

**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СУМСЬКИЙ НАЦІОНАЛЬНИЙ АГРАРНИЙ УНІВЕРСИТЕТ
ФАКУЛЬТЕТ ЕКОНОМІКИ І МЕНЕДЖМЕНТУ**

Кібернетики та інформатики

КВАЛІФІКАЦІЙНА РОБОТА

освітній ступінь - «Бакалавр»

**на тему: Розробка інформаційної підсистеми обліку
кадрового забезпечення інформаційної системи коледжу ВСП
«ГАФК СНАУ»**

Виконав:

студент спеціальності

126 «Інформаційні системи та технології»

ОП «Інформаційні системи та технології»

Басанець Данило Віталійович

Керівник:

Толбатов Андрій Володимирович

Кандидат технічних наук, доцент

Рецензент:

Гриценко Костянтин Григорович

доцент кафедри економічної кібернетики. кандидат технічних
наук, Сумський державний університет

Суми – 2023

СУМСЬКИЙ НАЦІОНАЛЬНИЙ АГРАРНИЙ УНІВЕРСИТЕТ

Факультет	Економіки і менеджменту
Кафедра	Кібернетики та інформатики
Освітній ступінь	«Бакалавр»
Спеціальність	126 «Інформаційні системи та технології» ОП «Інформаційні системи та технології»

Затверджую:
Світлана АГАДЖАНОВА
Завідувач
кафедри
« ____ » « ____ » ____ р.

ЗАВДАННЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ *Басанцю Данилу Віталійовичу*

1.Тема роботи: Розробка інформаційної підсистеми обліку кадрового забезпечення інформаційної системи коледжу ВСП «ГАФК СНАУ»

2. База практичного дослідження: Відокремлений структурний підрозділ «Глухівський агротехнічний фаховий коледж Сумського НАУ»

керівник роботи Толбатов Андрій Володимирович, к.т.н., доцент

затверджена наказом по університету від « ____ » ____ 2023 р. № ____ -н

3. Строк подання студентом закінченого проекту (роботи)

«23» червня 2023 року

4. Вихідні дані до проекту (роботи):

Результати аналітичних досліджень зарубіжних та вітчизняних вчених, дані про кадровий відділ та поточний стан ВСП «Глухівський агротехнічний фаховий коледж Сумського НАУ», інтернет-джерела, нормативні документи за чинним законодавством.

5. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити):

- Виявлення недоліків підприємства та визначення задач;
- провести аналіз наявних рішень для вирішення проблеми;
- розробити технічне завдання;
- розробити програмний продукт відповідно до поставлених задач;
- розробити інструкцію для користувача;

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту(роботи)	Строк виконання етапів проекту	Примітка
1	Визначення об'єкту, предмету дослідження, формулювання мети та задач кваліфікаційної роботи, складання плану	вересень-жовтень 2022 р.	виконано
2	Підбір та вивчення літературних джерел та технічної документації	січень 2023 р.	виконано
3	Розробка технічного завдання	лютий 2023 р.	виконано
4	Розробка веб-додатку	березень-квітень 2023 р.	виконано
5	Створення інструкції користування веб-додатком для користувачів	до 10.05.2023	виконано
6	Оформлення пояснювальної записки	до 30.05.2023	виконано
7	Оформлення презентації	до 01.06.2023	виконано
8	Опрацювання зауважень керівника, перевірка на автентичність	02.06.2023	виконано
9	Підготовка доповіді та її попередній захист	06.06.2023	виконано
10	Перевірка з академічної доброчесності кваліфікаційної роботи	09.06.2023	виконано
11	Представлення кваліфікаційної роботи на кафедрі, деканат	09.06.2023	виконано
12	Захист кваліфікаційної роботи	28.06.2023	виконано

Студент

(підпис)

Данило БАСАНЕЦЬ

Керівник роботи

(підпис)

Андрій ТОЛБАТОВ

Перевірку на автентичність проведено

(підпис)

Надія БАРАНИК

Дипломна робота допущена до захисту

(підпис)

Маргарита ЛИШЕНКО

АНОТАЦІЯ

Басанець Д.В. Розробка інформаційної підсистеми обліку кадрового забезпечення інформаційної системи коледжу ВСП «Глухівський агротехнічний фаховий коледж Сумського НАУ».

Кваліфікаційна робота зі спеціальності 126 «Інформаційні системи та технології» ОП Інформаційні системи та технології, СНАУ, Суми-2023р. – Рукопис.

Об'єктом дослідження являється ВСП «Глухівський агротехнічний фаховий коледж Сумського НАУ», зокрема його кадровий відділ

Метою є розробка веб-додатку «HRM system GATI» яке має забезпечити ефективне управління персоналом шляхом автоматизації основних функцій кадрового обліку, управління кадровими документами, ведення бази даних співробітників, а також покращити комунікацію між кадровим відділом та співробітниками організації.

В роботі проведено аналіз останніх досліджень і публікацій, аналіз аналогічних рішень, проектування системи.

На основі проведеного аналізу було запропоноване рішення у формі створення веб-додатку. Цей додаток передбачає роботу з електронними картками працівників та файлове сховище для збереження даних

Результатом проведеної роботи є створений веб-додаток для роботи кадрового відділу ВСП «ГАФК СНАУ».

Пояснювальна записка складається зі вступу, 3 розділів, висновків, списку використаних джерел із 16 найменувань, додатків. Загальний обсяг роботи – _ сторінки, у тому числі 49 сторінок основного тексту, 2 сторінки списку використаних джерел, 41 сторінок додатків.

Ключові слова: Інформаційна система, інформаційна технологія, веб-додаток, управління персоналом.

SUMMARY

Basanets D.V. Development of an information subsystem for the personnel accounting of the information system of the Hlukhiv Agrotechnical Professional College of Sumy NAU.

Qualification work in the specialty 126 "Information Systems and Technologies" in the field of Information Systems and Technologies, Sumy NAU, Sumy-2023 - Manuscript.

The object of research is the Hlukhiv Agrotechnical Professional College of Sumy NAU, specifically its personnel department.

The aim is to develop the web application "HRM system HAPC" which aims to provide effective personnel management through the automation of key functions of personnel accounting, management of personnel documents, maintenance of an employee database, as well as improve communication between the personnel department and organization employees.

