ЗМІСТ

ВСТУП

Організація діяльності й управління будь-якої установи здійснюється за допомогою великого обсягу документації. З плином часу в організації накопичуються й зберігаються багатотонні паперові архіви. Оформлення документів вимагає тривалого часу, кожен наказ доводиться кілька разів передруковувати при внесенні навіть незначних змін, а відстежити рух потрібних паперів дуже складно. З уведенням загальної комп'ютеризації процес обробки документів на конкретній ділянці або робочому місці значно спростився. Але для того, щоб діяльність установи була успішною, а управління ефективним, необхідно створити оперативний зв'язок між підрозділами з можливістю миттєвого обміну інформацією. Крім того, важливою є прозора схема відстеження змін у документах установи. Ці завдання вирішуються за допомогою автоматизації документообігу.

Використання сучасних інформаційних технологій прискорює документообіг, підвищує виконавчу дисципліну співробітників, удосконалює управління організації в цілому.

Питання автоматизації документообігу з розвитком технологій з обробки інформації набуває більшої актуальності, оскільки від правильного вибору технології роботи залежить успіх будь-якої організації.

Сьогодні установа, яка хоче отримати сучасне рішення для створення автоматизованої системи документообігу, має досить широкий вибір – від окремих приватних технологій до готових комплексних систем.

Актуальність роботи пов'язана зі значною поширеністю досліджуваної проблеми та полягає в необхідності використання сучасних інформаційних технологій у процесі автоматизації документообігу в закладі вищої освіти.

Мета і завдання роботи. Метою дипломної роботи є розробка системи автоматизації документообігу в університеті з можливістю синхронізації даних з єдиної державної електронної бази з питань освіти (ЄДЕБО).

Для досягнення поставлених цілей у дипломній роботі реалізовані такі завдання:

- розглянути особливості процесів документообігу та їх взаємозв’язок між собою;

- визначити основні процеси документообігу в закладі вищої освіти, які необхідно автоматизувати;

- вибрати інструментальні засоби розробки та проектування системи;

- розробити автоматизовану систему документообігу в закладі вищої освіти.

Об’єкт дослідження − автоматизація документообігу в закладі вищої освіти.

Предмет дослідження − розробка та впровадження інформаційної системи для ведення документообігу в закладі вищої освіти.

Практичне значення роботи. Автоматизація процесу документообігу дає змогу відмовитися від ручних форм обліку й перейти до сучасних форм реєстрації, обліку та зберігання документів. Зважаючи на те, що обсяг зареєстрованих документів в університеті постійно зростає, впроваджуються системи автоматизації в суміжних галузях, відтак використання нових технологій автоматизації документообігу є доцільним.

Структура й обсяг роботи. Робота складається зі вступу, трьох розділів, висновків, списку використаних джерел. Роботу викладено на 59 сторінках. Матеріали дослідження містять 7 рисунків, список використаних джерел, що налічує 50 позицій.

РОЗДІЛ 1. ТЕОРЕТИЧНІ АСПЕКТИ ПРОЦЕСІВ АВТОМАТИЗАЦІЇ ТА ОПТИМІЗАЦІЇ ІНСТИТУЦІЙНОГО ДОКУМЕНТООБІГУ

У першому розділі представлено основні поняття та принципи автоматизації діяльності установи; розглянуто цілі та завдання автоматизації господарської діяльності підприємства; описано сфери застосування і основні принципи побудови автоматизованих інформаційних систем; вивчено основні поняття документу й автоматизації документообігу в установі, їхні основні цілі та завдання; окреслено головні переваги автоматизації документообігу установи.

1.1. Автоматизація роботи установи, основні поняття

Кожна установа прагне до зниження витрат для того, щоб збільшити конкурентоспроможність й отримувати прибуток навіть в складних умовах ринку. Одним із основних чинників, який впливає на зниження витрат є зменшення питомих витрат на випуск продукції або послуги певного виду. Вже протягом кількох десятків років цю проблему успішно вирішують системи автоматизації роботи установи, основне призначення яких – це підвищення ефективності та продуктивності роботи будь-якої компанії.

Варто зазначити, що сьогодні важко уявити собі невелику організацію, не оснащену комп’ютерною технікою. Різноманітні процеси, які потребують використання сучасних інформаційних технологій, стають невід’ємним складником щоденної господарської діяльності будь-якої установи.

Зі свого боку, автоматизація – це дуже широке поняття. Кожен керівник і власник бізнесу може розуміти цей термін по-різному. Залежно від галузі діяльності та обсягу завдань певної компанії, які вирішуються у процесі автоматизації установи, значно відрізняються. Тут можна говорити про такі види автоматизації, як автоматизація управління, бюджетування, електронного документообігу, бізнес процесів, виробництва та інші. Першочергово, все здійснюється заради зменшення рутини і прискорення взаємодії та прозорості етапів [3].

Зауважимо, що автоматизація – один із напрямів науково-технічного прогресу, застосування саморегулюючих технічних засобів, економіко-математичних методів і систем управління, які звільняють людину від участі в процесах отримання, перетворення, передачі та використання енергії, матеріалів або інформації, істотно зменшують ступінь цієї участі або трудомісткість виконуваних операцій [3].

Передусім, автоматизація загалом являє собою комплекс дій і заходів технічного, організаційного й економічного характеру, який дає змогу знизити ступінь участі або повністю виключити безпосередню участь людини у здійсненні тієї чи тієї функції виробничого процесу або процесу управління.

Важливим є те, що автоматизація діяльності організації дає змогу збільшити виробництво й ефективність роботи, знизити витрати на людську працю, звільнити персонал від виконання рутинних завдань. Автоматизація у такий спосіб дає змогу зосередити увагу на основних бізнес-процесах і працювати більш продуктивно, вчасно реагуючи на зміни зовнішнього середовища, й ефективніше планувати подальші дії [1].

Отже, завданнями автоматизації є:

- переведення значної частини паперового документообігу в електронну форму;
- заощадження часу та коштів на проведення розрахунків, ведення необхідної документації;
- оперативне отримання даних на всіх ділянках роботи й постійний контроль над показниками діяльності компанії;
- збереження великого обсягу інформації та гарантія конфіденційності;
- організація й оформлення накопиченої інформації у відповідному форматі для підвищення ефективності та правильності прийняття рішень персоналом установи;
- зменшення витрат на зберігання й обробку інформації;

- прискорення і спрощення роботи з інформацією;
- підвищення ефективності роботи співробітників компанії.

Слід зауважити, що при проведенні автоматизації чи обговоренні питання про необхідність її проведення, будь-яка установа прагне до досягнення таких цілей:

1. Скоротити витрати праці на виконання певного обсягу робіт. Прикладом може слугувати підготовка бухгалтерської звітності або процес відвантаження продукції. За допомогою використання засобів автоматизації трудомісткість таких процедур скорочується в кілька разів.

2. Виключити повторювані процедури або ділянки бізнес процесу. Наприклад, оформлення первинних документів виконує один співробітник, а їх реєстрацію та облік інший. Після впровадження засобів автоматизації реєстрація й облік первинних документів відбуватиметься автоматично.

3. Автоматизувати засоби контролю, звітності й аналізу. Найбільшою складністю в роботі будь-якої установи, як правило, є отримання оперативних підсумків роботи за довільний інтервал, контроль ефективності торгового чи виробничого персоналу, отримання аналітичної інформації в розрізі співробітників, товарів, контрагентів, показників бухгалтерського та податкового обліків. Комплексна автоматизація обліку успішно вирішує таке завдання і підвищує «прозорість» бізнес процесу всередині компанії. Формалізація та централізація знань, отриманих у процесі роботи установи, формалізація функцій і критеріїв співробітників дасть змогу адміністрації проводити більш правильну організаційну та кадрову політику, оцінювати загальну схему роботи, ліквідувати слабкі місця в роботі установи.

4. Збирати, накопичувати й вести облік комерційної інформації. Кожна установа прагне безперервно покращувати якість обслуговування своїх замовників. Найважливішою складовою цього процесу є ведення історії клієнтів, історії їх замовлень, рівня задоволеності сервісом і багато іншого.

5. Підвищити ефективність установи. Результатом досягнення

перерахованих цілей стає поліпшення економічних показників установи, його ефективності [2].

Зважаючи на сказане вище, головною метою автоматизації господарської діяльності установи є, передусім, отримання максимального прибутку від діяльності при використанні мінімальних витрат і підвищення ефективності роботи установи задля отримання більшого прибутку.

Слід зазначити, що поняття «автоматизація» не може існувати без таких категорій, як інформаційні системи та інформаційні технології.

Перейдемо до більш детального тлумачення терміна «інформаційна технологія (ІТ)». Перш за все, це сукупність методів і засобів, для збору, передачі, обробки, зберігання та видачі інформації, які виконуються в певній послідовності з сукупністю засобів комп'ютеризації. Тоді як, інформаційна технологія невіддільна від технічної, програмної, організаційної та нероздільна від того середовища, в якому вона реалізована [4].

Варто підкреслити, що автоматизована інформаційна технологія (АІТ) являє собою сукупність методів і способів збору, передачі, накопичення, зберігання, пошуку та обробки інформації на основі застосування засобів обчислювальної техніки та зв'язку. Основним завданням сучасних інформаційних технологій організаційного управління є своєчасне надання достовірної достатньої за обсягом інформації фахівцям і керівникам для прийняття обґрунтованих управлінських рішень.

Інформаційна система (ІС) – це система, яка реалізує інформаційну модель предметної області, найчастіше – будь-якої сфери людської діяльності. Інформаційна система повинна забезпечувати: отримання (введення або збір), зберігання, пошук, передачу та обробку (перетворення) інформації [5].

Таким чином, автоматизована інформаційна система (АІС) – людино-машинна система з автоматизованою технологією одержання результатної інформації, необхідної для інформаційного обслуговування фахівців та оптимізації процесу управління в різних сферах людської діяльності.

1.2. Сфери застосування, цілі та принципи автоматизації роботи установи

Перш ніж говорити про цілі, треба ознайомитись зі сферами автоматизації. Необхідно зазначити, що певного списку галузей застосування автоматизації не існує, через їхній індивідуальний характер. Однак можна виділити найбільш типові напрями автоматизації, які дають змогу зрозуміти, по-перше, ділянки, що автоматизують різні компанії, по-друге, визначити, що необхідно автоматизувати в цій сфері.

Незважаючи на велику кількість галузей автоматизації в установі та значний перелік поставлених завдань, основними цілями автоматизації діяльності установи є:

- Збір, обробка, зберігання та надання даних про діяльність організації, зовнішньому середовищі у тому вигляді, який був би зручний для фінансового аналізу, прийнятті управлінських рішень;
- Автоматизація виконання бізнес операцій (технологічних операцій), що становлять цільову діяльність установи;
- Автоматизація процесів, що забезпечують виконання основної діяльності [7].

Варто зауважити, що постановка цілей у процесі автоматизації діяльності установи, є одним з найважливіших етапів. Кожна поставлена мета має бути кількісно обґрунтована певним результатом. Таким чином, саме на етапі постановки цілей і завдань необхідно визначити, який спосіб автоматизації буде найбільш оптимальним для конкретної установи з урахуванням розмірів компанії, постановки менеджменту, перспектив розвитку, стану конкурентного середовища [7].

У сучасній теорії інформаційних технологій виділяють чотири види підходів до автоматизації діяльності установи. Охарактеризуємо детально кожен з них.

