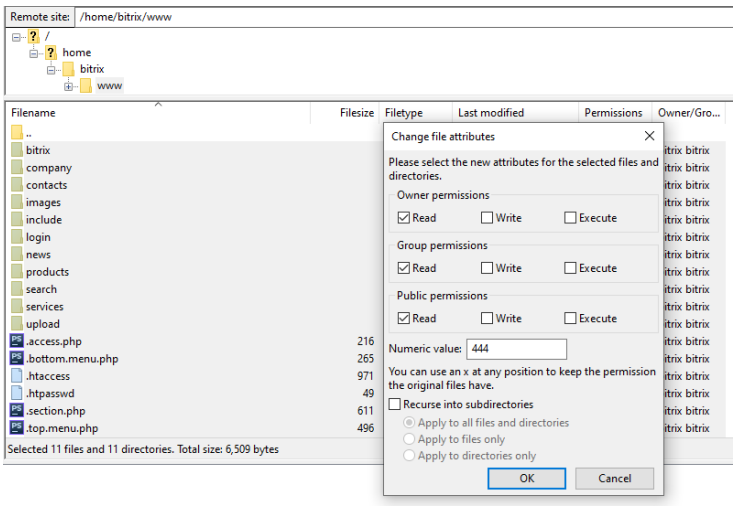


Рисунок 3.8 - Цементування файлів сайту

3.2 Створення інформаційних блоків та їх налаштування

Своєрідним аналогом звичної бази даних у бітріксі виступає модуль інформаційних блоків. Вся інформація про новини блогу, товари, статті на функціональні налаштування зберігаються у інформаційних блоках. Приклад інфоблоку товарів зображено на рисунку 3.10.

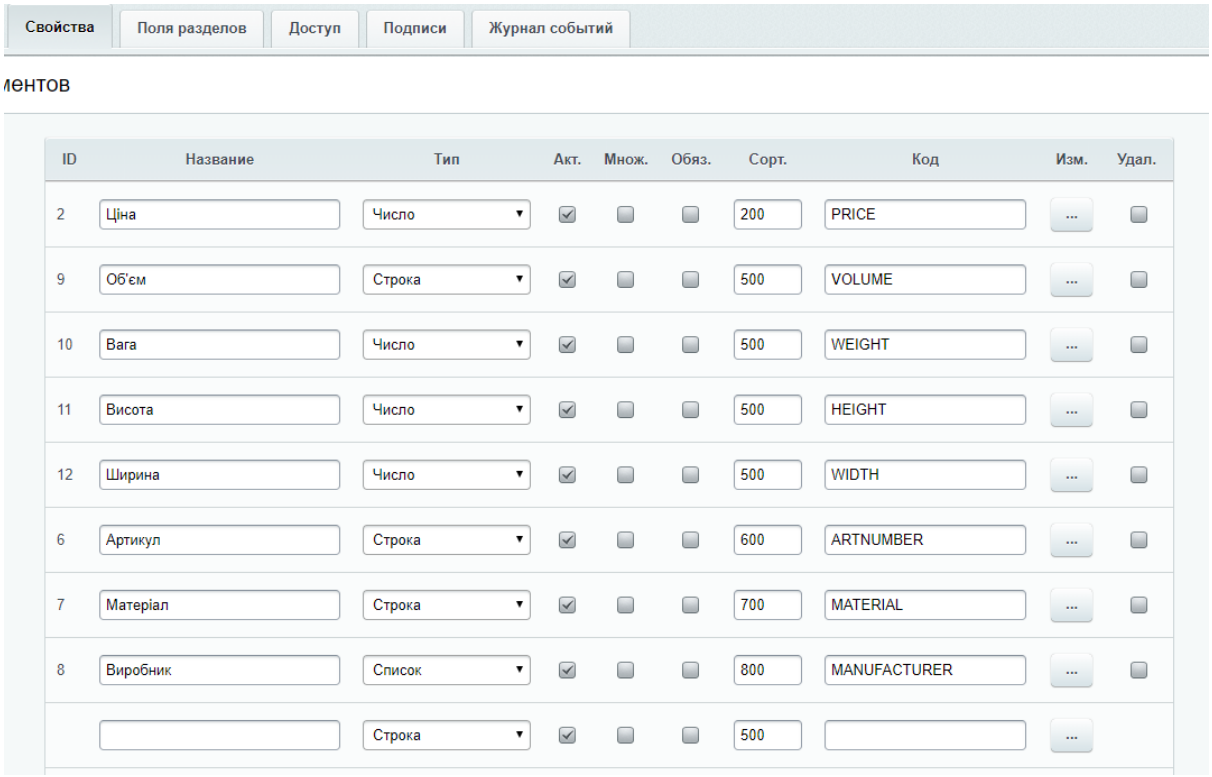


Рисунок 3.10 - Властивості елементів інформаційного блоку ”Продукція”

Для моніторингу змін у інформаційній системі необхідно увімкнути їх записування у журнал подій, приклад такого налаштування зображено на рисунку 3.11

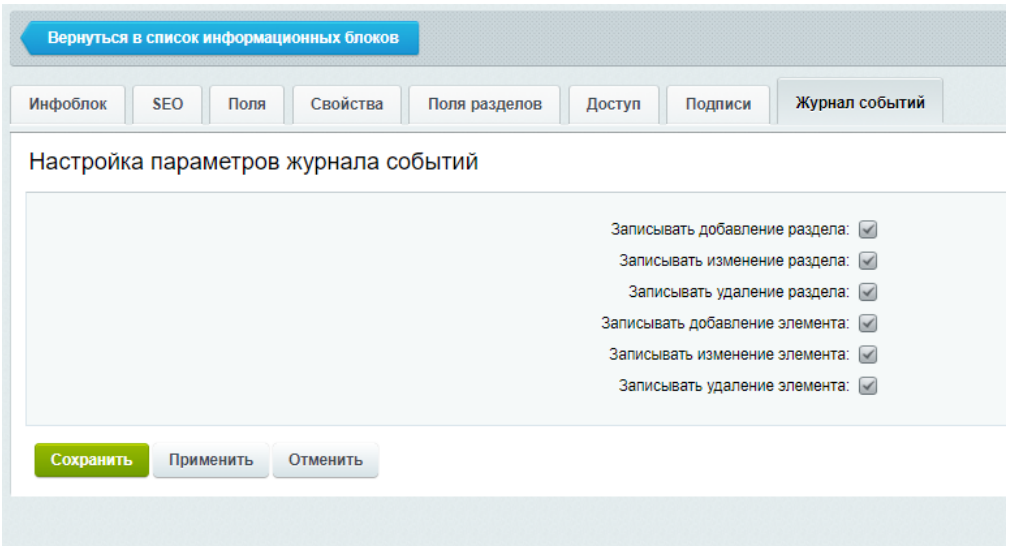


Рисунок 3.11 - Увімкнення логування змін елементів у інформаційному блоці

Після того, як інформаційний блок був створений, для його елементів були створені та налаштовані властивості необхідно зробити його доступним не зареєстрованим користувачам. Зробити це можна на вкладці “Доступ”, приклад налаштувань для публічного доступу до контенту інформаційного блоку представлено на рисунку 3.12. Для всіх користувачів необхідно встановити дозвіл на читання, а для адміністраторів та контент менеджерів дати право на редагування.

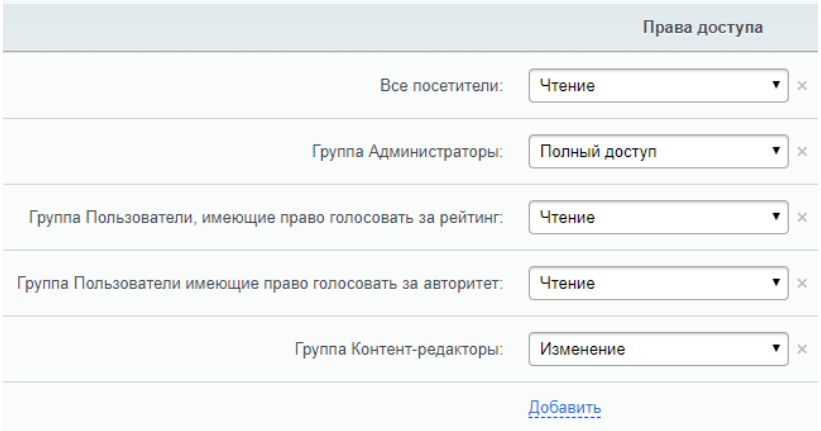


Рисунок 3.12 - Встановлення прав користувачам на доступ до елементів інформаційного блоку

3.3 Усунення вразливостей несанкціонованого доступу до ERP-системи

Якою б добре захищеною не була інформаційна система, у ній завжди залишаються певні вразливості, які можуть з’являтись під час додавання правок у готовий проект чи виправлення старих помилок.

Шевченко В.Л у своїй роботі [47] пояснює, що: “Однією з найбільш поширених і різноманітних інформаційних загроз, яка може завдати суттєвої шкоди інформаційній безпеці ERP-системи є несанкціонований доступ (НСД) до її інформаційних ресурсів. Досвід експлуатації ERP-системи показує, що незважаючи на тенденцію до підвищення рівня її

інформаційної захищеності, а також через постійне розширення її інформаційно-телекомунікаційних мереж, вона є досить уразлива з точки зору НСД. У зв'язку з цим захист ERP-системи від НСД до її інформаційних ресурсів розглядається як складова частина загальної проблеми забезпечення інформаційної безпеки ERP-системи".

Отже крім загальним небезпекам, які притаманні усім веб-ресурсам [48] додається проблема захисту від несанкціонованого доступу.