The work includes an analysis of recent research and publications, analysis of similar solutions, and system design.

Based on the conducted analysis, a solution in the form of a web application was proposed. This application involves working with electronic employee cards and a file repository for data storage.

The result of the conducted work is a created web application for the work of the personnel department of the Hlukhiv Agrotechnical Professional College of Sumy NAU.

The explanatory note consists of an introduction, 3 chapters, conclusions, a list of used sources with 16 titles, and appendices. The total volume of work is _ pages, including 49 pages of main text, 2 page of the list of used sources, and 41 pages of appendices.

Key words: Information system, information technology, web application, human resources management.

ЗМІСТ

ВСТУП.....	7
РОЗДІЛ 1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ	8
1.1 Огляд останніх досліджень і публікацій.....	9
1.2 Аналіз продуктів-аналогів.....	9
1.3 Вибір засобів реалізації	11
1.4 Постановка задачі.....	16
РОЗДІЛ 2. МОДЕЛЮВАННЯ ТА ПРОЕКТУВАННЯ ІНФОРМАЦІЙНОЇ ПІДСИСТЕМИ ОБЛІКУ КАДРОВОГО ЗАБЕЗПЕЧЕННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ КОЛЕДЖУ ВСП «ГАФК СНАУ»	18
2.1 Моделювання розроблюваної підсистеми.....	18
2.2 Проектування інформаційної підсистеми.....	20
2.3 Проектування моделі бази даних	24
РОЗДІЛ 3. РОЗРОБКА ІНФОРМАЦІЙНОЇ ПІДСИСТЕМИ ОБЛІКУ КАДРОВОГО ЗАБЕЗПЕЧЕННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ КОЛЕДЖУ ВСП «ГАФК СНАУ»	27
3.1 Архітектура програмного забезпечення	27
3.2 Програмна реалізація	28
3.3 Використання інформаційної системи.....	39
ВИСНОВКИ	48
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	50
ДОДАТОК А. ТЕХНІЧНЕ ЗАВДАННЯ	52
ДОДАТОК Б. ПЛАНУВАННЯ РОБІТ	61
ДОДАТОК В. ДОВІДКА ПРО ВИКОРИСТАННЯ РЕЗУЛЬТАТІВ ДИПЛОМНОЇ РОБОТИ.....	68
ДОДАТОК Г. КОПІЇ ПУБЛІКАЦІЙ.....	69
ДОДАТОК Д. ЛІСТИНГ ФАЙЛІВ ПРОГРАМНОГО КОДУ	76

ВСТУП

Розробка програмного забезпечення (далі - ПЗ) є одним з найбільш динамічних та перспективних напрямів у сучасній індустрії інформаційних технологій. ПЗ використовується практично у всіх сферах діяльності, починаючи від банків та фінансових установ та закінчуючи автомобільним виробництвом та медичними установами.

Розробка ПЗ є важливою галуззю сучасної економіки та відіграє ключову роль. Розробка ПЗ дозволяє скоротити витрати часу на виконання рутинних завдань, покращити якість та ефективність роботи, оптимізувати бізнес-процеси, підвищити безпеку даних, покращити взаємодію між різними підрозділами організації та навіть створити нові можливості для бізнесу. Крім того, автоматизація роботи користувачів через розроблене ПЗ сприяє підвищенню ефективності та зручності роботи. В даний час існує безліч компаній, що займаються розробкою ПЗ, та фахівців, які можуть створити високоякісне та ефективне ПЗ.

Розробка програмного забезпечення (ПЗ) включає складні та тривалі етапи, включаючи аналіз вимог, проектування, реалізацію, тестування і впровадження. Кожен з цих етапів вимагає від розробників спеціалізованих знань і навичок у програмуванні, архітектурі системи та тестуванні, з метою створення ефективного програмного продукту. Остаточним результатом розробки ПЗ є створення ефективного програмного продукту, який задовольняє потреби бізнесу та користувачів.

Метою даного дослідження є розробка веб-додатку який має забезпечити ефективне управління персоналом шляхом автоматизації основних функцій кадрового обліку, управління кадровими документами, ведення бази даних співробітників, а також покращити комунікацію між кадровим відділом та співробітниками організації.

Таким чином розробка ПЗ відіграє важливу роль у покращенні роботи організації, забезпечує оптимізацію процесів, покращує обмін інформацією між відділами та підвищення результативності компанії в цілому.

РОЗДІЛ 1

АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Огляд останніх досліджень і публікацій

Відповідно до мети теми та задач проєкту було виділено такі основні теми, що потребують детального вивчення шляхом розгляду літературних джерел: розробка зручного і сучасного дизайну веб-сайту, розробка логіки для веб-сайту, проектування бази даних.

Книга "HTML5, CSS3 та JavaScript. Вичерпний посібник" автора Дженніфер Роббінс – це практичний посібник для початківців та досвідчених веб-розробників, які хочуть вивчити сучасні технології веб-розробки на основі HTML5, CSS3 та JavaScript. Книга включає всі основні елементи мови HTML5, нові можливості, такі як семантичні теги і теги мультимедіа. Розглядаються можливості мови CSS3, включаючи трансформації, переходи, анімації та використання веб-шрифтів, але основна частина містить матеріал з мови JavaScript, включаючи основи мови, роботу з об'єктами та подіями, AJAX та JSON, використання бібліотек jQuery та Node.js.

Книга "Веб-дизайн, або Не змушуйте мене думати" автора Стіва Круга – це посібник з веб-дизайну, який допоможе будь-якому веб-розробнику-початківцю створювати зручні та зрозумілі інтерфейси для користувачів. Стів звертає увагу на те, що веб-сайти повинні бути легкими для розуміння та використання, а не складними та заплутаними. Автор пропонує безліч порад та прикладів, які допоможуть створити простий та зрозумілий інтерфейс для будь-якого сайту. Стів також говорить про те, як створити ефективний контент для веб-сайту, як використовувати колір та типографію, як створити прості та легко зрозумілі навігаційні системи та як тестувати та покращувати веб-сайти.