Перейдемо до ґрунтовного аналізу одного з підходів автоматизації діяльності установи – кускова (хаотична) автоматизація, яка є одним з

найменш ефективних видів інвестування коштів у розвиток установи. При цьому підході процес впровадження інформаційних технологій визначається локальними завданнями, а не реальними потребами бізнесу та характеризується відсутністю стратегічного плану. В результаті установа отримує розрізнені системи, що вимагають інтеграції або незакінчені фрагменти інформаційної інфраструктури. Однією з причин такого підходу є неправильне розуміння своєї ролі й функцій відділу інформаційних технологій [8].

Автоматизація за ділянками – це процес автоматизації окремих підрозділів підприємства, об'єднаних за функціональною ознакою. Даний підхід застосовується при недостатньому обсязі інвестицій для повної автоматизації, в разі отримання значного економічного ефекту від автоматизації. Умовами ефективності такого підходу є регулярно оновлюванні стратегічний і оперативний плани. Особливу увагу слід приділити питанням наступності комплексу стандартів на інформаційні технології [8].

Автоматизація за напрямками передбачає автоматизацію окремих напрямків діяльності, таких, як виробництво, збут, управління фінансами. Цей підхід враховує участь у процесі всіх служб і підрозділів, функціонування яких пов'язано з напрямком автоматизації. Реалізація цього підходу пов'язана з наявністю телекомунікаційної інфраструктури та вимагає створення моделі всієї установи. Ревізію стратегічного плану рекомендується проводити після закінчення автоматизації напрямків і оцінок результатів [8].

Зазначимо, що комплексна автоматизація діяльності підприємства передбачає об'єднання й узгодження управлінських функцій і процедур задля оптимізації процесу управління установою. Автоматизована система управлінням установи будується з орієнтацією на управління виробничим процесом як єдиним цілим. Цей вид автоматизації сприяє подоланню бар'єрів між різними службами управління, при підготовці управлінських рішень, заснованих на одних і тих самих даних. При реалізації цього підходу

першочергово увага приділяється питанням інтеграції. До особливостей зазначеного підходу належать підвищена економічна ефективність і надмірно високі вимоги до якості управління процесом впровадження системи [10].

Незважаючи на обрані сфери та підходи автоматизації діяльності установи, існує ціла низка загальних принципів, які висуваються до системи: системність, гнучкість, стійкість, ефективність.

Перейдемо до більш детального тлумачення кожного із зазначених понять. Системність автоматизованої інформаційної системи слід розглядати як системи, структура яких визначається функціональним призначенням. Гнучкість системи означає пристосованість до можливих перебудов, завдяки модульності побудови всіх підсистем і стандартизації їх елементів. Принцип стійкості полягає в тому, що автоматизована інформаційна система повинна виконувати основні функції незалежно від впливу на неї внутрішніх і зовнішніх чинників. Це означає, що помилки в окремих її частинах повинні бути легко усуненні, а працездатність системи швидко поновлювана [9].

Слід зазначити, що ефективність автоматизованої інформаційної системи розуміємо як інтегральний показник рівня реалізації наведених вище принципів, віднесений до витрат на створення й експлуатацію системи. Функціонування автоматизованої інформаційної системи може дати бажаний ефект за умови правильного розподілу функцій і навантаження між людиною та машинними засобами обробки інформації, ядром якої є комп’ютер.

Важливим є те, що створення й використання автоматизованої інформаційної системи ґрунтується на загальних принципах проектування систем обробки даних. До основних принципів належать:

1. Принцип максимальної орієнтації на кінцевого користувача. Цей принцип реалізується через створення спеціальних засобів адаптації автоматизованої інформаційної системи до рівня підготовки користувача і до можливості його навчання і самонавчання. Тому автоматизовані системи часто забезпечується спеціальними демонстраційними роликами. Необхідно, щоб введення нових даних і корегування інформації супроводжувалися

автоматизацією операцій, вбудованим контролем і системою підказок, що дає змогу швидко вивчити роботу некваліфікованому в комп'ютерній сфері працівнику [15, с. 25].

2. Проблемна орієнтація. Кожен модуль автоматизованої інформаційної системи спеціалізується на вирішенні певного класу завдань, об'єднаних загальною технологією обробки даних, єдністю режимів роботи та єдністю алгоритмів обробки даних.

3. Принцип відповідності інформаційних потреб користувачів, які використовують технічні засоби. Характеристики застосованих технічних засобів повинні відповідати обсягу інформації й алгоритмам її обробки. Це означає, що тільки після ретельного аналізу інформаційних потреб користувачів можна приступати до визначення складу та функцій автоматизованої інформаційної системи [15, с. 36].

4. Принцип творчого контакту розробників автоматизованої інформаційної системи та їхніх потенційних користувачів. Спільна робота користувача й розробника у створенні автоматизованої інформаційної системи допомагає краще усвідомити проблемну ситуацію, стимулює інтелектуальну діяльність майбутнього користувача і сприяє підвищенню якості автоматизованої інформаційної системи. Також повинна бути оформлена відповідна документація, яка містить пояснення до завдань, що виконуються за допомогою автоматизованої інформаційної системи, інструкцію з установки й експлуатації, інструкції щодо заповнення та ведення вхідних і вихідних документів.

1.3. Поняття та цілі автоматизації документообігу в роботі установи

Зауважимо, що управління будь-якою установою передбачає інформаційний процес, в якому інформація приймається, обробляється, зберігається. На її основі виробляється управлінське рішення, яке доводиться до виконавців, чиї дії контролюються. Документи – це основні інформаційні

ресурси установи, робота з якими вимагає правильної організації. Документи забезпечують інформаційну підтримку прийняття управлінських рішень на всіх рівнях і супроводжують ведення всіх бізнес-процесів.

Варто зауважити, що документ являє собою сукупність трьох складників:

1. Фізична реєстрація інформації.
2. Форма подання інформації.
3. Активізація певної діяльності.

Зауважимо, що саме певна діяльність і перетворює інформацію в документ. Однак документ перестає існувати, якщо в подальшому не має на увазі процедури обробки. При цьому форма документу тісно пов’язана з характером подальшої діяльності, вона породжує необхідність документів. Важливим є те, що, документ – це погано структурована сукупність блоків або об’єктів інформації. Загалом обійтися без документів неможливо. Сам по собі документ, незалежно від його форми (паперовий або електронний), не вирішує проблем установи, оскільки головним складником роботи установи є бізнес-процеси і чіткий контроль за їх виконанням [17].

Документообіг – це рух документів від моменту їх створення до моменту закінчення роботи з ними. Документообіг – це безперервний процес руху документів, об’єктивно відображає діяльність установи й дає змогу оперативно керувати ним. Тривалий пошук потрібного документа, втрати, дублюючі документи, затримки з відправкою й одержанням, помилки персоналу можуть значно загальмувати, а то й паралізувати роботу установи, особливо, якщо вона має в своєму складі територіально віддалені підрозділи. У таких випадках достатньо важко організувати наскрізний документообіг, а відтак і централізоване оперативне управління [23].

Слід зауважити, що документообіг як технологічний процес поділяється на кілька частин – потоків. Вони здійснюють прямий і зворотний зв’язок в управлінні. Документообіг може бути двох типів.

Перейдемо до більш детального розгляду першого типу документообігу. Універсальний документообіг автоматизує наявні інформаційні потоки недостатньо структурованої інформації. Справедливо було б назвати його аморфним або безладним документообігом.

Другий тип документообігу – операційний, передусім, орієнтований на роботу з документами, що містять операційну атрибутику [24, с. 83-85].

Крім документів, важливого значення набуває регламент роботи з ними. Будь-який досвідчений менеджер може підтвердити, що робота не за регламентом часом забирає набагато більше часу, ніж власне виробнича діяльність. Дублювання документів, їх втрата, нав'язливий спосіб їх поширення, а також заплутаний порядок їх проходження можуть істотно ускладнити роботу, підвищивши вірогідність допущення помилки внаслідок, наприклад, втрати потрібної інформації [28].

Отже, документ займає чільне місце у процесі певної діяльності на кордоні поділюваних функцій виконання. Тому правильно розглядати документ як інструмент розподілу функцій між працівниками.

Для того щоб налагодити ефективний документообіг в організації, на сьогоднішній день необхідно впровадити в роботу автоматизовану систему документообігу. Сьогодні досвід роботи великої кількості установ та інституцій доводить, що чим краще організована робота з документацією, тим швидше здійснюється пошук і обробка необхідної інформації і вищій рівень продуктивності компанії [29].

Варто зазначити, що електронним документообігом називається сучасна система ведення ділової документації, в якій всі створювані, що надсилаються і отримані документи обробляються за допомогою спеціальної комунікаційної технології на персональних комп'ютерах, які об'єднані в єдину мережу, що передбачає формування та ведення розподілених баз даних. При цьому введення електронного документообігу не є процесом відмови від паперових документів, проте вважається пріоритетним напрямом оптимізації і корекції роботи з документами. Ця система являє собою

специфічний механізм, що працює з електронними документами та включає концепцію без паперового діловодства [33].

Зазначимо, що системи електронного документообігу забезпечують зберігання документів, ведуть історію записів, створюють їхній рух, дають змогу стежити за здійсненням різних процесів, для яких, власне, вони і мають пряме відношення. В установі з активною системою документообігу, основним інструментом управління є документ. При цьому рішення, доручення або накази не мають ніякого значення. Для цих цілей існують відповідні документи, в яких містяться всі ці рішення, доручення, накази [37, с. 165].

В останні роки відбулося переосмислення ролі електронного документообігу. Якщо раніше він сприймався переважно як засіб автоматизації діловодства, то тепер його все частіше розглядають як інструментарій управління знаннями і інтеграції бізнес-процесів, в ході виконання яких створюються та рухаються документи. Тобто автоматизація процесів документообігу розглядається не як окрема система, а як частина автоматизованої інформаційної системи.

Використання автоматизованої інформаційної системи являє собою застосування бази для загального вдосконалення управління установами. Для автоматизації діяльності певної інституції необхідно створити єдиний інформаційний простір, в якому співробітники та керівництво можуть здійснювати свою діяльність, послуговуючись єдиними правилами подання та обробки інформації в документи, а також організувати єдину інформаційну систему, що охоплює основні завдання з обліку для різних категорій користувачів [13].

Створення та впровадження системи автоматизації документообігу переслідує досягнення таких цілей. Перейдемо до детального роз’яснення кожної мети у сфері обробки документів:

- забезпечення підвищення оперативності та якості роботи з документами, впорядкування документообігу, забезпечення контролю

виконання;

- створення умов для переходу від традиційного паперового документообігу до електронного без паперової технології;

- створення необхідних умов для підвищення інтелектуальної продуктивної частки у змістовній і смисловій роботі з документами та зниження трудовитрат на рутинні операції;

- забезпечення підвищення якості документів, що створюються в організації;

- виключення дублювання роботи по введенню інформації про документ на різних ділянках роботи з ним [12].

Розглянемо детально цілі, які передбачаються у сфері контролю виконавчої дисципліни:

- забезпечення автоматизованого контролю за проходженням документів у підрозділах організації з моменту їх отримання або створення до завершення виконання, відправки або оформлення у справу;

- забезпечення автоматизованого контролю за своєчасним виконанням документів, оперативне отримання інформації про стан виконання і місце знаходження будь-якого документа;

- скорочення термінів проходження і виконання документів.

Необхідно зазначати, що однією з важливих цілей у сфері організації доступу до інформації є забезпечення централізованого зберігання текстів документів, підготовлених в електронній формі, і їх графічних образів, а також всіх супровідних матеріалів (реєстраційних карток документів, резолюцій, супровідних документів) з можливістю організації логічного зв’язування документів, що відносяться до одного питання, і оперативного пошуку (добірки) документів за тематичним набором реквізитів [14, с. 54-58].