Підсумком роботи по дослідженню методів захисту інформації від несанкціонованого доступу [49] можна зробити наступний висновок: для запобігання атак і зниження ризиків від можливих загроз необхідно застосовувати спеціальні технології і засоби захисту web-додатків. Забезпечення захисту має здійснюватися як на етапі проектування і розробки самого web-додатки, шляхом створення безпечного коду web-додатки і планування раціонального складу системи захисту, так і в процесі його експлуатації з внесенням у разі необхідності своєчасних коригувань. оскільки навіть якщо web-додаток написано без помилок і уразливості в ньому немає, необхідна комплексний захист, що враховує наявність бази даних додатків, веб-сервера і інших елементів IT-платформи.

Виходячи з цього, захист пропонується будувати за наступними напрямами:

- контроль за безпекою коду web-додатки на протязі всього життєвого циклу розробки;
- забезпечення захисту інформації під час її передачі між комп'ютером клієнта і web-сервером;
- забезпечення захисту інформації, що зберігається на комп'ютері клієнта;
- використання спеціалізованих засобів захисту інформації на рівні web-сервера.

Це досягається за рахунок застосування наступних засобів і методів захисту:

- недопущення помилок в скриптах при розробці web-додатки;
- сканування коду web-додатки на наявність вразливостей і установка спеціальних водяних знаків
- використання систем багатофакторної аутентифікації користувачів (наприклад, парольний аутентифікація з секретним кодом, E-num, сертифікати, цифрові підписи, біометрична аутентифікація) [51];
- застосування антивірусного програмного забезпечення [50];
- застосування захищених каналів зв'язку і мережевих протоколів під час активного з'єднання клієнта.

Всі ці заходи потрібно виконувати в комплексі, оскільки захист по окремості не принесе бажаного ефекту. Таким чином, життєвий цикл розробки і супроводу захищеного web-додатки пропонується реалізовувати у вигляді циклічного процесу. Це рішення обумовлене тим, що загрози, як у відношенні певного напряму використання web-додатки, так і конкретних технологій змінюються дуже швидко, тому важливо регулярно відслідковувати статистику загроз інтернет-технологій, аналізувати виявлені інциденти безпеки і оцінювати ризики їх впливу на систему захисту web-додатки.

Оглянувши роботи [48], [49], [50], [51], ми виявили, що проблема несанкціонованого доступу залишається не вирішеною.

В свою чергу ми пропонуємо обмеження доступу до інформаційного ресурсу, у певні години для користувачів, які не є у “білому листі” сервісу.

Таким чином суттєво зменшити можливість доступу до ресурсу сторонніх осіб. Приклад алгоритму роботи даної системи захисту зображено на рисунку 3.13

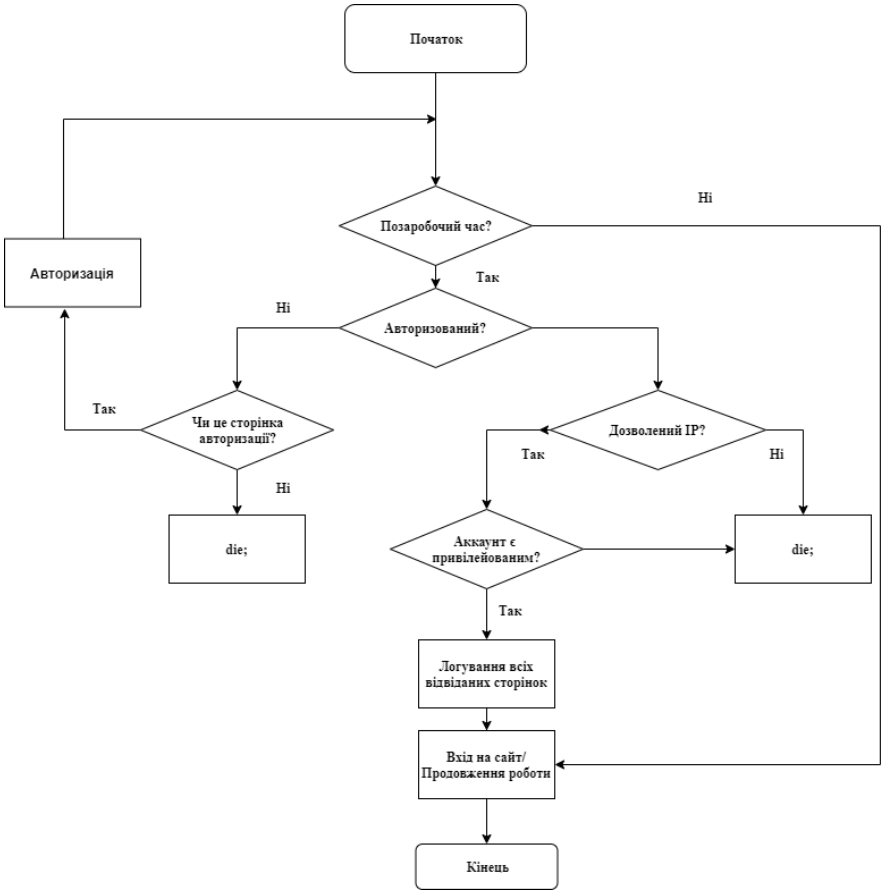


Рисунок 3.13 - Алгоритм захисту від несанкціонованого доступу

У якості “білого листа” IP адрес використовується інформаційний блок, приклад його властивостей наведено на рисунку 3.15.

Властивостями є IP адреса, Посада та електронна скринька користувача, яка використовується для авторизації у ERP-систему.

ID	Название	Тип	Акт.	Множ.	Обяз.	Сорт.	Код	Изм.	Удал.
13	IP	Строка	☑	☐	☐	500	IP	...	☐
14	Посада	Список	☑	☐	☐	500	POSITION	...	☐
16	E-mail	Строка	☑	☐	☐	500	EMAIL	...	☐
		Строка	☑	☐	☐	500		...	
		Строка	☑	☐	☐	500		...	
		Строка	☑	☐	☐	500		...	

Рисунок 3.14 - Створення інформаційного для “білого листа”.

Після створення і налаштування інформаційного блоку, його необхідно наповнити: вказати ім’я, дозволену IP адресу, посаду та електронну скриньку для авторизації.

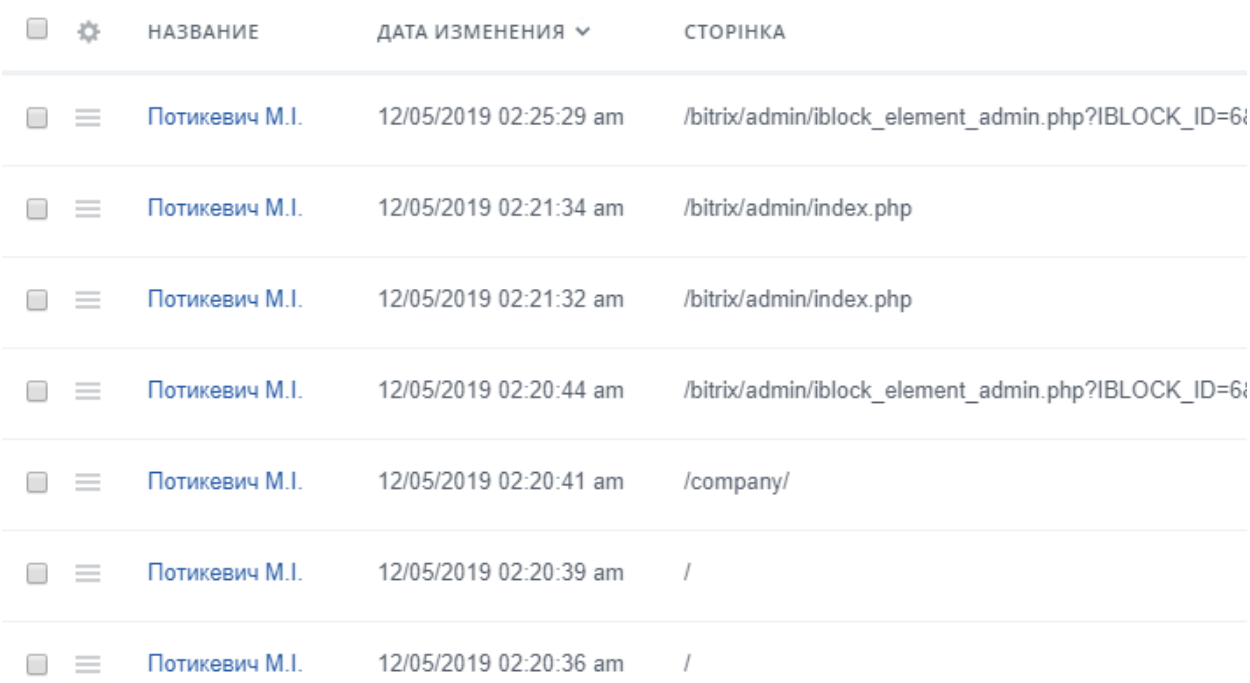
Оскільки доступною авторизація буде тільки для адміністраторів та директорів, контент-менеджерам доступ у позаробочий час буде забороненим. Приклад наповненого інформаційного блоку зображено на рисунку 3.15

НАЗВАНИЕ	E-MAIL	IP	ПОСАДА
Харченко С. І.	s.kharchenko@gmail.com	185.24.35.12	Direcotr
Данилюк І.О.	danylyuk@gmail.com	91.55.239.57	Content mnager
Потикевич М.І.	m.potykevych@gmail.com	192.168.1.103	Administator

Рисунок 3.15 - Приклад додавання дозволених IP адрес

Для ERP-системи критично важливим є точне керування ресурсами та уникання навіть найменших помилок, оскільки - це може призвести до значних фінансових втрат. Для цього у 1С:Бітрікс передбачено журнал подій, проте в ньому записуються лише зміни елементів чи розділів, але не записується історія відвідувань сторінок користувчаем.