"NoSQL Distilled" - це книга, написана Мартіном Фаулером та Прамодкумаром Дж. Садалаге, яка є посібником з баз даних NoSQL. У книзі розглядаються основи баз даних NoSQL, включаючи їхню історію та мотивацію для використання. Автори надають огляд різних типів NoSQL баз даних, таких як ключ-значення, документо-

орієнтовані, колонкові та графові бази даних. Крім того, вони розглядають, як вибрати тип бази даних, який найкраще підходить для конкретного завдання, розглядаються такі NoSQL бази як: MongoDB, Cassandra, HBase, CouchDB та Riak. У книзі розглянуто основні функції та можливості кожної бази даних, а також розглядає сценарії використання.

1.2 Аналіз продуктів-аналогів

Існує багато сервісів аналогів у сфері автоматизації кадрового відділу, тому перед початком розробки потрібно провести аналіз існуючих продуктів аналогів. Аналіз існуючих додатків може допомогти визначити переваги та недоліки кожного продукту, визначити завдання для власного продукту, а також зрозуміти, що можна зробити для покращення функціональності.

Для порівняння було обрано дві HRM системи, перша HURMA System [12] від українського виробника, друга Peopleforce [13], одна з дешевших, що є на ринку. Ці два додатки схожі за будовою, дизайном і мають типовий вигляд для веб-сайту з базою даних. На основній сторінці можемо побачити меню навігації, воно в обох веб-сайтах знаходиться вертикально зліва, у іншій частині розміщується сама база даних на функціональні кнопки, ці додатки мають світлу тему оформлення, деякі модулі виділені сірим, функціональні кнопки мають синє та помаранчеве забарвлення у HURMA System, а у Peopleforce кнопки виділені зеленим забарвленням, або зеленою обводкою.

Основні критерії для оцінки продуктів аналогів:

1. Цінова політика
2. Підтримувані мови інтерфейсу
3. Переваги
4. Недоліки
5. Вимоги до технічних характеристик обладнання

Якщо ПП працює як веб-додаток то буде обрано за технічні характеристики браузер Chrome [1]

Таблиця 1.1 – Аналіз сайтів та їх критерій

Параметри	HURMA System	Peopleforce
Ціна за 1 працівника на місяць	77,61 грн	55,11 грн
Підтримувані мови	Українська, англійська, іспанська, російська	Українська, польська, німецька, іспанська, російська
Переваги	• Вбудована інтеграція з сервісами google, Telegram, Work.ua, LinkedIn, Viber. • Календар подій • Особистий профіль співробітника компанії • Чат для співробітників компанії • Зручний UX/UI • Швидка технічна підтримка	• забезпечує доступ до детальної аналітики, що дозволяє компаніям аналізувати дані • Один з самих дешевих продуктів для українського ринку
Недоліки	• Нестабільність та помилки • Ціна за користувача • Не гнучка система прав доступу для користувачів	• Не всі функції доступні для всіх країн • Малий вибір інтеграцій з сервісами • Обмеження обробки даних: система може обробляти обмежену кількість даних, що може стати проблемою для великих компаній
Вимоги до технічних характеристик обладнання	Процесор Core i5 2-го покоління (від 2 ГГц), Core i5 3-го чи 4-го покоління або аналогічні Windows 10 та новіші версії.	Процесор Core i5 2-го покоління (від 2 ГГц), Core i5 3-го чи 4-го покоління або аналогічні Windows 10 та новіші версії

Джерело: запропоновано автором на основі даних HURMA System [12] та Peopleforce [13]

Розрахуємо приблизну вартість на місяць ПЗ для підприємства ВСП «Глухівський агротехнічних фаховий коледж СНАУ» з 400 працівників.

HURMA System – 31044 грн в місяць

Peopleforce – 22044 грн в місяць

Отже, проаналізувавши ПП для кадрового відділу було виявлено обмеження і недоліки, а також і переваги даного ПЗ. Розробка власного ПП забезпечить більш точну відповідність потребам підприємства. Власний ПП повинен містити лише потрібний функціонал, так як у більшості вже готових продуктів йде переплата за не

потрібні функції для підприємства, місячна плата повинна бути меншою за продукти аналогів, а отже така розробка може бути корисною інвестицією для підприємства.

1.3 Вибір засобів реалізації

Створення веб-сайту - це завдання, яке вимагає знань з різних галузей, включаючи програмування. Сьогодні на ринку існує велика кількість мов програмування, які можуть бути використані для розробки веб-сайтів. Кожна з них має свої переваги та недоліки. Не існує універсальної мови програмування, яка підходить для всіх проектів. Вибір мови програмування залежить від багатьох факторів, таких як тип проекту, технічні вимоги, розмір проекту, досвід програміста та багато іншого.

Мова програмування — це штучна мова, створена для передачі команд машинам, зокрема комп'ютерам. Мови програмування використовуються для створення програм, які контролюють поведінку машин, та для запису алгоритмів [2].

HTML та CSS - це дві базові мови, які використовуються для створення веб-сторінок.

HTML (Hypertext Markup Language) - це мова розмітки, що використовується для створення структури веб-сторінок. За допомогою HTML можна визначити заголовки, параграфи, списки, таблиці, зображення та інші елементи сторінки.

CSS (Cascading Style Sheets) - це мова стилів, яка використовується для оформлення та стилізації веб-сторінок. За допомогою CSS можна встановити кольори, шрифти, розміри, відступи, фони та інші стилізаційні параметри для елементів HTML. CSS дозволяє окремо відокремити візуальний вигляд сторінки від її структури, що забезпечує більш гнучке та швидке оновлення вигляду сторінки [10].

Разом HTML та CSS допомагають розробникам створювати красиві та функціональні веб-сторінки, що допомагає поліпшити користувацький досвід та ефективність веб-додатків [9].

Для зручності написання CSS коду буде використано SCSS (Sass) препроцесор. SCSS (Sass) - це розширена версія CSS, яка дозволяє використовувати змінні, міксіни,

функції та інші програмні конструкції для більш зручного та швидкого написання стилів. SCSS дозволяє написати складні стилі один раз та використовувати їх у всіх необхідних місцях. Це допомагає знизити час розробки та спростити процес зміни стилів.

SCSS може бути компільований до звичайного CSS для використання в браузері, тому що браузери не можуть прямо розуміти SCSS [8].