До основних переваг електронного документообігу можна віднести:

- єдиний порядок індивідуальної та спільної роботи з документами в підрозділах організації;

- об’єднання потоків електронних документів між підрозділами

організації;

- використання загальної для всіх організацій системи індексації (нумерації) документів, загальних довідників-класифікаторів (таких як перелік організацій, номенклатура справ), єдиної форми реєстраційно-контрольної картки документів тощо;

- забезпечення уніфікації управлінської документації та скорочення кількості форм і видів однакових документів;

- повний контроль за переміщенням й еволюцією документа, регламентація доступу і спосіб роботи користувачів з різними документами і їх окремими частинами;

- зменшення витрат на управління за рахунок вивільнення людських ресурсів, зайнятих різними видами обробки паперових документів, зниження бюрократичної тяганини за рахунок переміщення документів і жорсткого контролю за порядком і термінами проходження документів [16, с. 48-55];

- швидке створення нових документів із вже існуючих;

- скорочення часу пошуку потрібних документів;

- інформаційні технології без паперових носіїв покращують також процес придбання та використання знань. Вони є основою рішень, що забезпечують централізований обмін інформацією, дозволяючи отримувати тільки необхідну в даний момент інформацію з безлічі доступних джерел;

- системи електронного документообігу в майбутньому забезпечать створення абсолютно нової організаційної культури, значно полегшити роботу людей.

Таким чином, інформаційні технології дають змогу працівникам не тільки виконувати внутрішньовідомчі завдання, а й вирішувати більш широкий спектр завдань за допомогою спільних зусиль.

1.4. Життєвий цикл документа

Сьогодні на ринку інноваційних технологій представлено досить широкий спектр різних програмних рішень. Спробуємо їх проаналізувати Однією з важливих характеристик, особливо суттєвих для секретаря або діловода, який буде працювати в такій системі, є етап життєвого циклу документа, що автоматизує конкретна система.

Перш за все, розглянемо тлумачення поняття «життєвий цикл документа». Будь-який документ незалежно від його структури або змісту проходить низку стадій, які загалом називаються «життєвим циклом документа». Усі документи проходять через п’ять основних етапів життєвого циклу (деякі етапи можуть повторюватися, а деякі відвуваються тільки один раз). Виокремлюють такі основні етапи життєвого циклу документа:

1) створення документа;
2) рецензування та виправлення документа;
3) формальне або неформальне затвердження;
4) поширення або публікація документу для ширшої аудиторії;
5) виконання своєї основної функції і надходження до архіву;
6) за необхідності виклик з архіву та повторна архівація.

Варто зазначити, що поняття «життєвий цикл документа» відсутнє в підручниках з діловодства. Замість нього вживається термін «документообіг», який передбачає рух документів в організації з моменту їх створення або отримання до завершення виконання, відправлення [28, с. 47-53].

Виокремлюють такі етапи технологічного ланцюжка обробки документів:

- прийом і первинна обробка;
- попередній розгляд і розподіл;
- реєстрація документів;
- напрямок виконання та роботи документів;
- оформлення та засвідчення документів, їх відправка.

Окремо як завдання діловодства розглядаються питання контролю виконання та зберігання документів. Діловодство описує тільки певну частину етапів життєвого циклу документа, а саме, частину, у межах якої працює діловод [18].

Зазначимо, що в системах електронного документообігу з документами працюють всі співробітники, які потребують цього за специфікою своєї діяльності. Це керівники, яким потрібно прочитати лист і вирішити, що робити з ним далі, виконавці, що працюють над виконанням резолюції, готують відповідні листи. Загалом це може бути будь-який співробітник організації, якому потрібна інформація про документи, про їх зміст, призначеним виконавцям – про терміни виконання тощо.

Важливим є те, що у діловодстві прийнято виокремлювати групи документів – вхідні, вихідні, внутрішні, організаційно-розпорядчі. При роботі в системі електронного документообігу такий розподіл, як правило, залишається. Наведемо опис роботи в системі електронного документообігу з будь-яким типом документа, наприклад – вхідними. Документ створюється, рецензується, затверджується та поширюється (на цьому етапі документ потрапляє в організацію) [19].

Слід визнати, що на етапі створення (у цьому випадку отримання документа системою електронного документообігу із зовнішніх систем) необхідно *виконувати найбільш трудомістку частини* роботи, пов’язану із внесенням інформації про документ в систему електронного документообігу. Вважаємо необхідним виокремити частини роботи, які складаються з:

1) введення реквізитів документа;
2) введення образу документа.

Перейдемо до більш детального пояснення кожної з частин роботи. Перш за все, введення реквізитів документа схожий на процес складання реєстраційно-контрольної картки документа. Цю операцію виконують, як правило, всі системи електронного документообігу, але не всі можуть зберігати електронний образ документа. Далі документу присвоюється

реєстраційний номер, накладається резолюція, призначаються відповідальні за його виконання, визначається термін виконання. Виконує видачу резолюції її автор. Однак бувають ситуації, коли він не може (або не хоче) це робити. Тоді на плечі секретаря лягає робота, пов'язана із уведенням в систему резолюції, виконаної автором в традиційному – «паперовому» – форматі [39, с. 127-131].

Наступним етапом обробки документа є створення звіту про його виконання (позначки про виконання). Набагато зручніше, якщо можна зробити не просто позначку «виконано, дата», а мати змогу залишити коментар, зберегти файл, який буде підтвердженням, що документ дійсно виконаний.

Ця операція може бути розбита на дві частини:

- звіт самого виконавця;
- відмітка керівника про те, що результат його влаштовує.

Оскільки необхідно визначити, хто конкретно вносить інформацію про ці операції в систему, розглянемо всі чинники. Все залежить від того, як організована робота та хто її виконує. Варіанти можуть бути досить різноманітними: наприклад, виконавець, автор резолюції, діловод. При вивченні системи електронного документообігу ці операції потребують особливої уваги. Краще, коли самі виконавці вносять інформацію про виконання, оскільки такий підхід розвантажує службу діловодства. Крім того, додаткові можливості систем електронного документообігу (сортування документів на «виконані» та «невиконані», автоматична розсилка через систему повідомлень про наближення терміну виконання документа) дають змогу підняти виконавчу дисципліну серед співробітників [31].

Наступний етап «передача в архів» не означає, що документ стає непотрібним. Перш за все, необхідно організувати зберігання документа так, щоб його можна було легко знайти й отримати потрібну інформацію. На цьому етапі можуть знадобитись операції контролю виконання документів, а

також операції зі складання звітів за статистикою виконання документа і створювати власні звіти. [11].

Слід зауважити, що одним з етапів життєвого циклу документа є його узгодження, яке пов’язане з тим, коли документ переходить від автора і набуває чинності, з ним ознайомлюються, висловлюють свою думку і сперечаються зацікавлені користувачі. Якщо проаналізувати життєвий цикл документа з урахуванням кількості людей, які працюють з ним на кожній стадії, можна помітити, що найбільша кількість людей працює з документами на стадії узгодження.

Розглянемо детальніше процес узгодження, який передбачає одну з найбільш трудомістких стадій підготовки документа. Як правило, у ній бере участь декілька служб, і робота з документом кожної з них займає певний час. Відтак загальний час на узгодження документа протягом його виконання може стати досить тривалим через на певну кількість циклів узгодження. Крім того, процес можуть затягувати затримки при передачі документа від однієї служби до іншої. Виникають ситуації, коли документ випадково забувають в якій-небудь службі, у результаті чого процес просто зупиняється, і потрібно докладати зусиль, аби віднайти цей документ. Для вирішення цих проблем системи електронного документообігу пропонують можливість спільної підготовки та узгодження документів [34].

Варто зазначити, що ці процеси схожі, але реалізуються по-різному. При узгодженні документа, його зміст не змінюється. Учасники узгодження висловлюють свої зауваження («Згоден», «Не згоден», «Особлива думка») поза текстом документа. Підсумком є вихідний текст документа, а також набір думок і зауважень та згод сторін. Ініціатор узгодження сам вносить усі зміни в текст документу і, за необхідності, може направити документ на повторне узгодження.

Зауважимо, що проблема узгодження полягає в різноманітності маршрутів проходження документів і реакцій учасників цього процесу. Наприклад, узгодження може йти незалежно від порядку учасників, так і

згідно з ним. Таким чином, процес узгодження може добігати кінця незалежно від думок його учасників, так і припинятися в разі, якщо хтось висловить свою незгоду. Протягом розробці маршрутів узгодження документів у системі електронного документообігу Company Media, використовується спеціальний конструктор. Так, за допомогою конструктора користувач системи може вибирати параметри узгодження: паралельне, послідовне, комбіноване [39].

Вважаємо необхідним, з'ясувати яким чином відбувається паралельне узгодження. Перш за все, у роботі в СЕД, реального руху документа не відбувається. Він знаходиться в базі даних, тому експерти, включені у список погодження, одночасно отримують доступ до документа і можуть незалежно один від одного працювати з ним. Сигналом до початку роботи є отримання учасником узгодження повідомлення, яке надходить на електронну пошту. Воно містить стандартний текст, який інформує одержувача про те, що він призначений учасником процесу погоджень. Відкривши повідомлення, користувач за посиланням може перейти до самого документу [43].

З усього сказаного вище, можемо зробити висновок, що за умови послідовного узгодження, доступ до документа по черзі отримують експерти, включені у список погодження. Послідовне узгодження за необхідності можна ускладнити, зокрема: указати тривалість кожного етапу узгодження (зробити їх різними), прописати, що саме робити в разі, якщо термін узгодження пройшов, а віза «погоджено» не отримана. «Конструктор» пропонує такі варіанти дій: «нічого не робити»; «продовжити процес»; «одноразово повідомити ініціатора про прострочений строк узгодження і продовжити процес»; «призупинити погодження» та «налагодити процес періодичного повідомлення візування та ініціатора». Якщо один з учасників погодження не згоден, «конструктор» пропонує такі варіанти реакції: «повідомити ініціатора і продовжити процес»; «продовжити процес без відправки повідомлень»; «повідомити ініціатора і зупинити процес». Важливим є те, що від ініціатора узгодження потрібно набагато менше

зусиль і часу, щоб контролювати цей процес. Крім того, експлуатація системи дасть змогу витримати встановлені терміни підготовки документів.

Висновки до розділу 1

Аналіз наукових ресурсів і періодичних джерел з питань дослідження проблематики автоматизації документообігу в закладі вищої освіти дав змогу дійти таких висновків:

1. Сформульовано поняття автоматизованої інформаційної системи, як людино-машинної системи з автоматизованою технологією одержання результатної інформації, необхідної для інформаційного обслуговування фахівців та оптимізації процесу управління в різних сферах людської діяльності.

2. Розкрито значення автоматизованого документообігу як сучасної системи ведення ділової документації, в якій усі створювані й отримані документи обробляються з використанням спеціальної комунікаційної технології за допомогою персональних комп'ютерів, об'єднаних в єдину мережу.

3. Окреслено основні сфери, цілі та переваги застосування автоматизованого документообігу в закладі вищої освіти та виокремлено загальні принципи проектування систем обробки даних.

4. Узагальнено особливості процесів документообігу та їх взаємозв'язок між собою, а також ланки, що потребують автоматизації.

РОЗДІЛ 2. ОСОБЛИВОСТІ СТВОРЕННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ ДОКУМЕНТООБІГУ В ЗАКЛАДАХ ВИЩОЇ ОСВІТИ

У другому розділі викладено загальні відомості про інструменти, які буде використано для написання та впровадження інформаційної системи; описано визначні можливості й елементи функціонування системи; розглянуто основні елементи та бібліотеки, їх конфігурації і призначення.