Для контролю дій користувача на сайті у позаробочий час реалізовано функціонал логування всіх переходів та відвіданих сторінок. Прикладом такого логування є рисунок 3.16, на якому відображено історію переходів користувача.



НАЗВАНИЕ	ДАТА ИЗМЕНЕНИЯ	СТОРІНКА
Потикевич М.І.	12/05/2019 02:25:29 am	/bitrix/admin/iblock_element_admin.php?IBLOCK_ID=6
Потикевич М.І.	12/05/2019 02:21:34 am	/bitrix/admin/index.php
Потикевич М.І.	12/05/2019 02:21:32 am	/bitrix/admin/index.php
Потикевич М.І.	12/05/2019 02:20:44 am	/bitrix/admin/iblock_element_admin.php?IBLOCK_ID=6
Потикевич М.І.	12/05/2019 02:20:41 am	/company/
Потикевич М.І.	12/05/2019 02:20:39 am	/
Потикевич М.І.	12/05/2019 02:20:36 am	/

Рисунок 3.16 – Історія відвіданих сторінок коритувачем

В лістингу 3.3 наведено приклад реалізації скрипта фільтрації віхдного трафіку у неробочі години.

Лістинг 3.3 – Фільтр вхідного трафіку у неробочі години

```
<?
//Задаємо часові рамки 00:00 - 08:00
$paymentDate = strtotime(date("H:i:s"));
$contractDateBegin = strtotime("00:00:00");
$contractDateEnd = strtotime("08:00:00");
if($paymentDate > $contractDateBegin && $paymentDate < $contractDateEnd) {
 CModule::IncludeModule('iblock');

 //$ip- ip адреса користувача
 if (!empty($_SERVER['HTTP_CLIENT_IP'])) {
   $ip = $_SERVER['HTTP_CLIENT_IP'];
 } elseif (!empty($_SERVER['HTTP_X_FORWARDED_FOR'])) {
   $ip = $_SERVER['HTTP_X_FORWARDED_FOR'];
 } else {
   $ip = $_SERVER['REMOTE_ADDR'];
 }
```

```
      $user_list = [];
      $ip_list = [];
      $arSelect = Array("ID", "NAME", "DATE_ACTIVE_FROM",
      'PROPERTY_IP', 'PROPERTY_POSITION', 'PROPERTY_EMAIL');
      $arFilter = Array("IBLOCK_ID"=>IntVal(5),
      "ACTIVE_DATE"=>"Y", "ACTIVE"=>"Y");
      //Отримуэмо список довзолених IP
      $res = CIBlockElement::GetList(Array(),
      $arFilter, false, Array("nPageSize"=>50), $arSelect);
      while($ob = $res->GetNextElement()){
        $arField = $ob->GetFields();
        $user_list[] = $arField;
        $ip_list[] = array(
            'ip'    => $arField['PROPERTY_IP_VALUE'],
            'email' => $arField['PROPERTY_EMAIL']
            );
      }
      global $USER;
       //Перевірка чи авторизований
       if ($USER->IsAuthorized()){
         //Чи є IP адреса відвідувача у списку дозволених
         if(in_array($ip, $ip_list)){
          foreach($user_list as $key=>$user){
           if($ip == $user['PROPERTY_IP_VALUE']){
           //Чи є поточний коритсувач превілейованим аккаунтом
           if($user['PROPERTY_POSITION_VALUE']                 ===
'Administator'    ||    $user['PROPERTY_POSITION_VALUE']    ===
'Director'){
              //Додаємо запис про відвідувані сторінки
              $PROP[15] = $_SERVER['REQUEST_URI'];
              $el = new CIBlockElement;
              $arLoadProductArray = Array(
               "IBLOCK_ID"  => 6,
               "NAME"     => $user['NAME'],
               "PROPERTY_VALUES"=> $PROP,
               "ACTIVE"   => "Y",     // активен
               "ACTIVE_FROM"  => date()
              );
              $PRODUCT_ID = $el->Add($arLoadProductArray);
             }else{
              die;
             }
           }
          }
         }else{
          die;
         }
       }else{
         if($_SERVER['REQUEST_URI'] !=
'/bitrix/admin/index.php#authorize'){
```

```
        die();
      }
    }

  }
```

Отже результатом виконання нашого скрипта буде додаткова перевірка користувачів у післяробочий час, який заданий з 00:00 до 08:00. По IP адресі, з якої відбувається спроба авторизації, перевірка на привілєйованість користувацького облікового запису та врахування посади працівника у компанії. Дані перевірки суттєво зменшують можливість несанкціонованого доступу до ERP-системи.

4. СПЕЦІАЛЬНА ЧАСТИНА. ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ ДЛЯ РОЗРОБКИ ERP-СИСТЕМИ

4.1 IDE PhpStorm

У якості середовища розробки нами було обрано PhpStorm від компанії JetBrains.

Головною перевагою редактора редактора є висока швидкість та оптимізація.

Також до корисного функціоналу можна віднести:

1. Пошук класів Ctrl + N і файлів Ctrl + Shift + N . Дуже корисно і швидко, тому що файли як правило розташовані різних гілках дерева каталогів;

2. Скролл при пошуку або при навігації по своїх закладках рядків через Ctrl + цифра. Дуже допомагає не загубитися і інтуїтивно розуміти в який бік файлу ми рухаємось.

3. Рефакторинг імен файлів, класів і методів у всьому проекті і / або пошук їх використання.

4. Підключення зовнішніх папок.

Для випадків, коли потреба у повному завантаженні проекту відсутня у PhpStorm передбачено функціонал для віддаленого редагування файлів. На рисунку 4.1 зображено приклад модулю Deployment. Використання цього плагіна дозволяєш швидко підключить по FTP/SFTP до віддаленого файловго сервера, та внести зміни, не викачуючи при цьому файли на валсний ПК.

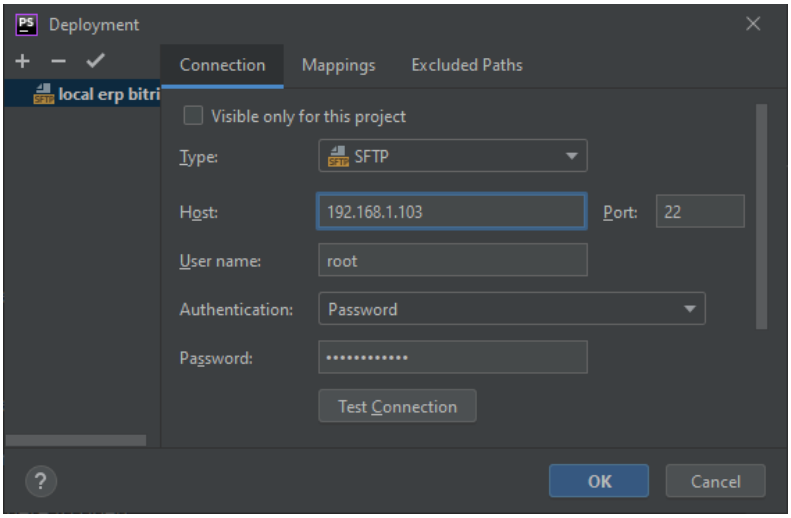


Рисунок 4.1 - Deployment у редакторі PhpStorm

4.2 Файловий менеджер FileZilla

Для роботи з файлами використовуємо FileZilla. На сьогоднішній день найкращим FTP-клієнтом ми вважаэмо FileZilla - безкоштовний багатоплатформовий менеджер. Крім безкоштовності, додаток має ряд якісних переваг, які дозволили їй зайняти сьоме місце в рейтингу найбільш скачуваних програм. FileZilla відрізняється легким і зручним призначеним для користувача інтерфейсом, переведеним більш ніж на 40 мов, високою швидкістю передачі даних, а також винятковою простотою використання.

Крім протоколів FTP, FTPS і SFTP програма підтримує роботу з HTTP / 1.1, SOCKS5, проксі FTP і новим ip протоколом IPv6.

Робота з багатьма джерелами не стає проблемою, оскільки FileZilla містить у собі “Менеджер сайтів” у якому зберігаються усі дані для доступу до віддаленого сервера.

З додаткових можливостей клієнта можна відзначити фільтрацію імен файлів, налаштування обмеження швидкості, створення закладок, підтримку докачки в разі обриву зв'язку, шифровку за допомогою Kerberos, захист з'єднання, а також віддалений пошук і редагування на стороні сервера.

Зручність даного файлового менеджера полягає у можливості збереження доступів до всіх наявних проектів, дякуючи цьому повністю відпадає необхідність вручну вводити дані для входу на FTP/SFTP сервер.

Для швидкої та оперативної роботи передбачено вікно з локальними файлами, та вікно з файлами розміщеними на віддаленому сервері. Приклад робочого процесу з FileZilla зображено на рисунку 4.2.