Загалом, SCSS - це потужний інструмент, який дозволяє писати більш ефективний та швидкий код для стилізації веб-сторінок.

Найпопулярнішою мовою логіки серед українських розробників є JavaScript (рисунок 1.1) [3].

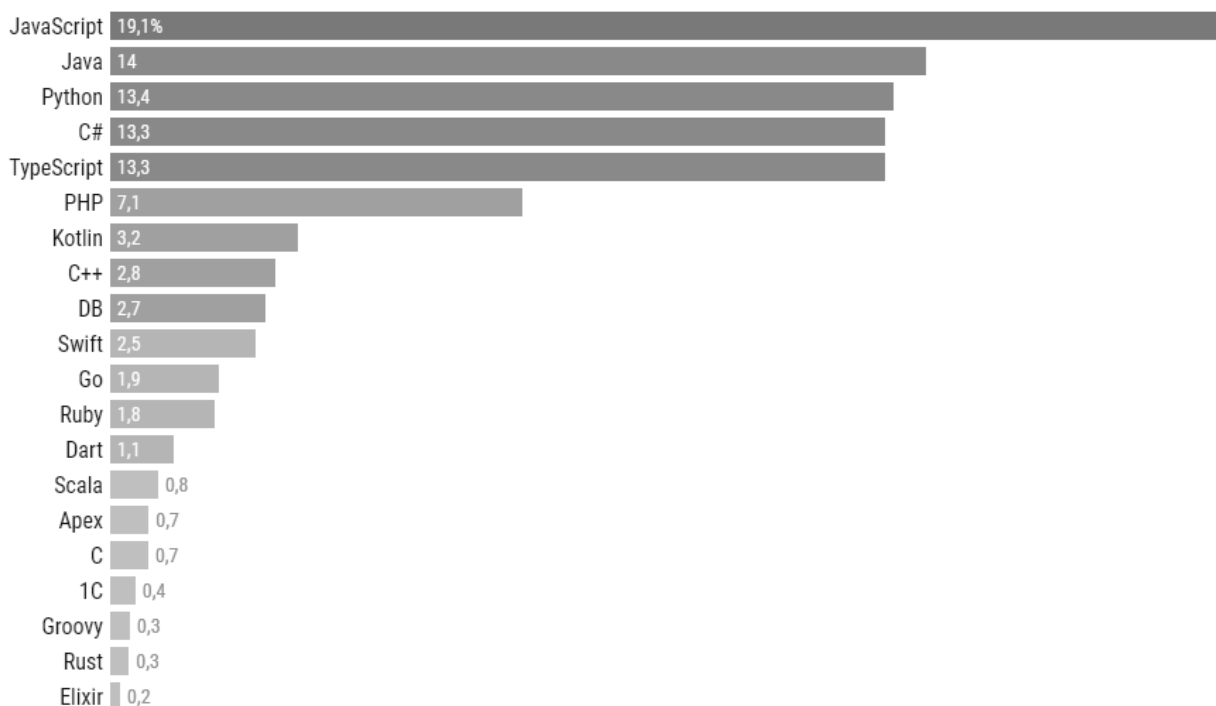


Рисунок 1.1 - Рейтинг мов використання у комерційній діяльності [3]

Джерело: [3]

JavaScript широко популярна у використанні у FROND-END розробці, а також може використовуватися і в BACK-END. про це свідчать данні з використання мов програмування на рисунку 1.2,1.3

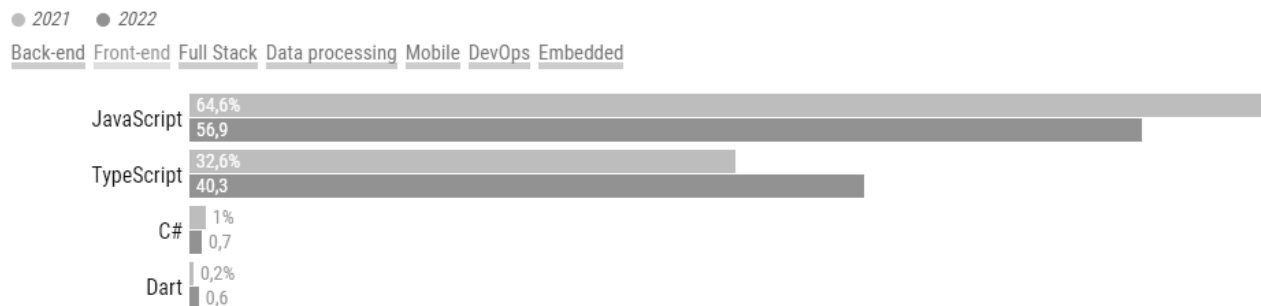


Рисунок 1.2 - Рейтинг Мови програмування FRONT-END

Джерело: [3]

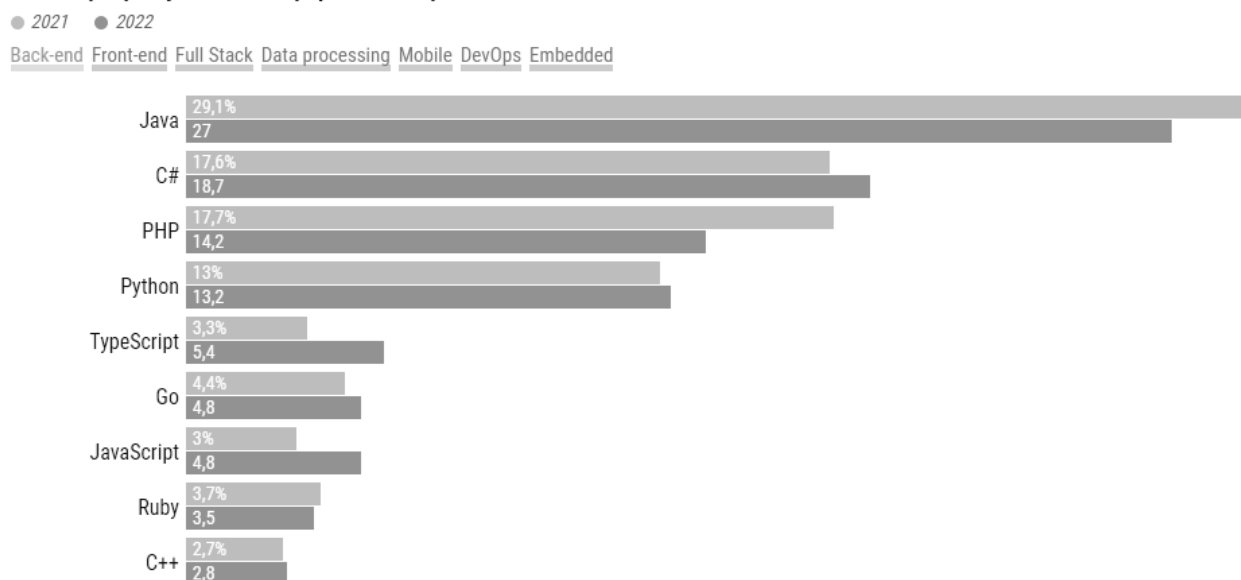
Мови програмування за сферами використання