2.1 Інструменти створення та впровадження інформаційної системи документообігу в закладах вищої освіти

2.1.1 Переваги застосування PHP у створенні інформаційних систем документообігу

Перш за все, перейдемо до тлумачення одного з інструмент для створення та впровадження інформаційної системи – PHP (Personal HyperText Processor). Зазначимо, що це мова програмування, що використовується на стороні WEB-сервера для динамічної генерації HTML-сторінок. PHP – одна з небагатьох мов програмування, створених спеціально для розробки веб-додатків. Вона включає в себе всі функції, необхідні саме для роботи на веб-сервері, і при цьому позбавлена надмірності, властивої багатьом її конкурентам.

Серед переваг цієї мови програмування варто зазначити, що PHP передбачає включення її команд у звичайні HTML-сторінки за допомогою спеціальних тегів, які примушують PHP-машину виконувати на сервері потрібні дії. Програмам, виконаним мовою PHP, не потрібні спеціальні CGI-директорії з особливими правами доступу. Більш того, на одній сторінці можна довільно чергувати «простий» HTML і PHP-код [12, с. 77-82].

Суттєвим чинником використання PHP є незалежність цього програмного коду від платформи. PHP прекрасно інтегрується в усі популярні веб-сервері: Apacce і IIS, Zens і Netscape Enterprise Server, працює під Windows і OS / 2, MacOS і практично всіма UNIX-подібними системами.

У результаті PHP працює практично у всіх хостерів, які дозволяють власні виконувані скрипти [13, с. 45].

Важливою особливістю PHP є її інтегрованість практично з усіма сучасними інтернет-технологіями. PHP підтримує більшість сучасних веб-протоколів: IMAP, FTP, POP, XML, SNMP і інші. PHP прекрасно працює з базами даних. Важко знайти СУБД, підтримка якої не була б реалізована в PHP. MySQL і MS SQL Server, PostgreSQL та Oracle, Sybase і Interbase. Сьогодні перелік баз даних, які підтримують PHP досить великий [12, с. 85-92].

Варто звернути увагу, що PHP включає в себе величезну кількість вбудованих функцій: обробки рядків і масивів, роботи з файловою системою і з HTTP, електронної поштою, датою і часом, кирилицею та іншими національними алфавітами. Програмування на PHP, дивує великою кількістю вбудованих функцій. Завдяки їм з'являється можливість до більшості алгоритмів, що вимагають у багатьох мовах написання достатньо великого за обсягом програмного коду; реалізуються на PHP однією командою (зокрема, викликом однієї функції) [50, с. 23-24] .

Слід зауважити, що PHP – це скрипт-мова (scripting language), яка вбудовується в HTML, інтерпретується та виконується на сервері. Продемонструємо це на прикладах. По-перше, відмінність PHP від JavaScript, полягає в тому, що PHP-скрипт виконується на сервері, а клієнту передається результат роботи, тоді як в JavaScript-код повністю встановлюється на клієнтську машину і виконується тільки там. По-друге, користувачі Internet Information Server помітять, що PHP дуже схожа на Active Server Pages (ASP), а прихильники Java скажуть, що PHP схожа на Java Server Pages (JSP). Усі три мови дають змогу розміщувати код, що виконується на Web-сервері, усередині HTML сторінок [35, с. 112-115].

Важливо підкреслити переваги PHP як інструменту для створення та впровадження інформаційної системи. Перш за все, PHP може зробити все, що виконується за допомогою CGI-програм, наприклад: обробляти дані з

форм, генерувати динамічні сторінки, одержувати і посилати куки (cookies). Крім цього, PHP включає підтримку багатьох баз даних (databases), що робить написання Web-додатків з використанням БД достатньо простим. Також PHP підтримує протоколи IMAP, SNMP, NNTP, POP3 і навіть http та має можливість працювати з сокетами (sockets) і спілкуватися по інших протоколах [36, с. 74].

Розробники Web-додатків розуміють, що web-сторінки – це не тільки текст і картинки. Гідний уваги сайт повинен підтримувати певний рівень інтерактивності з користувачем, а саме: пошук інформації, продаж продуктів, конференції і т.д. Традиційно все це реалізувалося CGI-скриптами, написаними на Perl. Однак CGI-скрипти дуже погано масштабуються. Кожен новий виклик CGI, вимагає від ядра породження нового процесу, а це займає процесорний час і витрачає оперативну пам’ять. PHP пропонує інший варіант – він працює як частина Web-сервера, і цим самим схожий на ASP від Microsoft [49, с. 34-35].

Важливою особливістю є те, що синтаксис PHP дуже схожий на синтаксис C або Perl. Люди, знайомі з програмуванням, дуже швидко зможуть почати писати програми мовою PHP, яка не має суворої типізації даних і не потребує виконання операцій з виділення, очищення пам’яті [22, с. 45]. Таким чином, ми розглянули один з інструментів створення і впровадження інформаційної системи – PHP. Крім того зауважимо, що PHP є мовою, що інтерпретується, і, внаслідок цього, не може зрівнятися за швидкістю з компільовані C. Однак протягом написання невеликих програм, що є характерною особливістю проектів на PHP, які складаються з багатьох невеликих сторінок із кодом, на заваді стають витрати на завантаження в пам’ять і виклик CGI-програми, написані на C. Така велика база готових модулів, як, наприклад, CPAN у Perl. Отже, ця проблема потребує додаткових розміркувань. PHP 5 передбачає спеціальний репозиторій PEAR, аналогічний CPAN, крім того у майбутньому планується написання достатньї кількості модулів для його наповнення.

2.1.2 Використання MySQL як інструменту обробки даних в інформаційній системі

На початку, з'ясуємо тлумачення одного з інструментів для створення та впровадження інформаційної системи. MySQL – це компактний багатопотоковий сервер баз даних, який характеризується високою швидкістю, стійкістю й легкістю використання. MySQL був розроблений компанією TcX для внутрішніх потреб, які полягали у швидкій обробці значних за обсягом баз даних. Компанія стверджує, що використовує MySQL з 1996 року на сервері з більше ніж 40 БД, які містять 10,000 таблиць, з яких понад 500 мають більше 7 мільйонів рядків [34, с. 27-31].

Зауважимо, що MySQL є ідеальним рішенням для малих і середніх додатків. Вихідні тексти сервера компілюються на безлічі платформ. Усе розмаїття можливостей сервера цього інструменту виявляється на Unix-серверах, де є підтримка багатопоточності, що дає значний приріст продуктивності. Сьогодні MySQL усе ще в стадії розробки, хоч версії 3.22 повністю працездатні. MySQL-сервер є безкоштовним для некомерційного використання. Використання інших інструментів потребує придбання ліцензії, вартість якої становить 190 EUR [37, с. 115].

Перейдемо до більш детального пояснення короткого переліку можливостей MySQL:

1. Підтримка необмеженої кількості користувачів, що одночасно працюють з базою даних.
2. Кількість рядків у таблицях може досягати 50 млн.
3. Швидке виконання команд. MySQL вважається одним з найшвидших сучасних серверів.
4. Проста й ефективна система безпеки.

MySQL дійсно дуже швидкий сервер, але для досягнення цього ефекту розроблювачам довелося поступитися деяким вимогам до реляційних СУБД [42, с. 145].

Слід підкреслити, що існують певні особливості MySQL, в якому відсутні підтримка вкладених запитів, типу SELECT FROM table1 WHERE id IN (SELECT id FROM table2). чи не реалізована підтримка транзакцій. Натомість пропонується використовувати LOCK або UNLOCK TABLE. Немає підтримки тригерів і збережених процедур [43, с. 111].

Отже, розглянувши специфіку MySQL, ми виокремили переваги та недоліки цього інструменту для створення та впровадження інформаційної системи. Зауважимо, що вони дають змогу досягти високої швидкодії, проте їх реалізація істотно знижує швидкість сервера. Таким чином, ці можливості не є критичними при створенні Web- додатків, що в поєднанні з високою швидкодією та низькою ціною дозволило набути велику популярність.

2.1.3 Переваги застосування функціональних можливостей JavaScript

Насамперед важливо зазначити, що мова JavaScript дуже популярна завдяки її присутності в будь-якому веб-браузері. Вона ні в чому не поступається іншим мовам, але при цьому підтримує багато сучасних уявлень про те, якою повинна бути мова програмування. Завдяки широкому поширенню є чимало досвідчених програмістів, які користуються JavaScript.

Характерною особливістю для динамічної мови JavaScript є низька типізованість з розширюваними об’єктами, які неформально оголошуються у разі необхідності. Функції в ньому є повноцінними об’єктами та зазвичай використовуються у вигляді анонімних замикань. Це робить JavaScript потужнішою мовою у порівнянні з іншими, часто вживаними для розробки веб-додатків. Теоретично наявність таких можливостей має підвищувати продуктивність програмістів [44, с. 56-58].

Зауважимо, що одним з основних недоліків JavaScript є Глобальний Об’єкт. Усі змінні верхнього рівня «звалюються» до Глобального Об’єкту і при застосуванні одночасно кількох модулів викликають неконтрольований хаос. Оскільки веб-додатки складаються з безлічі об’єктів, можливо,

створених різними організаціями, виникають побоювання, що програмування для JavaScript являє собою прогулянку замінованим полем, нашпигованим конфліктуючими між собою глобальними об’єктами. Однак це не так, тому що, насправді в JavaScript використовується система організації модулів CommonJS, а це означає, що локальні змінні певного модуля так і будуть локалізовані в ньому, навіть, якщо вони виглядають як глобальні. Таке чітке розмежування між модулями вирішує проблему Глобального Об’єкту [44, с. 138-147].

Слід зазначити, що використання єдиної мови програмування на сервері та на клієнті давно вже було мрією веб-розробників. Своїм корінням ця мрія йде в період становлення Java, коли аплети представлялися клієнтським інтерфейсом до написаним на Java серверним додаткам, а JavaScript спочатку виглядав як полегшена скриптова мова взаємодії з апплетами. Однак популярності набула не Java, а JavaScript, яка стала основною мовою з позиції клієнта-браузера. З появою JavaScript нарешті стало можливим реалізувати цю мрію, а саме: зробити JavaScript мовою, яка використовуватиметься по обидва боки веб-сервера – з боку клієнта й сервера.

Завдяки асинхронної орієнтованій архітектурі JavaScript демонструє таку високу продуктивність. До цього треба додати ще і стрімкість движка V8 JavaScript. У традиційній моделі додатків паралелізм забезпечується завдяки використанню блокуючого вводу або виводу та декількох потоків. Кожен потік повинен чекати завершення введення та виводу, перш ніж приступити до обробки наступного запиту.

Важливим фактором є те, що в JavaScript присутній єдиний потік виконання, без будь-якого контекстного перемикання або очікування введення та виведення. При будь-якому запиті введення та виведення задаються функції обробки, які згодом викликаються з циклу обробки подій, коли стануть доступні дані або станеться ще щось значуще. Модель циклу обробки подій і обробника подій – річ поширена, саме так виконуються

написані на JavaScript скрипти в браузері. Очікується, що програма швидко поверне управління циклу обробки, щоб можна було викликати наступне в черзі завдання. Райан Дав (в презентації «Cincode JavaScript») запитує, що відбувається при виконанні такого коду: result = query ('SELECT from db) [35, с. 47-53].