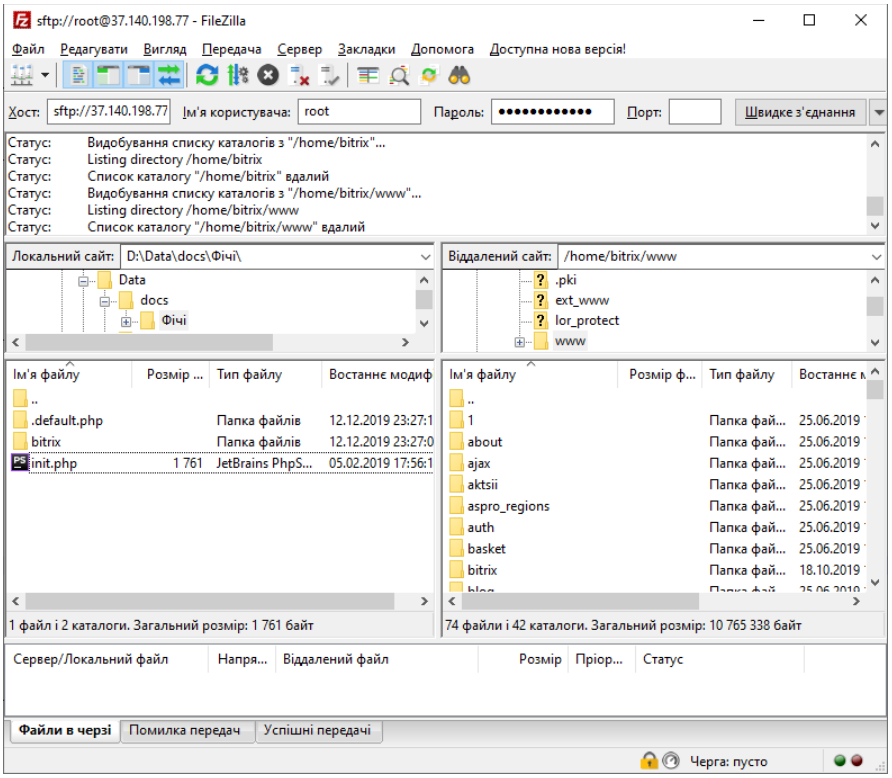


Рисунок 4.2 - Функціонал FileZilla

4.3 Віртуальна машина VMware Workstation Player

Компанія 1С для своїх клієнтів розробила програмне забезпечення, яке можна встановити на виділений віртуальний сервер чи локальну віртуальну машину.

В нашому випадку ми використали VMware Workstation Player. Програмне забезпечення VMware Workstation Player - це оптимізований

додаток для віртуалізації, призначений для одночасного запуску декількох операційних систем на одному і тому ж комп'ютері.

Зручний інтерфейс VMware Workstation Player, а також широкий вибір підтримуваних операційних систем та гнучкі можливості перенесення існуючих віртуальних машин істотно спрощують задачу надання корпоративним користувачам повноцінних робочих проектів для тестування.

Одразу після встановлення Bitrix VM на віртуальну машину ми отримуємо великий функціонал, що допоможе встановити та провести усі необхідні налаштування для роботи з 1С:Бітрікс. Приклад всього функціоналу зображено на рисунку 4.3.

Рисунок 4.3 - Функціональні можливості BitrixVM

Отже використовуючи PhpStorm для написання коду ми суттєво спрощуємо написання скриптів, оскільки дана IDE дозволяє підключати велику кількість сніпетів, що дозволяє заново використовувати вже написаний раніше код з мінімальними витратами на це часу.

FileZIlla дозволяє працювати з віддаленими серверами через протокол SFTP, що унеможливлює перехоплення користувацьких даних.

Використання Bitrix VM суттєво економить час під час налаштування VPS, для установки 1С:Бітрікс.

5 ОБҐРУНТУВАННЯ ЕКОНОМІЧНОЇ ЕФЕКТИВНОСТІ

Дана дипломна робота розглядає питання аналізу методів розробки захищених ERP-систем. У сучасних умовах ефективне управління являє собою цінний ресурс організації (поряд з фінансовими, матеріальними, людськими та іншими ресурсами). Отже, підвищення ефективності управлінської діяльності стає одним з напрямків вдосконалення діяльності підприємства в цілому. Найбільш очевидним способом підвищення ефективності протікання трудового процесу є його автоматизація.

Автоматизація бізнес процесів допомагає створювати, описувати та управляти виконуваними бізнес-процесами в прикладних програмах.

В даний час вдосконалення корпоративного управління стає ключовим стратегічним завданням розвитку будь-якого підприємства.

3.1 Розрахунок фінансових витрат

Капітальні інвестиції:

- Вартість розробки та налаштування;
- вартість створення основного програмного забезпечення (ПЗ);
- витрати на первісні закупівлі програмного забезпечення;
- витрати на навчання технічних фахівців і обслуговуючого персоналу.

Для початку розрахуємо час, який необхідний для розробки ERP-системи:

$$t = t_{тз} + t_{б} + t_{пр} + t_{опр} + t_{д}\text{ , годин,} \qquad (3.1)$$

де $t_{тз}$ – тривалість складання технічного завдання;

$t_{б}$ – тривалість розробки блок-схеми алгоритму;

tпр – тривалість програмування за готовою блок-схемою;

topr – тривалість опрацювання програми на ПК;

topr – тривалість встановлення та налаштування 1с:Бітрікс

$t_д$ – тривалість підготовки технічної документації на ПЗ.

Умовна кількість оперантів у програмі:

$Q = q \cdot c\ (1 + p)$, штук, (3.2)

Де:

q – очікувана кількість оперантів - 12;

с – коефіцієнт складності програми -1.5;

р – коефіцієнт корекції програми в процесі її опрацювання – 0.05.

$Q = 12 \cdot 0.8(1+0.05) = 10.7$.

Оцінка тривалості складання технічного завдання на розробку ПЗ

$t_{тз}$ – 4 год.

Тривалість вивчення технічного завдання: $t_в = 2.5$ год.

Тривалість розробки блок-схеми алгоритму:

$$t_б = \frac{Q}{(\ 20...25\ ) \cdot k} = \frac{10.65}{20 \cdot 0.8} = 0.67 \text{ , годин.}$$

Тривалість розробки програмного забезпечення :

$t_{опр}$ = 85 годин

Тривалість підготовки технічної документації на ПЗ:

$$t_д = \frac{Q}{(15...20) \cdot k} + \frac{Q}{(15...20)} \cdot 0{,}75 = \frac{25.2}{15 \cdot 0.8} + \frac{25.2}{15} \cdot 0.75 = 3.36 \text{ годин.} \quad (3.7)$$

t = 4+2.5+0.67+85+1.2 = 93.4 годин.

Заробітна плата виконавця враховує основну і додаткову заробітну плату,

а також відрахування на соціальне потреби (пенсійне страхування, страхування на випадок безробіття, соціальне страхування тощо) и визначається за формулою:

$$К_{пз} = З_{зп} + З_{мч}. \text{ грн} \quad (3.6)$$

$$З_{зп} = t \cdot З_{пр} = 93.4 \cdot 90 = 8406, \text{ грн}, \quad (3.7)$$

де t – загальна тривалість створення ПЗ, годин;

$З_{пр}$ – средньогодинна заробітна плата програміста з нарахуваннями,

грн/годину.

$$З_{пр} \frac{З_м}{168} = \frac{15120}{168} = 90 \text{ грн/год} \quad (3.8)$$

де

topp – трудомісткість налагодження програми на ПК, годин;

tд – трудомісткість підготовки документації на ПК, годин;

Смч – вартість 1 години машинного часу ПК, грн./година.

Вартість 1 години машинного часу ПК визначається за формулою:

$$P * C_e + \frac{Ф_{вал} * Н_а}{F_p} + \frac{К_{дпз} * Н_{апз}}{F_p} = 0.3 * 1.68 + \frac{2700 * 0,1}{1920} = 0,98 \quad (3.10)$$

де P – встановлена потужність ПК, 0.3 кВт;

$Се$ – тариф на електричну енергію, 1.68 грн/кВт·година;

$На$ – річна норма амортизації на ПК, 0.1 частки одиниці;

$Н_{апз}$ – річна норма амортизації на ліцензійне програмне забезпечення, частки одиниці;

$К_{лпз}$ – вартість ліцензійного програмного забезпечення, грн. 4500;

$F_р$ – річний фонд робочого часу (за 40-годинного робочого тижня *Fp* = 1920

год).

Таким чином, капітальні (фіксовані) витрати на проектування та впровадження

проектного варіанта системи інформаційної безпеки складають:

Витрати на навчання технічних фахівців і обслуговуючого персоналу, це є підготовчі курси з адміністрування та обслуговування системи виявлення

вторгнень що складають 1200 грн;

$К_{навч}$ – витрати на навчання технічних фахівців і обслуговуючого персоналу, тис. Грн = 1200 грн;

$К_{ср}$ – Вартість оренди vps 5712 грн. на перший рік

$$К = К_{пз} + К_{навч} + К_{ср} = 8406+1200+4500++5712 = 19\ 818 \text{ грн.} \quad (3.11)$$

3.2 Експлуатаційні витрати:

3.3

Де витрати на навчання адміністративного персоналу й кінцевих користувачів(Сн). визначаються за даними організації з проведення тренінгів персоналу, курсів підвищення кваліфікації – 1 тис. грн.

$$Ск = Сн + Са + Сз + Сєв + Се + Сел + Стос \quad (3.12)$$

Річний фонд амортизаційних відрахувань (Са) визначається у відсотках від суми капітальних інвестицій за видами основних фондів і нематеріальних активів (ПЗ) – 20% або 922 грн.