Рисунок 1.3 - Рейтинг Мови програмування BACK-END

Джерело: [3]

Якщо подивитися на графіки рейтингу використання мов програмування то можна помітити, що JavaScript є однією з універсальних і зростаючих мов програмування, але слід зауважити, що і популярність TypeScript зростає пропорційно з JavaScript, а в BACK-END розробці навіть випереджає. Виходячи з отриманих даних для створення веб-сайту буде обрана мова програмування – JavaScript.

JavaScript (JS) — динамічна, об'єктно-орієнтована прототипна мова програмування. Реалізація стандарту ECMAScript. Найчастіше використовується для

створення сценаріїв веб-сторінок, що надає можливість на боці клієнта (пристрої кінцевого користувача) взаємодіяти з користувачем, керувати браузером, асинхронно обмінюватися даними з сервером, змінювати структуру та зовнішній вигляд вебсторінки.

Наразі для створення бази даних для веб-сайту існує два способи реалізації:

1. Хмарне сховище - Firebase, mongoDB, Amazon Web Services, Microsoft Azure та інші.
2. Локальне сховище – SQL, SQLite, PostgreSQL та інші.

Кожен зі способів реалізації має свої переваги і недоліки. Хмарні сховища надають можливість швидкої розробки, синхронізацію даних між додатками в реальному часі, готові інструменти для роботи, вже створені методи шифровки при передачі даних. Локальні сховище також мають такі функції, але для їх реалізації потрібно створювати власноруч, через це розробка може сильно упольнитися або потребувати більшої кількості людей залучених до проєкту.

У проєкту створення веб-сайту для ВСП «Глухівський агротехнічних фаховий коледж СНАУ» було обрано хмарне сховище Firebase. Дане сховище використовується у таких популярних додатках як: WhatsApp, Instagram, Twitter, Snapchat, SoundCloud та інші, всі вони використовують функції синхронізації даних та аутентифікації користувачів.

Firebase – це платформа для розробки додатків з величезним функціоналом від компанії Google. Головна перевага платформи в тому, що вона дозволяє розробнику не відволікатися на створення BACK-END, тобто прихованої від користувача програмної частини проєкту, наприклад, серверного коду. Firebase містить потужний REST API, який дозволяє розробникам отримувати доступ і маніпулювати даними Firebase із зовнішніх програм і служб [4].

Firebase REST API використовує стандартні HTTP-запити та відповіді для взаємодії з базами даних Firebase. Розробники можуть використовувати будь-яку мову програмування, яка може надсилати запити HTTP для надсилання запитів до Firebase. API підтримує різні методи HTTP, зокрема GET, POST, PUT, PATCH, DELETE.

Firebase REST API також підтримує автентифікацію, дозволяючи розробникам захищати свої бази даних Firebase і контролювати, хто може отримати до них доступ. Firebase надає кілька методів автентифікації, включаючи автентифікацію електронною поштою та паролем, автентифікацію через соціальну мережу та анонімну автентифікацію.

Використовуючи Firebase REST API, розробники можуть виконувати різні операції, такі як читання та запис даних у бази даних Firebase, надсилення сповіщень та керування автентифікацією користувачів. API добре задокументований і простий у використанні, що робить його чудовим вибором для створення веб-додатків і мобільних додатків, які вимагають синхронізації даних і зв'язку в реальному часі [6].

Однією з переваг при виборі було те, що Firebase має власну систему аутентифікації, власне файлове сховище, інтеграцію з системами Google, хостинг, саме головне, що компанія надає доступ до всіх цих функцій безкоштовно, хоча і є обмеження, але для веб-сайту кадрового відділу і не великої кількості користувачів, цих обмежень буде достатньо, але якщо потрібно розширити можливості, то цінова політика компанії є дуже гнучкою, плата йде лише за кожний використаний гігабайт, тобто не потрібно платити зайві кошти за не використаний простір.

Один з вибраних інструментів для розробки є Visual Studio Code. Перш за все, він може похвалитися потужним і універсальним набором функцій, які задовольняють потреби різних розробників. Незалежно від того, чи працюєте над веб-розробкою, аналізом даних або проектами машинного навчання, Visual Studio Code пропонує широку підтримку мови та інструменти для полегшення завдань кодування.

Крім того, розширюваність Visual Studio Code є видатною функцією. Завдяки великій бібліотеці розширень і плагінів можна розширити функціональність редактора та налаштувати його відповідно до своїх конкретних вимог.

Редактор забезпечує підсвічування синтаксису, автозавершення коду та розумні пропозиції, що значно прискорює процес розробки. Ці функції допомагають виявляти помилки та надають контекстно-залежні рекомендації, що дозволяє розробникам писати чистий ефективний код [11].

1.4 Постановка задачі

Головна мета цього проекту полягає в розробці веб-сайту для кадрового відділу та адміністрації коледжу ВСП "Глухівський агротехнічний фаховий коледж Сумського НАУ" з використанням мов програмування JavaScript, мови розмітки сторінки HTML та мови стилів CSS з препроцесором SCSS. Для зберігання даних буде використано хмарне сховище Firebase.