Зрозуміло, що програма в цій точці припиняється на час, поки шар доступу до бази даних відправляє запит базі, яка обчислює результат і повертає дані. Залежно від складності запиту його виконання може зайняти досить значний період час. Часовий недолік спричиняє припинення потоку, хоч у цей час може пройти інший запит, але якщо зайняті всі потоки, то запит буде просто відкинутий. Така ситуація викликає непотрібні витрати часу, Крім того, контекстне перемикання відбувається таким чином, що чим більше запущено потоків, тим більше часу процесор витрачає на збереження і відновлення їх стану. Відтак, стек кожного потоку займає місце в пам'яті, передусім, завдяки асинхронного орієнтованого введення та виведення JavaScript, яке усуває більшу частину цих накладних витрат, привносячи зовсім небагато власних [44, с. 17-22].

Зазначимо, що існує багато суперечок щодо реалізації паралелізму за допомогою потоків, які супроводжуються застереженнями, пов'язаних з високою собівартістю та значною кількістю помилок при роботі. Також виникають ненадійні примітиви синхронізації в Java, проектування паралельних програм виявляється складним та не виключена поява помилок. Причиною цієї складності є доступ до поділюваних змінним і різні стратегії запобігання блокувань і змагань між потоками. «Примітиви синхронізації в Java» – один із прикладів такої стратегії, і, очевидно, багато програмістів вважають, що користуватися ними важко.

Слід наголосити, для усунення складності, притаманної багатопотоковому паралелізму, створюються каркаси типу java.util.concurrent, але все одно деякі експерти вважають, що спроба усунути цю складність вирішує цю проблему. JavaScript закликає підходити до

паралелізму. Зворотні виклики з циклу обробки подій – це модель паралелізму, яка набагато простіша для розуміння і реалізації.

Зауважимо, щоб пояснити необхідність асинхронного введення та виводу, Райан Дав нагадує про відносність часу доступу до об'єктів. Доступ до об'єктів у пам'яті (близько наносекунда) проводиться швидше, ніж до об'єктів на диску (мілісекунди або секунди). Час доступу до зовнішніх об'єктів вимірюється незліченною кількістю тактових циклів і може тривати достатньо довгий проміжок часу. Відтак клієнт може не дочекатися завантаження сторінки протягом двох секунд і відкрити інше вікно браузера, замість потрібного [43, с. 87-93].

Важливим є те, що в JavaScript згаданий запит вище слід було б записати так:

```
query( SELECT * from db', function (result) {
//
});
```

Передусім, різниця в тому, що тепер результат запиту не повертається як значення функції, а передається функцією зворотного виклику, яка буде викликана пізніше. Таким чином, повернення в цикл обробки подій відбувається майже відразу і сервер може перейти до обслуговування інших запитів. Одним з таких запитів буде відповідь на запит, відправлена до базі даних, і тоді буде викликана функція зворотного виклику. Така модель швидкого повернення в цикл обробки подій підвищує ступінь використання ресурсів сервера. Ця особливість є значною перевагою для власника сервера, але ще більший виграш отримує користувач, перед яким швидше відкривається вміст сторінки [33, с. 28-30].

З усього сказаного вище можемо зробити висновок, що в наші дні веб-сторінки частіше збирають дані з десятків джерел. Кожному потрібно відправити запит і дочекатися відповіді на нього. Асинхронні запити дають змогу виконувати ці операції паралельно – відправити одразу всі запити і задати для кожного власний зворотний виклик, водночас, не чекаючи

відповіді, повернутися в цикл обробки подій. У разі надходження відповіді буде викликана відповідна функція. Таким чином, завдяки декільком потокам, дані можна зібрати набагато швидше, ніж за умови синхронного виконання запитів. У результаті цього користувач отримує швидке завантаження потрібної сторінки браузера.

2.1.4 Обробка конфігураційних файлів за допомогою Apache HTTP-сервера

Вважаємо необхідним детальніше розглянути ще один з інструментів створення та впровадження інформаційної системи – Apache.

По-перше, Apache підтримує операційні системи Linux, BSD, Mac OS, Microsoft Windows, Novell NetWare, BeOS та є кросплатформним ПО. Основними перевагами Apache вважаються надійність і гнучкість конфігурації [46, с. 77].

По-друге, ядро Apache включає в себе основні функціональні можливості, такі як обробка конфігураційних файлів, протокол HTTP і система завантаження модулів. Ядро (на відміну від модулів) повністю розробляється Apache Software Foundation, без участі сторонніх програмістів. Ядро Apache повністю написано на мові програмування C [43, с. 113-114].

Отже, ми проаналізували особливості Apache, дізналися, що його ядро має власну мову конфігураційних файлів, засновану на блоках директив. Практично всі параметри ядра можуть бути змінені через конфігураційні файли, навіть управління MPM. Велика частина модулів характеризується низкою власних параметрів. Частина модулів використовує у своїй роботі конфігураційні файли операційної системи (наприклад / etc / passwd і / etc / hosts).

2.1.5 Практичність використання AJAX у побудові інтерфейсів і web-додатків інформаційної системи

Варто зазначити, що наразі розробка WEB додатків прагне до розмежування клієнтської частини, цим і зумовлене широке використання шаблонів, таких як Smarty і XSLT. Сучасні проекти стають складнішими і розробнику важче поєднувати між собою різні технології, оскільки це потребує значних фінансових і часових витрат. Так, наприклад, усі стилі форматування виносяться в CSS або в XSL файли, HTML або XML дані зберігаються в інших розділах. І якщо ще 5-6 років тому практично скрізь можна було побачити поєднання цих компонентів в одному файлі, то зараз такий підхід стає рідкістю.

Перейдемо до роз'яснення AJAX (Asynchronous JavaScript and XML), який передбачає підхід до побудови призначених для користувача інтерфейсів веб-додатків. Його використання передбачає, що web-сторінка, не перезавантажується, а самостійно довантажує потрібні користувачу дані. AJAX – один з компонентів концепції DHTML.

Слід наголосити, що протягом розробки більш складних проектів виникає необхідність у структурованості і легкості читання коду. У підсумку кожному учаснику проекту достатньо знати тільки ті дані, з якими йому доведеться працювати. У такому випадку продуктивність групи і якість проекту підвищується в кілька разів. Сьогодні ця проблема успішно вирішується за допомогою використання шаблонів, однак це теж створює певні труднощі. Наприклад, щоб підключити Smarty, необхідно підключити програмний модуль обробки шаблонів, і чітко зв'язати зі структурою проекту. Однак фактичне здійснення цих операцій не завжди можливе і вимагає певних витрат [13, с. 115].

Отже, важливим є те, що трохи простіше використовувати зв'язки XML+XSL, оскільки вони надають більше можливостей, але таке рішення є лише альтернативним. Радикально новим є підхід, який дав би змогу об'єднати наявні переваги. Прикладом цього може стати оновлена JavaScript, яка характеризується всіма можливостями PHP або Perl, включаючи роботу з

графікою та базами даних та має набагато зручнішу розширюваність, практичність і крос-платформи.

2.2. Використання бібліотеки PHPWord

Передусім слід зазначити, що PHPWord – це бібліотека, написана мовою PHP, яка надає набір класів для написання та читання різних форматів файлів документів. Поточна версія PHPWord підтримує Microsoft Office Open XML (OOXML або OpenXML), формат відкритих документів OASIS для програм Office (OpenDocument або ODF) та формати Rich Text (RTF).

Крім того PHPWord – це проект з відкритим кодом, ліцензований відповідно до умов LGPL версії 3. PHPWord і спрямований на те, щоб бути високоякісним програмним продуктом, включивши безперервну інтеграцію та тестування пристроїв. Більше інформації щодо PHPWord представлено в документації розробників та документації API. Бібліотека PHPWord, яка вже майже рік знаходиться в стадії бета-тестування, надає можливість створення складних документів формату OOXML (*.docx) [48, с. 145-148].

Розглянемо основні можливості цієї бібліотеки. Для початку роботи достатньо розпакувати архів з бібліотекою в каталог зі створюваним вами документом PHP і довантажити основний клас бібліотеки, розташований в файлі PHPWord.php. Зі свого боку, створення документу починається з оголошення примірника класу PHPWord. Конструктор не вимагає передачі аргументів: $ Word = new PHPWord ();

Наступний крок передбачає створення назви і редагування розміру шрифту:

$ Word-> setDefaultFontName ('Times New Roman');

$ Word-> setDefaultFontSize (14);

У наведеній версії до застосування зазначених вище функцій шрифтом є Arial розміром 20. Тепер можна задати час створення документу, ім’я автора, тощо:

$ Meta = $ word-> getProperties ();

$ Meta-> setCreator ('Ім’я творця документа');

$ Meta-> setCompany ('Організація');

$ Meta-> setTitle ('Назва документа');

$ Meta-> setDescription ('Опис документа');

$ Meta-> setCategory ('Категорія документа');

$ Meta-> setLastModifiedBy ('Ім’я останнього редактора');

$ Meta-> setCreated (mktime (0, 0, 0, 5, 12, 2011)); // Дата і час створення документу;

$ Meta-> setModified (time ()); // Дата та час останньої зміни документу;

$ Meta-> setSubject ('Тема документу');

$ Meta-> setKeywords ('ключові, слова, документу');

Зазначимо, що в якості дати створення та зміни документу вказується поточний час, а інші властивості заповнюються порожніми значеннями. Якщо потрібно вказати конкретну дату створення або зміни, тоді використовується функція mktime або будь-яка інша функція, яка змінює час у стилі Unix.

Основним елементом документа Word є розділ. Розділ являє собою прямокутне поле, усередині якого розміщуються інші елементи сторінки: текст, зображення, таблиці і тощо.

Розділ може мати книжкову або альбомну орієнтацію, налаштування поля (margins), що налаштування кольору, кордонів розділу і їх товщину:

orientation - орієнтація сторінки;

null - книжкова орієнтація;

landscape - альбомна;

marginTop - розмір верхнього поля у внутрішніх одиницях;

marginRight - розмір правого поля у внутрішніх одиницях;

marginBottom - розмір нижнього поля у внутрішніх одиницях;

marginLeft - розмір лівого поля у внутрішніх одиницях;

borderTopSize - товщина верхнього кордону рамки у внутрішніх одиницях;

borderRightSize - товщина правого кордону рамки у внутрішніх одиницях;

borderBottomSize - товщина нижнього кордону рамки у внутрішніх одиницях;

borderLeftSize - товщина лівого кордону рамки у внутрішніх одиницях;

borderTopColor - колір верхнього кордону рамки в шістнадцятковому форматі;

borderRightColor - колір правого кордону рамки в шістнадцятковому форматі;

borderBottomColor - колір нижнього кордону рамки в шістнадцятковому форматі;

borderLeftColor - колір лівого кордону рамки в шістнадцятковому форматі.