Річний фонд заробітної плати інженерно-технічного персоналу, що обслуговує систему інформаційної безпеки (Сз), складає:

$$Сз = Зосн + Здод = 3200\ 12 + 3200\ 0.22\ 12 = 46\ 848 \text{ грн.} \quad (3.13)$$

де

$З_{осн}$, $З_{дод}$ – основна мінімальна заробітна плата на 01.12.2017, грн на рік.
Єдиний соціальний внесок – 0.22, частки одиниці;

Вартість електроенергії, що споживається апаратурою системою

інформаційної безпеки протягом року (Се), визначається за формулою:

$$Сел = Р \cdot Fp \cdot Це = 0.4 \cdot 365 \cdot 24 \cdot 1.68 = 3\ 433.92 \text{ грн,} \quad (3.14)$$

$Ц_е$ – тариф на електроенергію, грн/кВт·годин

Витрати на технічне й організаційне адміністрування та сервіс системи виявлення вторгнень визначаються у відсотках від вартості капітальних витрат 2%. А саме:

$$Стос = К \cdot 0.2 = 19\ 818*0{,}2 = 3\ 964 \text{ грн} \quad (3.15)$$

3.3 Оцінка можливого збитку від атаки (злому) на ERP-систему

Кінцевим результатом впровадження й проведення заходів щодо забезпечення інформаційної безпеки є величина відвернених втрат, що розраховується, виходячи з імовірності виникнення інциденту інформаційної безпеки й можливих економічних втрат від нього. По суті, ця величина відображає ту частину прибутку, що могла бути втрачена.

Загалом можливо виділити такі види збитку, що можуть вплинути на erp-системи:

- порушення конфіденційності ресурсів;
- порушення доступності;
- порушення цілісності ресурсів;

Вихідні дані:

tп =28 годин – час простою вузла або сегмента корпоративної мережі внаслідок атаки, годин;

tв=8 годин – час відновлення після атаки персоналом, що обслуговує erp-систему, годин;

tви=3 годин – час повторного введення загубленої інформації співробітниками атакованого вузла або сегмента корпоративної мережі, годин;

Зо=4173 грн – місячна заробітна плата обслуговуючого персоналу, грн на місяць;

Чо=1 – чисельність обслуговуючого персоналу;

О = 350 000 грн – обсяг чистого прибутку;

N = 18 – середнє число можливих атак на рік.

$$U = Пп + Пв + V, \text{ грн.} \quad (3.16)$$

де Пп – оплачувані втрати робочого часу та простої співробітників атакованого вузла або сегмента корпоративної мережі, грн;

Пв – вартість відновлення працездатності вузла або сегмента корпоративної

мережі (переустановлення системи, зміна конфігурації та ін.), грн;

V – втрати від зниження обсягу продажів за час простою атакованого вузла або сегмента корпоративної мережі, грн.

Пзч – вартість заміни устаткування або запасних частин, грн.

Витрати на повторне введення інформації Пви розраховуються виходячи з розміру заробітної плати 4000 грн 3 співробітниа атакованого

вузла або сегмента корпоративної мережі Зс, які зайняті повторним введенням втраченої інформації, з урахуванням необхідного для цього часу $t_{ви}$=3:

$$Пв = \frac{4173}{2160} * 8 = 624 \text{ грн} \quad (3.17)$$

Витрати на відновлення вузла або сегмента корпоративної мережі Ппв визначаються часом відновлення після атаки tв = 15 і розміром середньогодинної заробітної плати обслуговуючого персоналу (адміністраторів):

Втрати від зниження очікуваного обсягу продаж в 350 000 грн за 90 годин простою ERP-системи:

$$V = \frac{O}{F} * (t_{п} + t_{в} + t_{ви}) = \frac{350000}{8760} * (60 + 8 + 3) = 5893 \quad (3.18)$$

де Fг – річний фонд часу роботи організації (прийом заказів інтернет-магазином) становить близько 8760 ч.

Таким чином, загальний збиток від атаки erp-систему складе:

U = Пп + Пв + V= 4173 + 624 + 5893 = 10690 грн. (3.15)

$$B = \sum \sum U * N * I = 10690 \cdot 24 \cdot 1 = 256\,560 \text{ грн.} \quad (3.19)$$

3.4 Загальний ефект від впровадження системи інформаційної безпеки

Загальний ефект від впровадження системи інформаційної безпеки визначається з урахуванням ризиків порушення інформаційної безпеки і становить:

$$E = B \cdot R - C = 256\,560 * 0.7 - 46\,848 = 132\,000 \quad (3.20)$$

де B – загальний збиток від атаки на вузол або сегмент корпоративної мережі, грн;

R – очікувана імовірність атаки на вузол або сегмент корпоративної мережі, частки одиниці;

С – щорічні витрати на експлуатацію системи інформаційної безпеки, тис. грн.

Отже використовуючи захищену ERP-систему на базі 1С:Бітрікс компанія в рік економить 132 тис. грн., що є хорошим результатом. Розробка захищеної ERP-системи є доцільною.

6 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА У НАДЗВИЧАЙНИХ СИТУАЦІЯХ

6.1 Охорона праці

Аналізі методів розробки захищених ERP -систем проводився з використанням інформаційних ресурсів комп'ютерних систем. Дана робота цілком пов'язана з дисплеями моніторів. Важливими є вимоги до освітлення приміщень,оскільки відомо, що тривала робота за комп'ютером при недостатньому рівні освітленості може призвести до значного перенапруження зору.

Природне освітлення забезпечено коефіцієнтом природної освітленості і був не нижче, ніж 1,5%. Для регулювання рівня освітлення природним світлом застосовувати ролети. Робоче місце, було обладнане ПК і розташоване так, що попадання в очі прямого сонячного світла унеможливлене.

Приміщення було обладнано штучним освітленням і системою загального рівномірного освітлення.

Відповідно до ДСН 3.3.6.042-99, щодня перед початком роботи необхідно очищати монітор від пилу та інших забруднень. Після закінчення роботи персональний комп'ютер і периферійні пристрої повинні бути відключені від електричної мережі. У разі виникнення аварійної ситуації необхідно негайно відключити персональний комп'ютер і периферійні пристрої від електричної мережі.

Персональні комп'ютери, периферійні пристрої підключались до електромережі тільки з допомогою справних штепсельних з'єднань і електророзеток заводського виготовлення.

Приміщення для роботи з персональними комп'ютерами обладнані системами опалення, кондиціонування повітря. У приміщеннях на робочих

місцях мають забезпечуватись оптимальні значення параметрів мікроклімату: температури, відносної вологості та рухливості повітря відповідно до норм та правил, а також ДБН В.2.5-67:2013 «Опалення, вентиляція та кондиціонування», затверджених наказом Мінрегіону від 25.01.2013 р. № 24.

Відповідно до санітарних норм мікроклімату виробничих приміщень ДСН 3.3.6.042-99 в офісних приміщеннях температура повітря повинна становити +22 - +24°С, відносна вологість повітря — 40–60%, швидкість руху повітря — не більше 0,1 м/с.

Як відомо, тривала робота за комп'ютером та з документами при недостатньому рівні освітленості може призвести до значного перенапруження зору, тому вимоги до освітлення є досить важливими.
Додатково, окрім вже перелічених документів, вимоги до освітлення встановлено ДБН В.2.5-28:2018 «Природне і штучне освітлення», затвердженими наказом Мінрегіону від 28.02.2019.

Робоче місце було розміщене таким чином, щоб уникнути попадання прямого світла в очі.

Рівні шуму та вібрації на робочих місцях осіб, що працюють з ПК, визначаються відповідно до ДСанПіН 3.3.2.007-98.
Для забезпечення дотримання допустимих рівнів шуму на робочих місцях застосовуються засоби звукопоглинання, вибір яких обґрунтовується спеціальними інженерно-акустичними розрахунками
Перелік організаційно-технічних заходів щодо обмеження несприятливого впливу шуму та вібрації на працюючих наведено в ДСН 2.3.6.037-99 та ДСН 3.3.6.039-99, серед яких зменшення шуму та вібрації на шляху розповсюдження засобами ізоляції та поглинання, наприклад, за рахунок використання гумових, поролонових, інших шумо- чи вібропоглинаючих матеріалів, або інших матеріалів аналогічного призначення, що дозволені

для оздоблення приміщень органами державного санітарно-епідеміологічного нагляду.

Допустимі параметри неіонізуючого електромагнітного випромінювання

Вимоги щодо рівня неіонізуючих електромагнітних випромінювань, електростатичних та магнітних полів встановлюються відповідно до ДСанПіН 3.3.2.007-98, а також вимог до роботодавців щодо захисту працівників від шкідливого впливу електромагнітних полів, затверджених наказом Міненергетики від 05.02.2014 р. № 99, ДСанПіН 3.3.6.096-2002.

Значення напруженості електростатичного поля на робочих місцях (як у зоні екрана дисплея, так і на поверхнях обладнання, клавіатури, друкувального пристрою) мають не перевищувати гранично допустимих відповідно до встановлених норм.

Таким чином аналіз методів розробки проводився у допустимих для роботи умовах праці, зберігаються правила охорони праці та пожежної безпеки.

6.2 Здоровий спосіб життя людини та його вплив на професійну діяльність при керуванні компёютерною системою

Здоровий спосіб життя, далі – ЗСЖ, все активніше входить в побут сучасної людини. З'являються напрямки, лабораторії, організації, що пропагують і впроваджують новий образ в звичайне життя людей. Сьогодні перед суспільством гостро постає безліч проблем. Як навчитися гармонійно жити у відповідності з об'єктивними законами не тільки самого суспільства, а й природи - як того вимагає біологічна основа людини; як не хворіти, зберігаючи високий робочий потенціал.

Ці питання привертають пильну увагу дослідників всіх галузей знань, які вивчають шляхи формування здорового способу життя як єдино правильного. Вивчення всіх складових даного напрямку привело до

переконання, що основою формування і підтримки ЗСЖ працівникам компёютерної системи є свідомість і рух. Саме вони визначають характер будь-якого прагнення до гармонійного розвитку особистості в системі «свідомість - тіло».