Створення веб-сайту є складним процесом, тому перед розробкою необхідно створити детальний план дій, щоб уникнути проблем під час розробки. З метою реалізації проекту можна виділити наступні основні задачі:

- Створення макету: На основі проаналізованих програмних аналогів та інших ПП з використанням графічного редактора Figma буде створений макет, який буде використовуватися для FRONT-END розробки сайту.
- Розробка веб-сторінок: За допомогою HTML, CSS та JavaScript будуть створені веб-сторінки, що відповідають за зовнішній вигляд сайту та його функціональність.
- Робота з базою даних: Заповнення таблиць з даними буде відбуватися за допомогою звернень до сервера бази даних. Firebase буде використовуватися для зберігання та управління даними.
- Управління файлами: Файли, що завантажуються на сайт, будуть зберігатися у хмарному сховищі Firebase Storage.
- Безпека авторизації: Забезпечення захищеної авторизації на сайті є важливою задачею. Повинно використовуватися механізми аутентифікації та захисту даних, щоб запобігти несанкціонованому доступу до сайту та його функціоналу.
- Хостинг та розгортання: Пошук хостингу, оренда місця на хостингу та розміщення сайту на ньому. Результатом розробки даного проекту буде створений сайт, у якому більшість даних буде виводитися з бази даних, користувачі даного сайту зможуть працювати з текстовою, файловою складовою та робити функції CRUD.

Результатом розробки даного проекту буде створений веб-сайт, в якому більшість даних буде виводитися з бази даних. Користувачі даного сайту зможуть працювати з текстовими та файловими складовими, а також виконувати функції CRUD (створення, читання, оновлення та видалення даних). Цей проект надасть зручність та ефективність у роботі кадрового відділу та адміністрації коледжу.

РОЗДІЛ 2

МОДЕЛЮВАННЯ ТА ПРОЕКТУВАННЯ ІНФОРМАЦІЙНОЇ ПІДСИСТЕМИ ОБЛІКУ КАДРОВОГО ЗАБЕЗПЕЧЕННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ КОЛЕДЖУ ВСП «ГАФК СНАУ»

2.1 Моделювання розроблюваної підсистеми

У даному розділі наводиться моделювання процесу роботи веб-сайту за методологією IDEF. Ця методологія включає в себе контекстну діаграму та діаграму декомпозиції першого та другого рівня, та може бути корисною для аналітиків, проектувальників, менеджерів проектів та інших фахівців, які оптимізують і покращують роботу системи.

Контекстна діаграма для веб-сайту коледжу ВСП «ГАФК СНАУ» являє собою основний процес створення та редагування картки працівника, даний процес наведено на рисунку 2.1.

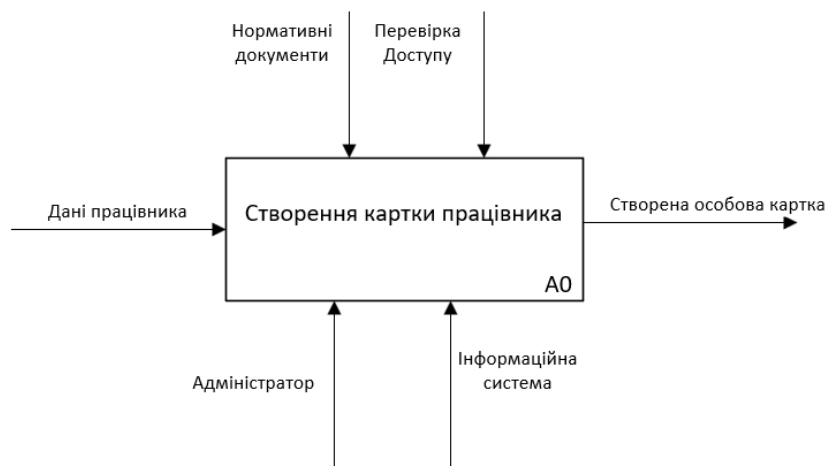


Рисунок 2.1 – Контекстна діаграма

Джерело: запропоноване автором

За контекстною діаграмою, яка зображена на рисунку. 2.1 можемо побачити, що на вхід до постуває:

- Форма для створення особової картки;
- Особисті файли про працівника.

Елементами керування є:

- Автентифікація користувача;
- Перевірка доступу;
- Редагування прав доступу;
- Перевірка наявності дублікатів;
- Створення нового запису.

Механізмами системи є:

- Адміністратор;
- База даних;
- Читач.

На виході системи отримуємо:

- Особову картку працівника;
- Оновлення бази даних.

Діаграми першого рівня деталізують функції процесу нульового рівня. Так, функціональний блок 0 розкладається на сукупність взаємопов'язаних підфункцій, а саме створення картки працівника, перевірка введених даних, відображення інформації. На рисунку 2.2 зображена діаграма першого рівня.