Для створення розділу існує функція createSection. Серед внутрішніх одиниць використовуються друкарські твіпи. Якщо вам незвично вказувати розміри в твіпах, можна написати просту функцію, що перетворює міліметри в твіпи:

```
function m2t ($ millimeters) {
  return floor ($ millimeters * 56.7); // 1 твіп дорівнює 1/567 сантиметра
} // m2t
```

Зазначені в таблиці параметри можуть бути передані у вигляді масиву при створенні розділу:

```
$ SectionStyle = array ( 'orientation' => 'landscape',
            'MarginLeft' => m2t (15), // Ліве поле дорівнює 15 мм
            'MarginRight' => m2t (15),
            'MarginTop' => m2t (15),
            'BorderTopColor' => 'C0C0C0'
        );
$ Section = $ word-> createSection ($ sectionStyle);
$ Section = $ word-> createSection ();
```

```
$ SectionStyle = $ section-> getSettings ();
$ SectionStyle-> setLandscape (); // або setPortrait ()
$ SectionStyle-> setMarginLeft (m2t (15));
$ SectionStyle-> setMarginRight (m2t (15));
$ SectionStyle-> setBorderBottomSize (m2t (1));
$ SectionStyle-> setBorderTopColor ( 'C0C0C0');
```

Додавання блоку тексту. Під блоком тексту розуміється уривок тексту, який має однакове форматування (колір, розмір шрифту і т.д.). Для створення блоку тексту в обраному розділі використовується функція addText:

```
$ Section-> addText ($ text, [$ fontStyle [, $ paragraphStyle]]);
```

Тут $ text - додається текст, необов’язковий параметр $ fontStyle - ім’я певного текстового стилю, необов’язковий параметр $ paragraphStyle - ім’я певного стилю абзацу.

Зміна форматування тексту. Форматування тексту, як і форматування розділу, може здійснюватися протягом його створення:

```
$ FontStyle = array ( 'color' => 'FFFF00', 'size' => 18, 'bold' => true);
$ Section-> addText ( 'Привіт!', $ FontStyle); // Масив з параметрами
```

Висновки до розділу 2

Аналіз теоретичного матеріалу дав змогу зробити висновки щодо теоретичного підґрунтя і практичного застосування інформаційної системи, зокрема:

1. Проаналізовано й обрано інструментальні засоби розробки та проектування інформаційної системи. Проведено всебічний розгляд методів та інструментів, необхідних для створення інформаційної системи

2. Систематизовано переваги використання мови програмування PHP у побудові інформаційної системи документообігу для закладу вищої освіти (поліфункціональність, незалежність від платформи, гнучкість тощо).

3. Узагальнено особливості застосування інструменту MySQL для досягнення високої швидкодії доступу до даних та низької собівартості інформаційного продукту.

4. Розкрито основні характеристики мови програмування JavaScript та її пріорітетне значення для сучасних web-систем.

5. Простежено переваги застосування інструменту AJAX у побудові зручних інтерфейсів веб-додатків, що автоматично дають користувачу доступ до необхідної інформації і не потребують додаткового перезавантаження web-сторінок.

6. Розглянуто основні елементи, конфігурації та призначення бібліотеки PHPWord у створенні інформаційної системи.

РОЗДІЛ 3. РОЗРОБКА ТА ТЕСТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ ВЕДЕННЯ ДОКУМЕНТООБІГУ У ЗАКЛАДІ ВИЩОЇ ОСВІТИ

У третьому розділі описано і представлено розробку та впровадження інформаційної системи для введення документообігу у закладі вищої освіти. Описано поетапну розробку всіх елементів конфігурації та їхнє функціонування в межах цієї системи.

3.1. Особливості побудови інформаційної системи документообігу в закладах вищої освіти

Функції введення інформації до бази даних.

Функція saveStudents() приймає файл формату csv та отримує дані, ці дані вводяться до бази даних:

```
function saveStudents($file)
{
	global $link;
	$first = TRUE;
	$headers = array();
	if(($handle = fopen($file, "r")) !== FALSE)
		{
			while(($data = fgetcsv($handle, 0, ';', '"')) !== FALSE)
				{
					if($first)
						{
							$headers = $data;
							$count = count($data);
							$last = $count - 1;
							$first = FALSE;
							continue;
						}
					$sql = "INSERT INTO `students` (";
					for($i = 0;$i < $count;$i++)
						{
							$sql .= "`" . $headers[$i] . "`";
							if($headers[$i] == 'faculty')
								{
									$faculty = $i;
								}
```

```
                                        if($headers[$i] == 'stupin')
                                                {
                                                        $stupin = $i;
                                                }
                                        if($headers[$i] == 'special')
                                                {
                                                        $special = $i;
                                                }
                                        if($headers[$i]                 ==
'specialization')
                                                {
                                                        $specialization    =
$i;
                                                }
                                        if($i != $last)
                                                {
                                                        $sql .= ", ";
                                                }
                                }
                        $sql .= ") VALUES (";
                        for($i = 0;$i < $count;$i++)
                                {
                                        $data[$i] = trim($data[$i]);

                                        if($i == $faculty)
                                                {
                                                        $facultetm          =
$data[$i];
                                                        $data[$i]           =
getPole($data[$i],'facultet');
                                                        $facultet           =
$data[$i];
                                                }

                                        if($i == $stupin)
                                                {
                                                        $data[$i]           =
getPole($data[$i],'stupin');
                                                }

                                        if($i == $special)
                                                {

        if(trim($data[($i+2)])=='')
                                                                {
```

```
        $data[$i] = getPole($data[$i], 'special',$facultet);
                                                                }
                                                        else
                                                                {

        $data[($i+2)] = trim($data[($i+2)]);

        $data[$i] = getPole($data[$i] . ' (' . $data[($i + 2)] . ')', 'special',$facultet);
                                                                }
                                                        }

                                                if($i == $specialization)
                                                        {
                                                                if($data[$i]=='')
                                                                   {
                                                                       $facultet    =
0;
                                                                   }
                                                                $data[$i]           =
getPole($data[$i],'specialization',$facultet);
                                                        }

                                                $sql          .=          '"'"          .
mysqli_real_escape_string($link, $data[$i]) . '"';
                                                if($i != $last)
                                                        {
                                                                $sql .= ", ";
                                                        }
                                                }
                                        $sql .= ");";
                                        $query = mysqli_query($link, $sql);
                                        if(!$query)
                                                {
                                                        //echo $data[4] . PHP_EOL;
                                                        echo $sql;exit;
                                                }
                                }
                        fclose($handle);
                }

        }
```

Функція getPole() приймає змінну $name, в якій зберігаються дані отримані з csv файлу, та змінну $table з назвою таблиці, до якої буде введено дані:

```
function getPole($name,$table,$alt='')
{
	global $link;
	$sql = "SELECT * FROM `".$table."` WHERE `name` = '" . mysqli_real_escape_string($link,$name). "'";
	$query = mysqli_query($link, $sql);
	if($query && $query->num_rows > 0)
		{
			$row = mysqli_fetch_array($query);
			return ($row['id']);
		}
	if($alt=='')
		{
			$sql = "INSERT INTO `" . $table . "` (`id`, `name`) VALUES (NULL, '" . mysqli_real_escape_string($link, $name) . "');";
		}
	else
		{
			$sql = "INSERT INTO `" . $table . "` (`id`, `name`, `facultet`) VALUES (NULL, '" . mysqli_real_escape_string($link, $name) . "', '". $alt."');";
		}

	$query = mysqli_query($link, $sql);
	if($query)
		{
			return (mysqli_insert_id($link));
		}
	else
		{
			return (0);
		}
}
```

Функція myping() приймає змінну $conn, перевіряє та провидить підключення до бази даних:

```
function myping($conn)
{
	global $dbserver, $dbuser, $dbpass, $dbname;
```

```
        if(!@mysqli_ping($conn))
            {
                @mysqli_close($conn);
                $conn = mysqli_connect($dbserver, $dbuser, $dbpass,
$dbname);
                if(!$conn)
                    {
                        printf("Невозможно подключиться к базе
данных. Код ошибки: %s\n", mysqli_connect_error());
                        exit;
                    }
                mysqli_query($conn, "set character_set_server='utf8'");
                mysqli_query($conn, "set names 'utf8'");
            }
        return $conn;
    }
```

Функції генерації документів.

Функція toWord() приймає кілька змінних із бази даних та генерує документи в форматі docx на основі даних з бази даних і вивантаження документів:

```
    function toWord($Faculty, $Higher_education, $Specialty,$Specialization,
$FIO,$Birthday,$Place_of_birth,$Citizenship,$name_educational_institution,
$Year,$Family_status,$residence,$Privileges,$Day,$Month,$Year2,$Num,
$transfer_order,$quotas,$special_conditions,$Employment_history,
$Registration_card_number)
    {
      header('Content-Type: text/html; charset=utf-8');
      require 'vendor/autoload.php';
      $document=new PhpOffice\PhpWord\TemplateProcessor
    ($_SERVER['DOCUMENT_ROOT'].'/front/toWord/Template.docx');
      $document->setValue('Faculty', $Faculty);
      $document->setValue('Higher_education', $Higher_education);
      $document->setValue('Specialty', $Specialty);
      $document->setValue('Specialization', $Specialization);
      $document->setValue('FIO', $FIO);
      $document->setValue('Birthday', $Birthday);
      $document->setValue('Place_of_birth', $Place_of_birth);
      $document->setValue('Citizenship', $Citizenship);
                    $document->setValue('name_educational_institution',
$name_educational_institution);
```

```
            $document->setValue('Year', $Year);
            $document->setValue('Family_status', $Family_status);
            $document->setValue('residence', $residence);
            $document->setValue('Privileges', $Privileges);
            $document->setValue('Day', $Day);
            $document->setValue('Month', $Month);
            $document->setValue('Year2', $Year2);
            $document->setValue('Num', $Num);
            $document->setValue('transfer_order', $transfer_order);
            $document->setValue('quotas', $quotas);
            $document->setValue('special_conditions', $special_conditions);
            $document->setValue('Employment_history', $Employment_history);
                                    $document->setValue('Registration_card_number',
$Registration_card_number);
            $temp_file = tempnam(sys_get_temp_dir(), 'PHPWord');
            $document->saveAs($temp_file);
            header("Content-Disposition: attachment; filename=myFile.docx");
            readfile($temp_file);
            unlink($temp_file);
        }
```

Використані класи контролери.

Клас ControllerKdpuCards виконує пошук в базі даних та друк інформації про кожного студента:

```
        class ControllerKdpuCards extends Controller
        {
                public function index()
                        {

                                if(!$this->user->isLogged())
                                        {
                                                $this->response->redirect($this->url-
>link('common/login'));
                                                exit;
                                        }

                                $this->document->setTitle('КДПУ');
                                $data['footer']                                         =
$this->load->controller('common/footer');
                                $data['header']                                         =
$this->load->controller('common/header');
```

```
                $this->load->model('kdpu/cards');
                $data['faculty'] = $this->model_kdpu_cards->getData('facultet');
                $data['stupin'] = $this->model_kdpu_cards->getData('stupin');
        //              $data['special'] = $this->model_kdpu_cards->getData('special');
        //              $data['specialization'] = $this->model_kdpu_cards->getData('specialization');


        $this->response->setOutput($this->load->view('kdpu/cards', $data));
                }

        public function getStudents()
                {
                $this->load->model('kdpu/cards');
                $fac = $this->request->post['fac'];
                $stup = $this->request->post['stup'];
                $spec = $this->request->post['spec'];
                $speci = $this->request->post['speci'];

                echo json_encode($this->model_kdpu_cards->getStudents($fac, $stup, $spec, $speci));
                }

        public function getData()
                {
                $this->load->model('kdpu/cards');
                $fac = $this->request->post['fac'];
                $type = $this->request->post['type'];

                echo json_encode($this->model_kdpu_cards->getData($type, $fac));
                }

        public function printCard()
                {
                $this->load->model('kdpu/cards');
                $id = $this->request->get['id'];
                $stud = $this->model_kdpu_cards->getStudent($id);
                require $_SERVER['DOCUMENT_ROOT'] . '/front/toWord/toWord.php';
                $toz = explode(';',$stud['zarah']);
                $numz = explode(':',$toz[1]);
```

```
                $numz = trim($numz[1]);
                $dataz = explode(':',$toz[2]);
                $dataz = trim($dataz[1]);
                $zarah = explode('.', $dataz);
                $day = $zarah[0];
                $month = $this->date_ukr(intval($zarah[1]));
                $year = substr($zarah[2],2);
                if($stud['grom']=='Так')
                        {
                                $stud['grom'] = "України";
                        }
                else
                        {
                                $stud['grom'] = "";
                        }
                toWord($stud['faculty'], $stud['stupin'], $stud['special'], $stud['specialization'], $stud['fio'], $stud['birth'], '', $stud['grom'], '', '', '', '', '', $day, $month, $year, $numz, '', '', '', '', $stud['nalog']);
        }

        private function date_ukr($m)
        {
                $months = array(
                        '',
                        'січня',
                        'лютого',
                        'березня',
                        'квітня',
                        'травня',
                        'липня',
                        'червня',
                        'серпня',
                        'вересня',
                        'жовтня',
                        'листопада',
                        'грудня',
                );
                return $months[$m];
        }

}
```