Прагнення до руху закладено в самій сутності людини, тому що рухова активність має величезну силу. Це ж помітив свого часу С.А. Тіссо, висловивши думку про те, що рух може замінити будь-які ліки, в той час як всі лікувальні засоби світу не можуть замінити дії руху [63].

Розвиток інтересів до ЗСЖ необхідно починати з формування потреб, які обов'язково повинні бути відображені в свідомості, інакше не відбудеться їх реалізація. Потреби співвідносяться з мотивами і мотивацією. Під мотивацією, вважає В.І. Ковальов, розуміється сукупність мотивів поведінки і діяльності [65].

Однак, як уточнює В.І. Глухів, визначальним моментом в мотивації все ж не є мотиви, а цілі діяльності і вже потім - відповідні їм мотиви. Отже, вміння ставити цілі діяльності, цілі здорового способу життя, прагнення до досягнення цих цілей є суттєвими характеристиками мотивації. Мета, таким чином, є невід'ємний компонент мотивації, яка є відображенням певної людської потреби. Мотивація людей, які ведуть здоровий спосіб життя, на думку В.І. Глухова, відображає ідейно-моральні джерела активності в забезпеченні фізкультурно-оздоровчої роботи, її конкретну спрямованість, стійкість і дієвість спонукань людини в боротьбі за своє здоров'я, переконаність у необхідності здорового способу життя.

Сутність теорії ЗСЖ полягає у використанні оздоровчих комплексів, що забезпечує позитивний вплив на організм людини, на зміцнення його здоров'я. Цей напрямок має перш за все показати шляхи формування здорового способу життя. Важливою ланкою на етапі формування здорового способу життя виступає правильність і чіткість сприйняття

елементів цього образу. Якщо ж цього не відбувається, то правильний образ може не сформуватися або відбитися у свідомості в спотвореному вигляді.

Тобто думка про ЗСЖ формує і реалізує рух в цьому ж напрямку. Важливо також сформувати у свідомості людини, що здоров'я - це капітал. Необхідно розкрити кожному, що цей капітал - надбання не тільки особистості, але і суспільства. Як сил, що спонукають до формування потреби в ЗСЖ, можуть виступати також різні види стимулів: матеріальні, духовні, соціальні, внутрішні, зовнішні і ін. Внутрішніми стимулами в цьому випадку є: активний емоційний стан, гарне самопочуття, висока м'язова активність, бадьорий настрій, висока працездатність. Ці ж явища виступають в якості засобів зворотної інформації про успішність і доцільність ведення ЗСЖ. Стимули можуть виходити від керівника, педагога, членів сім'ї, друзів та ін.,

В процесі засвоєння фізичної культури відбувається розвиток соціально значущих якостей особистості, активізація пізнавальних психічних процесів, створення нових ціннісних орієнтацій, а в підсумку - формування людини нового типу і мислення з його величезними потенційними можливостями [65; 66; 67]. Для успішної реалізації поставлених цілей необхідно також забезпечити займаються ті необхідні умови, в яких можливе проведення спеціальної роботи з формування ЗСЖ.

Отже, з огляду на вищесказане, структуру ЗСЖ і його елементи можна представити таким чином:

1. Усвідомлення необхідності вести здоровий спосіб життя.
2. Придбання знань (інформації) про ЗСЖ, формування та актуалізація потреб в ЗСЖ.
3. Формування мотивів ЗСЖ (зміцнення здоров'я, відмова від шкідливих звичок, збереження здоров'я, активізація відновних процесів, підвищення працездатності і т.д.).
4. Визначення цілей і завдань ЗСЖ.

5. Практична реалізація оздоровчих програм.
6. Контроль за змінами, що відбуваються в організмі, корекція дій, регуляція навантажень.

Підсумовуючи все вище сказане ми отримуємо висновок, що підтримуючи здоровий спосіб життя, займаючись щоденними вправами, приділяючи час прогулянкам на чистому повітрі та слідкуючи за своїм раціоном людина здатна суттєво підвищити рівень працездатності, ефективності та продуктивність на робочому місці.

7. ЕКОЛОГІЯ

7.1 Етапи та техніка збору та обробки екологічної інформації.

Використання інформаційного підходу, що базується на нових інформаційних технологіях, дозволяє не тільки кількісно описати процеси, що відбуваються в складних еко- і геосистемах, але і, змоделювавши механізми цих процесів, науково обґрунтувати методи оцінки стану різних компонентів навколишнього природного середовища.

Аналіз екологічної інформації включає:

1. аналіз ефектів впливу різних чинників на навколишнє середовище (виявлення критичних чинників впливу і найбільш чутливих елементів біосфери);
2. визначення допустимих екологічних впливів і навантажень на компоненти навколишнього середовища з урахуванням комплексного та комбінованого впливу на екосистему ;
3. визначення допустимих навантажень на регіон з еколого-економічних позицій.

Етапи інформаційного аналізу екологічної інформації включають наступні стадії:

1. збір інформації про стан навколишнього середовища: експедиційні дослідження; стаціонарні дослідження; аеровізуальні спостереження; дистанційне зондування; космічна і аерофотозйомка; тематичне картографування; гідрометеорологічні спостереження; система моніторингу; літературні, фондові та архівні дані;
2. первинна обробка і структуризація: кодування інформації; перетворення в машинну форму; цифрование картографічного матеріалу; обробка зображень; структуризація даних; приведення даних до стандартного формату;

3. Заповнення бази даних і статистичний аналіз: вибір логічної організації даних; заповнення бази даних і редагування; інтерполяція і екстраполяція відсутніх даних; статистична обробка даних; аналіз закономірностей в поведінці даних, виявлення трендів і довірчих інтервалів;
4. моделювання поведінки екосистем: використання ускладнюються моделей; варіювання граничними умовами; імітація поведінки екосистем при одиничних впливах; картографічне моделювання; дослідження діапазонів відгуку при різних впливах;
5. експертне оцінювання: оцінка діапазонів зміни впливів на екосистеми; оцінка поведінки екосистем при різних впливах за принципом «слабкої ланки»;
6. аналіз невизначеності: вхідних даних; параметрів моделей; результатів моделювання; величин експертних оцінок;
7. виявлення закономірностей і прогнозування екологічних наслідків: розробка можливих сценаріїв поведінки екосистем; прогнозування поведінки екосистем; оцінка результатів різних сценаріїв;
8. прийняття рішень щодо обмеження впливу на навколишнє природне середовище: вироблення «щадять» (зберігаючих) стратегій скорочення впливів на навколишнє природне середовище; обгрунтування обраних рішень (екологічне та соціально-економічне).

Експертно-моделююча геоінформаційна система являє собою об'єднання загальним призначеним для користувача інтерфейсом звичайної ГІС з оболонкою експертної системи і блоком математичного моделювання.

Критичні навантаження (КН) на екосистеми - це «максимальне випадання подкисляющіх з'єднань, що не викликає протягом тривалого періоду шкідливих наслідків для структури і функцій цих екосистем».

Критичні навантаження є індикатором стійкості екосистем. Вони забезпечують значення максимально «можливо розв'язати» навантаження забруднюючої речовини, при якій практично не відбувається руйнування біогеохімічної структури екосистеми. Чутливість екосистеми наприклад, до кислотних випадання може бути визначена виміром або оцінюванням певних фізичних або хімічних параметрів екосистеми; тим самим може бути ідентифікований рівень кислотних випадінь, який не робить або робить украй незначний вплив на цю чутливість.

На даний момент екологічні ГІС представляють собою складні інформаційні системи, що включає потужну операційну систему, інтерфейс користувача, системи ведення баз даних та відображення екологічної інформації. Вимоги до екологічної ГІС співзвучні вимогам до ідеальної ГІС:

1. можливість обробки масивів покомпонентної гетерогенної просторово-координованої інформації;
2. здатність підтримувати бази даних для широкого класу географічних об'єктів;
3. можливість діалогового режиму роботи користувача;
4. гнучка конфігурація системи, можливість швидкого налаштування системи на рішення різноманітних завдань;
5. здатність «сприймати» і обробляти просторові особливості геоекологічних ситуацій.

Велике значення має здатність сучасних ГІС перетворювати наявну екологічну інформацію за допомогою різних моделей (здатність до синтезу).

7.2 Вимоги до приміщень для експлуатації моніторів і ПЕОМ

Шляхи дотримання цих вимог. Приміщення для роботи на ПЕОМ повинні бути обладнаними аптечкою для надання першої медичної допомоги і вуглекислотними вогнегасниками.

Основні правила експлуатації користувачем засобів обчислювальної техніки. Перед початком роботи необхідно:

- злегка вологою ганчірочкою видалити пил з екрану і поверхні монітора, при цьому вимикач і роз'єм електроживлення повинен бути вимкненим;
- провести зовнішній огляд пристроїв, кабелів живлення і інтерфейсу.
- При виявленні механічних пошкоджень корпусів, екрана монітора, кабель живлення і інтерфейсу користувачів повинен негайно звернеться в Ремонтну службу.

Включення живлення техніки виробляється відповідним перемикачем на корпусі при підключення до мережі роз'ємом живлення.
Забороняється підключати або відключати роз'єм живлення до мережі при включенні перемикачі на корпусі пристрою
Після закінчення роботи слід відключити монітор, блок живлення комп'ютера, принтер і ін. Оргтехніки, відключити роз'єм живлення.
Забороняється експлуатувати техніку:

1. визнану фахівцем Центру несправної або непридатною до експлуатації;
2. не підключену до контуру заземлення;
3. має механічні пошкодження корпусів;
4. має порушення пломб;
5. має несправне електроживлення.