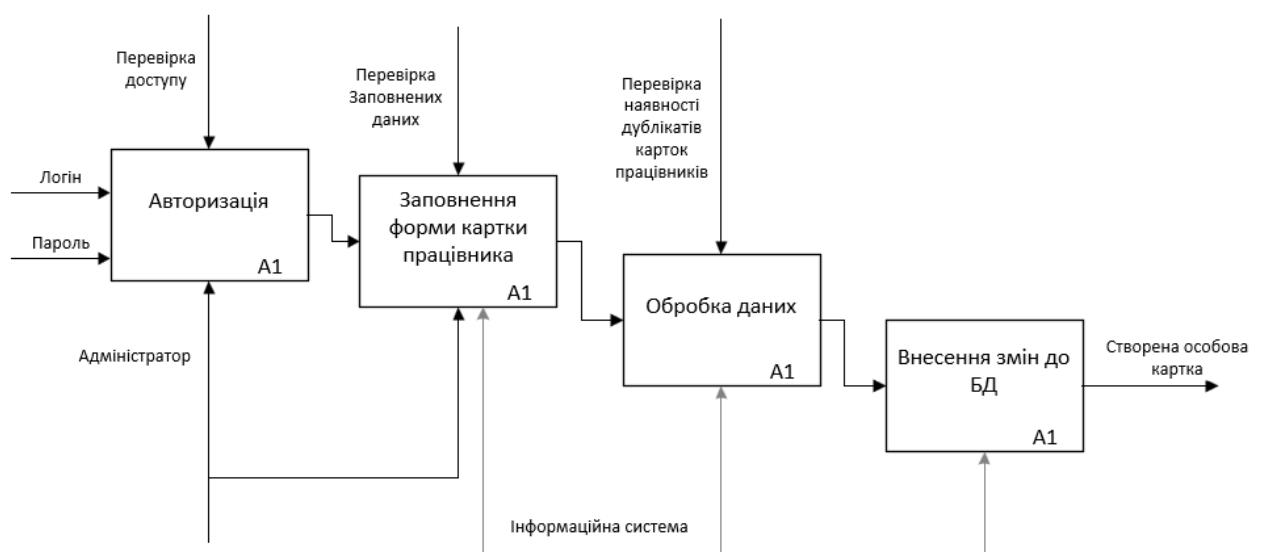


Рисунок 2.2 – Діаграма першого рівня

Джерело: запропоноване автором

2.2 Проектування інформаційної підсистеми

У сучасному світі веб-сайти є невід'ємною частиною бізнесу та суспільного життя. Користувачі звертають увагу не лише на зміст сайту, а й на його візуальне оформлення та зручність використання. Важливою складовою дизайну є те, щоб дизайн був інтуїтивно зрозумілий для користувачів, легко читабельний і зручний для використання.

На основі отриманих даних від аналізу програм-аналогів буде створено план та початковий прототип інтерфейсу, який буде доопрацьовуватися та оптимізуватися упродовж усієї роботи, на рисунку 2.3 та рисунку 2.4 зображено інтерфейси, які використовуються в програмах-аналогах.

Перед початком розробки прототипу потрібно розробити план того, що буде розміщено на веб-сайті. В даному випадку веб-сайт являє собою роботу з базою даних, тож сайт буде містити такі сторінки як:

- Сторінка реєстрації
- Сторінка авторизації
- Сторінка профілю користувача. Для ролі «Адміністратор» таблиця всіх користувачів системи;
- Головна сторінка бази даних кадрового відділу. Для ролі «Адміністратор» доступ до функцій CRUD ;
- Сторінка роботи з файловим сховищем. Для ролі «Адміністратор» доступ до функцій CRUD;
- Сторінка на якій відображатимуться працівники які були звільнені . Для ролі «Адміністратор» доступ до функцій CRUD;
- Сторінка з календарем подій.

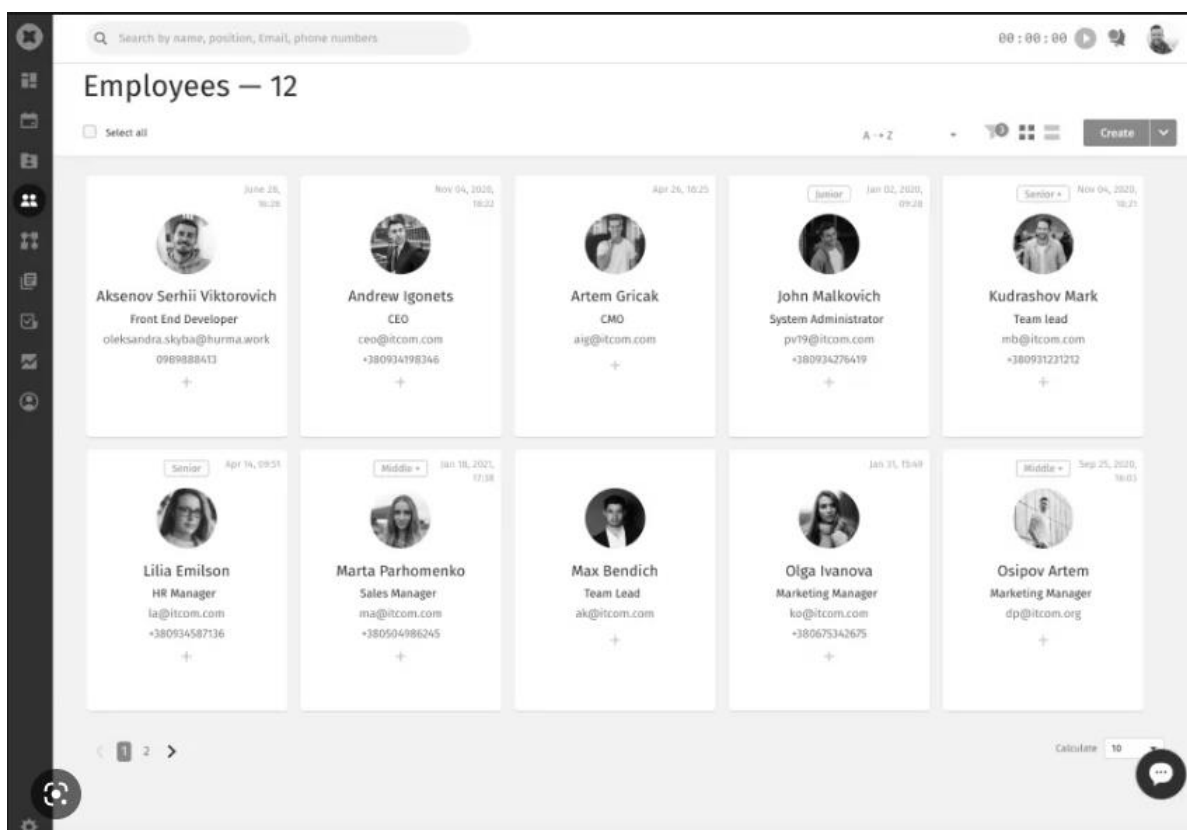


Рисунок 2.3 – дизайн веб-сайту HurmaSystem

Джерело: [12]