Клас ControllerKdpuCards відповідає за доведення даних до контролеру:

class ControllerStartupRouter extends Controller

```
	{
		public function index()
			{
				// Route
				if(isset($this->request->get['route']) && $this->request->get['route'] != 'startup/router')
					{
						$route = $this->request->get['route'];
					}
				else
					{
						$route = $this->config->get('action_default');
					}

				// Sanitize the call
				$route = preg_replace('/[^a-zA-Z0-9_\/]/', '', (string) $route);

				// Trigger the pre events
				$result = $this->event->trigger('controller/' . $route . '/before', array(
					&$route,
					&$data,
				));

				if(!is_null($result))
					{
						return $result;
					}

				// We dont want to use the loader class as it would make an controller callable.
				$action = new Action($route);

				// Any output needs to be another Action object.
				$output = $action->execute($this->registry);

				// Trigger the post events
				$result = $this->event->trigger('controller/' . $route . '/after', array(
					&$route,
					&$data,
					&$output,
				));
```

```
                if(!is_null($result))
                    {
                        return $result;
                    }

                return $output;
            }
    }
```

Код сторінок веб додатків.

Сторінка реєстрації та авторизації:

```
<div id="content">
    <div class="container-fluid"><br/>
      <br/>
      <div class="row">
        <div class="col-sm-offset-4 col-sm-4">
          <div class="panel panel-default">
            <div class="panel-heading">
                <h1 class="panel-title"><i class="fa fa-lock"></i>
Вхід</h1>
            </div>
            <div class="panel-body">
                                <?php if($success) { ?>
                <div class="alert alert-success"><i class="fa fa-check-
circle"></i> <?php echo $success; ?>
                                <button type="button" class="close" data-
dismiss="alert">&times;</button>
                </div>
                                <?php } ?>
                                <?php if($error_warning) { ?>
                <div class="alert alert-danger"><i class="fa fa-
exclamation-circle"></i> <?php echo $error_warning; ?>
                                <button type="button" class="close" data-
dismiss="alert">&times;</button>
                </div>
                                <?php } ?>
                <form action="<?php echo $action; ?>" method="post"
enctype="multipart/form-data">
                <div class="form-group">
                  <label for="input-username">Логін</label>
                  <div class="input-group"><span class="input-group-
addon"><i class="fa fa-user"></i></span>
                <input type="text" name="username" value="<?
```

```
php echo $username; ?>" placeholder="Логін" id="input-username" class="form-control"/>
                                    </div>
                                </div>
                                <div class="form-group">
                                    <label for="input-password">Пароль</label>
                                     <div class="input-group"><span class="input-group-addon"><i class="fa fa-lock"></i></span>
                                                <input type="password" name="password" value="<?php echo $password; ?>" placeholder="Пароль" id="input-password" class="form-control"/>
                                    </div>
                                </div>
                                <div class="text-right">
                                        <button type="submit" class="btn btn-primary"><i class="fa fa-key"></i> Вхід</button>
                                </div>
                                                                    <?php if($redirect) { ?>
                                    <input type="hidden" name="redirect" value="<?php echo $redirect; ?>"/>
                                                                    <?php } ?>
                            </form>
                        </div>
                    </div>
                </div>
            </div>
        </div>
```

Сторінка користувача:

```
<?php echo $header; ?>
    <div class="container-fluid">
        <div class="row">
            <div class="col-sm-6">
                <form>
                    <div class="form-group">
                        <select class="form-control" id="faculty">
                            <option selected disabled value="0">Факультет</option>
                            <?php foreach($faculty as $fac) { ?>
                                <option value="<?php echo $fac['id']; ?>"><?php echo $fac['name']; ?></option>
                        <?php } ?>
                        </select>
                        <select class="form-control" id="stupin">
                            <option selected disabled>Ступінь</option>
```

```
                                        <?php foreach($stupin as $fac) { ?>
                                <option value="<?php echo $fac['id']; ?>"><?php echo
$fac['name']; ?></option>
                                        <?php } ?>
                        </select>
                        <select class="form-control" id="special">
                           <option selected disabled>Спеціальність</option>

                        </select>
                        <select class="form-control" id="specialization">
                           <option selected disabled>Спеціалізація</option>

                        </select>
                                    <button type="submit" class="btn btn-primary
forlist">Список</button>
                     </div>
                  </form>
               </div>
               <div class="col-sm-6">
                    <table class="table table-striped table-bordered table-hover table-
sm">
                     <thead>
                     <tr>
                        <th scope="col">Студент</th>
                        <th scope="col"></th>
                     </tr>
                     </thead>
                     <tbody class="studlist">

                     </tbody>
                  </table>
               </div>
            </div>
         </div>
      <?php echo $footer; ?>
```

3.2 Керівництво користувача інформаційної системи документообігу в закладах вищої освіти

Зазначимо, що для старту роботи користувач повинен авторизуватися за допомогою «Форми авторизації» (Рис. 3.1).

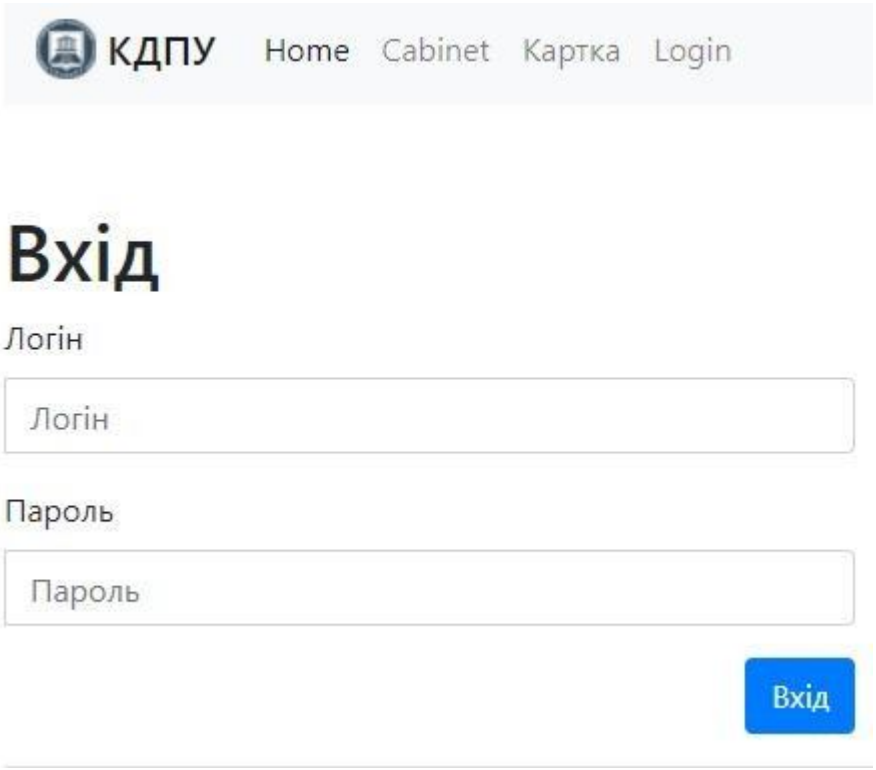

Рис. 3.1. «Форма авторизації»

Спочатку, перед користувачем відкриється вікно налаштувань, де користувач створить необхідний фільтр за списком студентів (Рис. 3.2).

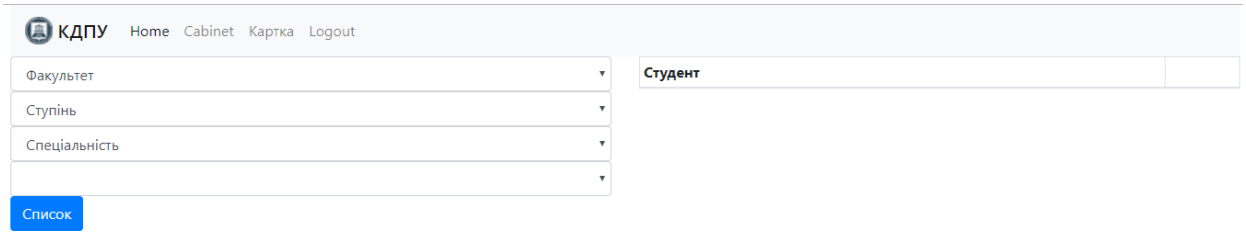

Рис. 3.2. Вікно файлового менеджеру

Наступним кроком буде заповнення користувачем всіх полів. Звертаємо увагу на те, що у звичайного користувача поле «Факультет» заповнюється автоматично. Після чого необхідно натиснути на кнопку «Список» (Рис. 3.3).

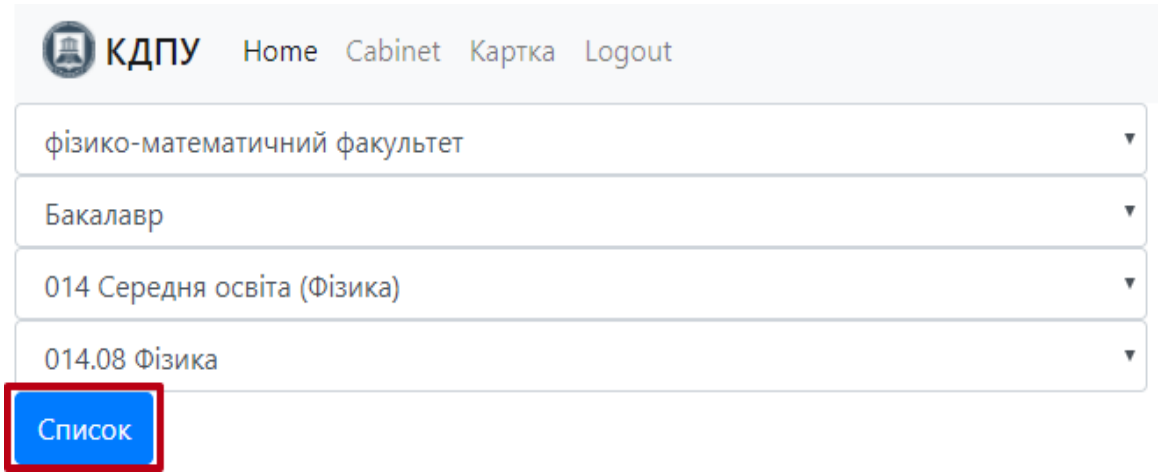

Рис. 3.3. Приклад заповнення форми

Після чого в правій частині сайту з’явиться список студентів, що відповідають фільтру (Рис. 3.4).