Вимоги до приміщень для експлуатації ПЕОМ.

Приміщення з ПЕОМ повинні мати природне і штучне освітлення.

Вікна повинні бути орієнтовані переважно на північ і північний схід.

Штучне освітлення повинно здійснюється системою загального рівномірного освітлення. У випадках переважне роботи з документами, допускається застосування комбінованого освітлення (додатково світильники місцевого освітлення). Освітленість на поверхні стола повинна бути 300-500 лк.

Джерела штучного освітлення повинні застосовуватися переважно люмінесцентні лампи тип ЛБ. Застосування світильників без розсіювачів не допускається. У приміщеннях слід проводить чистку скла віконних рам і світильників не рідше двох разів на рік і проводить своєчасну заміну перегорілих ламп.

Площа на одне робоче місце з ПЕОМ повинна бути не менше 6,0 кв.м, а обсяг - не менше 24,0 куб.м.

Звукоізоляція приміщень повинна відповідати гігієнічним вимогам і забезпечувати нормовані параметри шуму не більше 50 дБА.

Приміщення повинні бути обладнані системами опалення, конденціонірованія повітря або припливно-витяжною вентиляцією.

У приміщенні, де експлуатується ПЕОМ, повинні підтримуватися наступні кліматичні умови:

1. температура повітря - 21-24 градусів С;
2. відносна вологість повітря - 10 - 80% без конденсації;
3. вібрація - 0,25 - 55Гц.

Дотримуючись цих правил для працівників буде забезпечено комфортні та безпечні умови праці, без загрози погіршення стану здоров’я через недоліки екологічної складової місця праці.

ВИСНОВКИ

У даній дипломній роботі було проведено комплексний аналіз, щодо порівняння існуючих методів розробки захищених ERP-систем.

У ході дослідження і аналізу існуючих проектів було виявлено основні переваги таких рішень.

Перевагою ERP-системи є хороші показники швидкодії під час обробки інформації про рахуноки компанії, оформлення замовлень та ведення аудиту ресурсів підприємства.

Проведено дослідження по збору інформації щодо можливих вразливостей притаманних веб-проектам. На основі зібраних даних та інформації від видання OWASP було виділено найбільш поширені та небезпечні загрози.

Після дослідження та збору даних про можливі загроз нами було проведено порівняльний аналіз методів розробки ERP-систем. Серед варіантів розробки було обрано метод з використанням багатофункціональних бібліотек, а саме комплексних фреймворків. Серед лідерів на території України було виділено Laravel, Symfony, YII, Zend Framework та Bitrix Framework.

У результаті порівняння функціоналу, який забезпечить стабільну роботу ERP-системи, а також здатний запропонувати варіанти захисту критично важливих модулів системи, нами було обрано Bitrix Framework.

Логіка роботи даного фреймворку дозволяє уникнути прямих запитів до бази даних, що суттєво знижує можливість допущення критичних помилок під час роботи проекту.

У третьому розділі нами було продемонстровано чіткий механізм по налаштуванню всіх пунктів модулю “Проактивний захист”, який містить у собі функціонал захисту від DDOS атак, відслідковування змін у

функціональних файлах скриптів, безпечна авторизація без SSL, проактивний фільтр та інструмент аудиту для безпечного PHP-коду.

На період розробки ERP-система була захищена паролем, який був реалізований за допомогою правил роботи веб-сервера у файлі .htaccess.

Після реалізації проекту усім виконуваним файлам встановлено дозволи тільки на читання, що унеможливлює їх зміну за допомогою можливих вразливостей.

Для інформаційних блоків проведено комплексне налаштування, що дозволяє запобігти зміни інформації користувачами, які мають недостатні права.

На основі наукових робіт було виявлено, що основною проблемою ERP-системи є можливість несанкціонованого доступу. Дану проблему ми вирішили за допомогою розширення стандартного функціоналу 1С:Бітрікс, за допомогою фільтрації вхідних запитів.

Фільтрація відбувається на основі перевірки часу запиту на вхід, IP-адреси, якої відбувається спроба увійти та наявність користувацького аккаунту у “білому листі”.

Отже в кінцевому результаті нами було реалізовано ERP-систему, яка відповідає усім критеріям безпеки, що унеможливлює витоки інформації за межі системи, а також забезпечує захист від атак типу “відмова у обслуговуванні”.

Бібліографія

1. Что такое erp система?, 2018.url:https://seosreda.com.ua/chto-takoe-erp-sistema/) (дата звернення: 25.10.2019)
2. Топ 10 erp систем для украины,2018.url:https://www.livebusiness.com.ua/tools/erp/. (дата звернення: 25.10.2019)
3. Хочу всё знать. Язык abap ,url:https://geekbrains.ru/posts/abap. (дата звернення: 25.10.2019)
4. Криптографічний захист інформації, що циркулює в інформаційних ресурсах erp-систем / м. М. Степанов, в. В. Вишнівський, в. Л. Бурячок, і. Р. Пархомей // зв'язок. - 2018. - № 2. - с. 60-63. - url:http://nbuv.gov.ua/ujrn/zvjazok_2018_2_13(дата звернення: 25.10.2019)
5. Харченко ю. А. Аналіз сучасних систем управління ресурсами підприємства, 2018.url:http://eprints.kname.edu.ua/5903/1/103-110%d0%a5%d0%b0%d1%80%d1%87%d0%b5%d0%bd%d0%ba%d0%be_%d0%ae%d0%90.pdf.(дата звернення: 25.10.2019)
6. Власов а. П. Анализ современных erp-систем, 2019.
7. Баронов, в.в. Особенности использования и внедрения erp - систем, 2019. Url:http://www.citforum.ru/seminars/cis99/epr.shtml(дата звернення: 25.10.2019)
8. Объем и крупнейшие игроки мирового рынка erp-систем url:http://www.tadviser.ru(дата звернення: 25.10.2019)
9. Типовые пакеты решений (тпр) sap business all-in-one - http://www.businessone.ru/solutions/ sapbo40.asp#_toc11. (дата звернення: 25.10.2019)
10. Microsoft business solutions - navision url:http://www.mcdsoft.ru/3-1-1.htm.(дата звернення: 25.10.2019)

11. Отраслевое решение галактика машиностроение url:http://www.galaktika.by (дата звернення: 25.10.2019)
12. Полнофункциональная erp-система "компас" url:http://www.compas.ru/ (дата звернення: 25.10.2019)
13. Ганзен, в. А. Системные описания в психологии,2019. Url:http://www.medbookaide.ru/books/fold1002/ book1226/p9.php (дата звернення: 25.10.2019)
14. Owasp - top 2019, 2019 url:https://www.owasp.org/images/9/96/owasp_top_10-2017-ru.pdf.
15. Prashant, (дата звернення: 25.10.2019)url:uttps://pdfs.semanticscholar.org/0d1c/57269a6437caf883880cc5ba2e71101f8196.pdf.(дата звернення: 25.10.2019)
16. Jingchi z. Cross-site scripting (xss) detection integrating evidences in multiple stages , / z. Jingchi, j. Yu-tsern, l. Xiangyang. – 2019.url:https://scholarspace.manoa.hawaii.edu/bitstream/10125/60153/0713.pdf.(дата звернення: 25.10.2019)
17. Roshan s. Defending cross-site request forgery (csrf) attacks on web applications , / shaikh roshan. – 2019.url:https://search.proquest.com/openview/8535a9862f4b8e8b7c82b25cf7aec196/1?pq-origsite=gscholar&cbl=18750&diss=y. (дата звернення: 25.10.2019)
18. Голян в. В. Порівняння моделей життєвих циклів програмного забезпечення з метою виявлення найефективнішого, 2019. Url:http://www.hups.mil.gov.ua/periodic-app/article/19328/soi_2019_2_10.pdf. (дата звернення: 11.11.2019)
19. Дригалкин в. В. Html в примерах. Как создать свой web-сайт : самоучитель. [текст] / в.в. Дригалкин. – м. И др. : даилектика, 2018. – 190 с.

20. Выбор cms систем, 2018. Url:https://babosik.ru/158-cms-or-itsengine.html (дата звернення: 11.11.2019)
21. Что такое cms?, 2018 url: http://webstudio2u.net/ru/programming/96-cms.html (дата обращения: 20.02.17) (дата звернення: 11.11.2019)
22. Php 5 cms framework development - 2nd edition, 2019. Url:https://books.google.com.ua/books?hl=uk&lr=&id=wyqyktv7tjoc&oi=fnd&pg=pt15&dq=what+is+framework+php&ots=4sr1hdpr4_&sig=8iovlu6d0kgsvccaojjkpg1ilc8&redir_esc=y#v=onepage&q=what%20is%20framework%20php&f=false. (дата звернення: 11.11.2019)
23. Otwell Taylor Otwell - Creator of laravel, 2019. Url:https://github.com/taylorotwell. (дата звернення: 11.11.2019)
24. A comparative study of laravel and symfony php frameworks, 2019. Url:https://www.researchgate.net/profile/abir_yamami/publication/330656531_a_comparative_study_of_laravel_and_symfony_php_frameworks/links/5c4c9067458515a4c7424c9d/a-comparative-study-of-laravel-and-symfony-php-frameworks.pdf. (дата звернення: 11.11.2019)
25. Шакиров А.А, зарипова р.с современные тенденции web-разработки , 2019 .url:http://journal-s.org/index.php/sisp/article/view/12105/pdf. (дата звернення: 11.11.2019)
26. Dennis durairaj. Comparison and analysis of web technologies in node, php and python-django, 2019.url:http://repo.bg.pw.edu.pl/index.php/en/r#/info/master/wutb1f1c5da382e40798061dbe1f13cccae/. (дата звернення: 11.11.2019)
27. https://pidruchniki.com/1701120547727/informatika/modeli_zhittyevogo_tsiklu (дата звернення: 11.11.2019)
28. Kevin Bunglass. Persistence in php with the doctrine orm. Url:https://books.google.com.ua/books?hl=uk&lr=&id=0ntdagaaqbaj&oi=fnd&pg=pt5&dq=orm+doctrine&ots=e6-x4hckmh&sig=lsegmczgptm5bl4bdydas-

hipic&redir_esc=y#v=onepage&q=orm%20doctrine&f=false. (дата звернення: 15.11.2019)

29. Raw url:https://twig.symfony.com/doc/2.x/filters/raw.html (дата звернення: 15.11.2019)

30. Ceating form classes,. Url:https://symfony.com/doc/current/forms.html#creating-form-classes (дата звернення: 15.11.2019)

31. Фильтры. Url:https://yiiframework.com.ua/ru/doc/guide/2/structure-filters/ (дата звернення: 15.11.2019)

32. Query builder. Url:https://www.yiiframework.com/doc/guide/1.1/en/database.query-builder (дата звернення: 15.11.2019)

33. Zend_db_select. Url:https://framework.zend.com/manual/1.12/ru/zend.db.select.html (дата звернення: 15.11.2019)

34. Owasp .url:https://blog.sucuri.net/2019/01/owasp-top-10-security-risks-part-v.html. (дата звернення: 15.11.2019)

35. Высоконагруженные приложения. Программирование, масштабирование, поддержка, 2018.

36. Гладкий а. Веб-самоделкин. Как самому создать сайт быстро и профессионально, 2014. – 250 с.

37. Васвани В. Разработка веб-приложений на php / викрам васвани. – спб: питер, 2014. – 389 с.

38. а. С. Строганов. «ваш первый сайт с использованием php-скриптов». – диалог мифи., москва, 2012. – 288 с.

39. Справочник по html i css, 2019. Url:http://htmlbook.ru (дата звернення: 21.11.2019)

40. Шевченко В.Л. Несанкціонований доступ до інформаційних ресурсів erp-системи , 2019. Url:http://www.irbis-nbuv.gov.ua/cgi-bin/irbis_nbuv/cgiirbis_64.exe?c21com=2&i21dbn=ujrn&p21dbn=ujrn&image_file_download=1&image_file_name=pdf/znpcvsd_2014_1_2.pdf. (дата звернення: 21.11.2019)
41. Яковлев Георгий Олегович. Обеспечение безопасности сторонних компонентов веб приложений, 2019. Url:https://cyberleninka.ru/article/n/obespechenie-bezopasnosti-storonnih-komponentov-veb-prilozheniy/viewer. (дата звернення: 21.11.2019)
42. Михнев Илья Павлович. Защита конфиденциальной информации от несанкционированного доступа при проектировании автоматизированных систем радионуклидной спектрометрии на базе сцинтилляционного гамма-спектрометра, 2019. Url:https://www.researchgate.net/profile/ilya_p_mikhnev/publication/326730931_zashchita_konfidentsial'noi_informatsii_ot_nesanktsionirovannogo_dostupa_pri_proektirovanii_avtomatizirovannykh_sistem_radionuklidnoi_spektrometrii_na_baze_stsintilliatsionnogo_gamma-spektrometra/links/5c756ad192851c6950439f61/zashchita-konfidentsialnoi-informatsii-ot-nesanktsionirovannogo-dostupa-pri-proektirovanii-avtomatizirovannykh-sistem-radionuklidnoi-spektrometrii-na-baze-stsintilliatsionnogo-gamma-spektrometra.pdf. (дата звернення: 18.11.2019)
43. Савчук Т.О. Аналіз впливу обфускації програмного коду на виявлення шкідливого програмного забезпечення, 2019.
44. Висоцька Оолена Олександрівна, 2019. Url:http://er.nau.edu.ua:8080/bitstream/nau/40426/1/diser_vysotska.pdf. (дата звернення: 21.11.2019)

45. Bożena małysiak-mrozek, incorporating fuzzy logic in object-relational mapping layer for flexible medical screenings, 2019. Url:http://www.bookmetrix.com/detail/chapter/192b3815-032f-4c5f-b482-eb2fc89000b7#downloads (дата звернення: 21.11.2019)
46. Url:https://www.theseus.fi/bitstream/handle/10024/160666/thesis_developing_a_web_service.pdf?sequence=1
47. Markus moilanen, developing a web service databases, security and access control, 2019. (дата звернення: 21.11.2019)url:https://www.theseus.fi/bitstream/handle/10024/160666/thesis_developing_a_web_service.pdf?sequence=1 (дата звернення: 21.11.2019)
48. Symfony vs express: a server-side framework comparison, 2019. Url:http://www.diva-portal.org/smash/record.jsf?pid=diva2%3a1327290&dswid=-1912 (дата звернення: 21.11.2019)
49. Henryk Rybinski, intelligent methods and big data in industrial applications, 2019. Url:http://www.bookmetrix.com/detail/book/ace984d6-5141-48a0-9153-96898d138450#downloads (дата звернення: 25.11.2019)
50. Козич полина александровна , особенности использования современных php-фреймворков для реализации корпоративных веб-порталов, 2019. Url:http://e-postulat.ru/index.php/postulat/article/view/2372/2413 (дата звернення: 25.11.2019)
51. Indri Handayani1, designing popular classes on viewboard public assessment of lectures based on yii framework, 2019. (дата звернення: 25.11.2019)url:https://s3.amazonaws.com/academia.edu.documents/6049

1539/j.paper_5_att_v1n220190904-11116-ni1gak.pdf?response-content-disposition=inline%3b%20filename%3ddesigning_popular_classes_on_viewboard_p.pdf&x-amz-algorithm=aws4-hmac-sha256&x-amz-credential=akiaiwowyygz2y53ul3a%2f20191213%2fus-east-1%2fs3%2faws4_request&x-amz-date=20191213t193149z&x-amz-expires=3600&x-amz-signedheaders=host&x-amz-signature=652120f3ca1058ef23941cc269f811d3df341d998676098112f0fafa7538ca0b (дата звернення: 25.11.2019)

52. Implementation of business intelligence using highlights in the yii framework based attendance assessment system,2019, url:https://media.neliti.com/media/publications/288104-implementation-of-business-intelligence-f7a6ab2f.pdf (дата звернення: 25.11.2019)
53. Ninda lutfiani, the online sales application of black and white print based on yii framework on higher education e-commerce website, 2019. (дата звернення: 25.11.2019) url:https://s3.amazonaws.com/academia.edu.documents/60491489/m._paper_2_att_v1n220190904-44318-1ct5gtw.pdf?response-content-disposition=inline%3b%20filename%3dthe_online_sales_application_of_black_an.pdf&x-amz-algorithm=aws4-hmac-sha256&x-amz-credential=akiaiwowyygz2y53ul3a%2f20191213%2fus-east-1%2fs3%2faws4_request&x-amz-date=20191213t193333z&x-amz-expires=3600&x-amz-signedheaders=host&x-amz-signature=5c9146be46f6d0d3009450d5bd3ca7c7168b67214744b59de30950ff789336fd (дата звернення: 01.12.2019) (дата звернення: 25.11.2019)
54. Nirmalasari, Harahap, e. P., & Faradilla, F. Implementation of problem formulation management in improving the quality of research in higher

education. Aptisi transactions on management, 2(1), 2018. 20-27c (дата звернення: 01.12.2019)

55. Rahardja, U., Lutfiani, N., Lestari, A. D., & manurung, e. B. P.. Inovasi perguruan tinggi raharja dalam era disruptif menggunakan metodologi ilearning. Jurnal ilmiah teknologi informasi asia, 13(1),2019 23-34c (дата звернення: 01.12.2019)
56. Choshin, M., & Ghaffari, a. An investigation of the impact of effective factors on the success of e-commerce in small-and medium-sized companies. Computers in human behavior, 66,2017 67-74c (дата звернення: 01.12.2019)
57. Rasouli, N., Abedi, l., & Ghaei, s. (2018). Designing an agent for information extraction from persian e-shops. Telkomnika, 16(1), 455-462c (дата звернення: 01.12.2019)
58. Объедков п.и., возможности использования системы "1с-битрикс: управление сайтом" для развития финансовой грамотности школьников, 2019. Url: https://elibrary.ru/item.asp?id=36858017 (дата звернення: 01.12.2019)
59. Добычин а.с, интеграция "1с:предприятие", "1с-битрикс:управление сайтом" и "1с:битрикс24" для повышения качества работы менеджера приемной кампании, 2019. Url:https://elibrary.ru/item.asp?id=36858023 (дата звернення: 02.12.2019)
60. Синицын о.в., онлайн-сервис "школьный олимп" на базе "1с-битрикс" как компонент внутришкольной системы оценки качества образования, 2019. Url: https://elibrary.ru/item.asp?id=36858376 (дата звернення: 01.12.2019)
61. информационные блоки ciblockelement getlist. Url: https://dev.1c-bitrix.ru/api_help/iblock/classes/ciblockelement/getlist.php (дата звернення: 01.12.2019)

62. Усова а.а сайт на 1с-bitrix для коммерческих компаний ведущих свой бизнес в интернете. Url:http://elib.sfu-kras.ru/bitstream/handle/2311/19028/s17_037.pdf?sequence=1 (дата звернення: 01.12.2019)