Студент	
Амангельдієва Балауса Альдамзарівна	Друк
Андріященко Ольга Сергіївна	Друк
Архипченко Ксенія Віталіївна	Друк
Бердімирадов Джошмират Таджибаєвич	Друк
Березовський Владислав Максимович	Друк
Богданова Олександра Андріївна	Друк
Буравіцин Ігор Валерійович	Друк
Василяка Наталя Вікторівна	Друк
Галпаков Мейліс	Друк
Гапотченко Олена Володимирівна	Друк
Гарригулов Башім	Друк
Гарсіа Гарсіа Данило Олегович	Друк
Грицай Яна Валеріївна	Друк
Дешевих Олена Валентинівна	Друк
Зеркаль Владислав Олексійович	Друк
Зімін Карім Русланович	Друк
Іноземцева Ганна Андріївна	Друк

Рис. 3.4. Список студентів

Потім, натискаємо на кнопку «Друк» (Рис. 3.5).

Рис. 3.5. Кнопка «Друк»

Після цього, на комп'ютер завантажується картка обраного студента (Рис. 3.6).

ДЕРЖАВНИЙ ВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД
«КРИВОРІЗЬКИЙ ДЕРЖАВНИЙ ПЕДАГОГІЧНИЙ УНІВЕРСИТЕТ»

Факультет, відділення фізико-математичний факультет

Ступінь вищої освіти Бакалавр
(бакалавр, спеціаліст, магістр)

Спеціальність 014 Середня освіта (Фізика)
(шифр і назва)

Спеціалізація014.08 Фізика
(назва)

Фото
картка
3х4 см

НАВЧАЛЬНА КАРТКА СТУДЕНТА

1. **Прізвище, ім'я, по батькові** Амангельдієва Балауса Альдамзарівна
2. **Дата народження** **14.12.1998**

Рис. 3.6. Картка студента

Висновки до розділу 3

У третьому розділі розроблено і протестовано програмне забезпечення для ведення документообігу в закладі вищої освіти. Узагальнення практичних результатів дослідження засвідчило досягнення мети, розв'язання поставлених завдань і дало змогу сформулювати такі висновки:

1. Описано поетапну розробку всіх елементів конфігурації та їх функціонування в межах створеної автоматизованої інформаційної системи документообігу.
2. Розроблено web-інтерфейс та інструкцію для користувача.
3. Проведено тестування програмного забезпечення на базі

деканатів Криворізького державного педагогічного університету.

4. Успішно впроваджено створену автоматизовану інформаційну систему для ведення документообігу у Криворізькому державному педагогічному університеті.

ВИСНОВКИ

Узагальнення теоретичних і практичних результатів роботи засвідчило досягнення мети та розв’язання поставлених завдань. Аналіз наукового підґрунтя та практичного застосування систематизованого теоретичного матеріалу з проблеми автоматизації документообігу в закладі вищої освіти дав змогу дійти таких висновків:

1. Сформульовано поняття автоматизованої інформаційної системи та розкрито значення автоматизованого документообігу.
2. Окреслено основні сфери, цілі, особливості, принципи та переваги застосування автоматизованого документообігу в закладі вищої освіти.
3. Виокремлено інструментальні засоби, їх основні елементи, конфігурації та призначення для розробки та проектування інформаційної системи (PHP, MySQL, JavaScript, Apache, AJAX, PHPWord).
4. Розроблено та впроваджено автоматизовану інформаційну систему для ведення документообігу в закладі вищої освіти.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Blackstone T. A Collaborative Applications Market Forecast and Analysis / Tomas Blackstone. – London : Education Review, 2000-2004. – 75 p.

2. Брусницин Ю. В. Экономическое обоснование инженерных разработок / Ю. В. Брусницин // Открытый урок. – 2011. − № 6. – С. 44–47.

3. Бутинець Ф. Ф. Інформаційні системи бухгалтерського обліку: підруч. для студ. ВНЗ спеціальності 7.050106 «Облік і аудит» / Ф. Ф. Бутинець, С. В. Івахненков, Т. В. Давидюк, Т. В. Шахрайчук; [за ред. проф. Ф. Ф. Бутинця. – 2-е вид., перероб. І доп. – Житомир: ПП «Рута», 2002. – 544 с.

4. Вендров А. М. Практикум по проектированию программного обеспечения экономических информационных систем / М. А. Вендров. – Москва: Финансы и статистика, 2002. – 192 с.

5. Вендров А. М. Проектирование программного обеспечения экономических информационных систем / М. А. Вендров. – Москва: Финансы и статистика, 2002. – 352 с.

6. Вендров А. М., CASE-технологии. Современные методы и средства проектирования информационных систем / М. А. Вендров. – Москва: Финансы и статистика, 2006. – 189 с.

7. Виейра Т. Программирование баз данных MS SQL Server 2005 / Т. Виейра. – Москва : Базовый курс , 2001. – 86 с.

8. Гаврилова А. Базы знаний интеллектуальных систем / А. Гаврилова. – Питер : Курс, 2000. – 384 с.

9. Гаджинский А. М. Основы логистики / М. А. Гаджинский. – Москва : Маркетинг, 2005. – 121 с.

10. Гольдштейн Я. Инновационный менеджмент / Я. Гольдштейн. – Таганрог: ТРТУ, 1996 . – 236 с.

11. Даниленко Л. И. Введение в системы баз данных / Л. И. Даниленко. – Київ: Логос, 2000. – 848 с. 55

12. Дари К. AJAX и PHP. Разработка динамических веб-сервер

приложений: учебник / К. Дари, Б. Бринзаре, Ф. Черчез-Тоза, М. Бусика. – Москва: Символ Плюс, 2006. – 354 с.

13. Джексон В. Проектирование реляционных баз данных для использования с микро / В. Джексон. – Москва: Финансы и статистика, 1991. – 365 с.

14. Диго М. Базы данных: проектирование и использование: Учебник / М. Диго. – Москва: Финансы и статистика, 2005. – 592 с.

15. Зеленков Ю. А. Введение в базы данных / А. Ю. Зеленков. – Санкт-Петербург: Центр Интернет ЯрГУ, 1997. – 167 с.

16. Зелковиц В. Принципы разработки программного обеспечения / В. Зелковиц, А. Шоу, Д. Гэннон. – Москва : Мир, 1982. – 386 с.

17. Калянов Г. Н. CASE. Структурный системный анализ (автоматизация и применение) / Н. Г. Кальянов. – Москва, Лори, 1996. – 214 с.

18. Карминский А. М. Информатизация бизнеса / А. М. Карминский, А. С. Карминский, В. П. Нестеров и др. – Москва: Финансы и статистика, 2004. – 624 с.

19. Карпова Т. С. Информационные системы: учебник для вузов / С. Т. Карпова. – Санкт-Петербург: «Питер», 2005 г. – 656 с.

20. Карпова, Т. С. Базы данных: модели, разработка, реализация / С. Т. Карпова. – Санкт-Петербург : Питер, 2011. – 87 с.

21. Крейн Д. AJAX в действии / Д. Крейн, Э. Паскарелло, Д. Джеймс. – Москва : Вильямс, 2006. – 490 с.

22. Крис Д. Введение в базы данных / Д. Крис. – Киев : Диалектика, 1998. – 68 с.

23. Кузнецов С. Д. Введение в СУБД / Д. С. Кузнецов. – Киев : Луч, 1996. – 67 с.

24. Кузнецов С. Д. Доступ к базам данных с использованием технологии WWW / Д. С. Кузнецов. – Киев : Луч, 1998. – 147 с.

25. Курочкина В. М. Язык компьютера / М. В. Курочкина. – Москва :

Мир, 2004. – 175 с.

26. Курочкина В. М. Язык компьютера. / М. В. Курочкина. – Москва: Мир, 2000. – 240 с.

27. Ловцов Д. А. Информационные системы в профессиональной деятельности : учеб.- метод. комплекс / Д. А. Ловцов, А. В. Зайцев, Е. С. Бурмистрова. – Москва : РАП, 2008. – 14 с.

28. Мазуркевич Р. настольная книга программиста, PHP / Р. Мазуркевич, Д. Еловая. – Москва : Новое знание, 2004. – 132 с.

29. Маклаков С. В. BPwin и ERwin. CASE – средства разработки информационных систем / В. С. Маклаков. – Москва : Диалог-МИФИ, 2000. – 257 с.

30. Маклаков С. В. Создание информационных систем с ALLFusion Modeling Suite (Практикум по BPWin и ERwin, приложение) / В. С. Маклаков. – Москва : ДИАЛОГ-МИФИ, 2003. – 432 с.

31. Мика С. Практическое руководство по программированию / С. Мика, П. Хит, Н. Рашби. – Москва : Связь, 1996. – 168 с.

32. Орлов П. І. Інформаційні системи та технології в управлінні, освіті, бібліотечній справі / П. І. Орлов, О. М. Луганський. — Харків : Вид. «Прометей-Прес», 2002. – 292 с.

33. Пасмор Ю. Інформаційне забезпеченння: соціально-комунікативний аспект : монографія / Ю.Пасмор. – Харків : Юрайт, 2013. – 272 с.

34. Пратт Т. Языки программирования: разработка и реализация / Т. Пратт, М. Зелковиц. – Санкт-Петербург : Питер, 2002. – 89 с.

35. Раден М. Данные, данные и только данные / М. Раден // ComputerWeek. – 2004. – № 8. – С. 28.

36. Семионов Ю. Разработка программного обеспечения / Ю. Семионов. – Москва: «Питер», 2004 . – 592 с. 57

37. Симионов Ф. Информационный менеджмент / Ф. Симионов, В. Боромотов . – Ростов на Дону : Феникс, 2006. –250 с.

38. Смирнова А. Проектирование и использование баз данных: учебник / А. Смирнова. – Москва : Финансы и статистика, 2005 . – 191 с.

39. Смирнова Д. Реляционные базы данных: практические приемы оптимальных решений / Д. Смирнова. – Санкт-Петербург : Инфо, 2005. – 400 с.

40. Смирнова И. П. Начала Web-дизайна, BHV / П. И. Смирнова. – Киев : Луч, 2003. – 128 с.

41. Смирнова Н. Проектирование экономических информационных систем : учебник / Н. Смирнова, А. Сорокин, Ю. Тельнов. – Москва: Финансы и статистика, 2002. – 512 с.

42. Терещенко Л. О. Інформаційні системи і технології обліку : навч. посіб. / Л. О. Терещенко. – Київ: КНЕУ, 2005. – 187 с.

43. Трубилина Т. Автоматизированные информационные технологии в экономике: учебник / Т. Трубилин, В. Лойко. – Москва : Финансы и статистика, 2000. – 416 с.

44. Федорова, Д. Э. CASE-технологии / Д. Федорова, Ю. Сесенов. – Москва : Горячая линия Телеком, 2005. – 168 с.

45. Флэнаган Д. JavaScript. Подробное руководство: учебник / Д. Флэнаган. – Москва: Символ Плюс, 2008. – 249 с.

46. Фокс Дж. Программное обеспечение и его разработка / Д. Фокс. – Москва : Мир, 1985. – 368 с.

47. Черемных С. В. Моделирование и анализ систем. IDEF-технологии: практикум / С. В. Черемных, И. О. Семенов. – Москва : Финансы и статистика, 2006. – 254 с.

48. Чубукова Г. Основы правовой информатики / Г. Чубукова, Д. Элькин. – Киев : Диалог-МИФИ, 2004. – 148 с.

49. Шапошников И. В., Самоучитель HTML4 / В. И. Шапошников. – Санкт-Петербург : БХВ-Петербург, 2001. – 354 с.

50. Шилдт Г. О. Полное руководство С 4.0 / О. Г. Шилдт. – Киев : Луч, 2001. – 54 с.