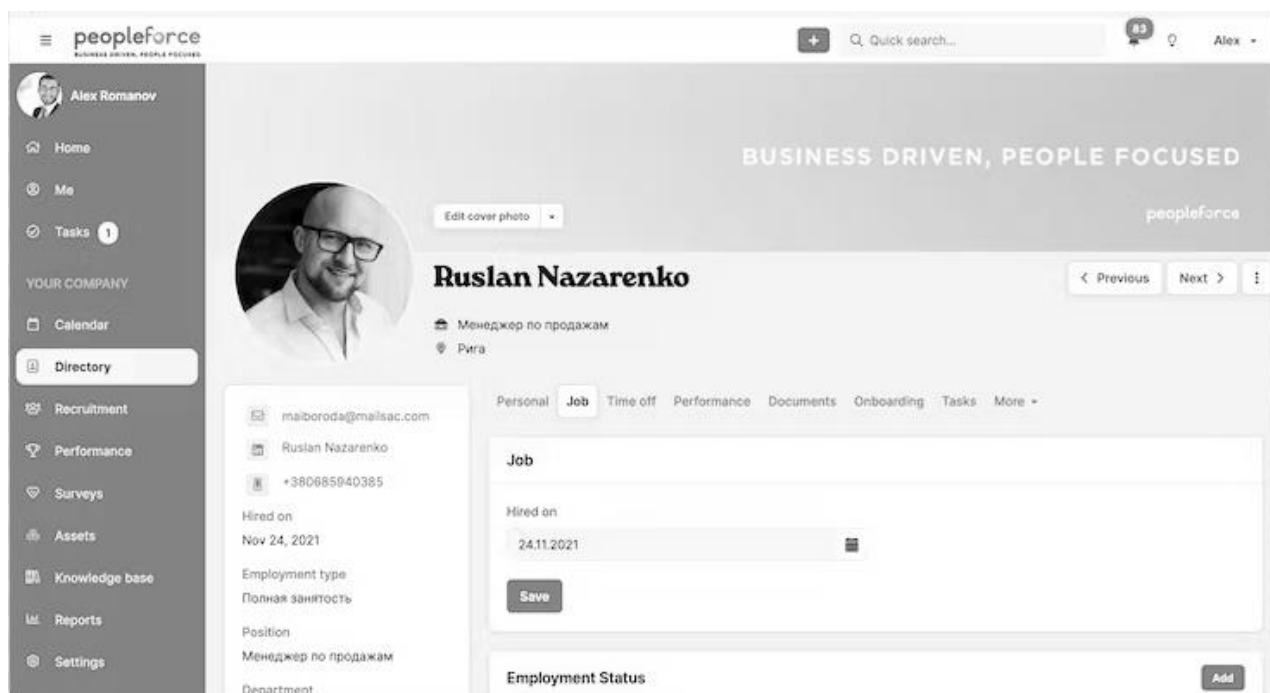


Рисунок 2.4 – дизайн веб-сайту PeopleForce

Джерело: [13]

Прототип інтерфейсу може бути зроблений вручну або за допомогою спеціальних програм для дизайну інтерфейсу, таких як Sketch, Figma, Adobe XD і т.д. Прототип дозволяє протестувати функціональність інтерфейсу та визначити, яким чином користувачі будуть взаємодіяти з сайтом.

Програми для дизайну дуже схожі між собою, за принципом роботи відрізняються лише зовнішнім оформленням, тож не має жодного сенсу обирати кращий з них. Так як я вже маю досвід роботи з Figma, то буде обрано цей ПП.

Figma — векторний онлайн-сервіс розробки інтерфейсів та прототипування з можливістю організації спільної роботи, що розробляється однойменною компанією[7]

Було розроблено прототип для наступних сторінок:

- Сторінка авторизації ;
- Сторінка реєстрації;
- Основна сторінка;
- Діалогове вікно для створення, редагування, видалення записів
- Сторінка особистого профілю.

На рисунку 2.5 зображено прототип сторінки авторизації та мобільна адаптація.

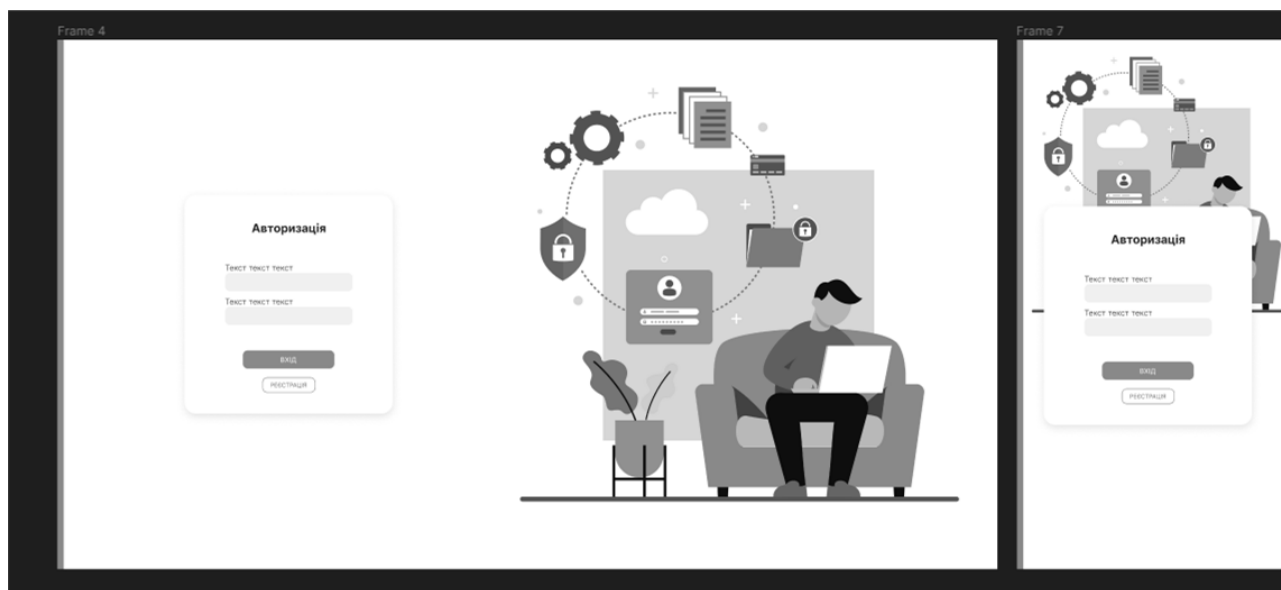


Рисунок 2.5 – Прототип сторінки авторизації і мобільної адаптації до неї

Джерело: запропоноване автором

На рисунку 2.6 зображено прототип сторінки реєстрації та мобільна адаптація до неї.

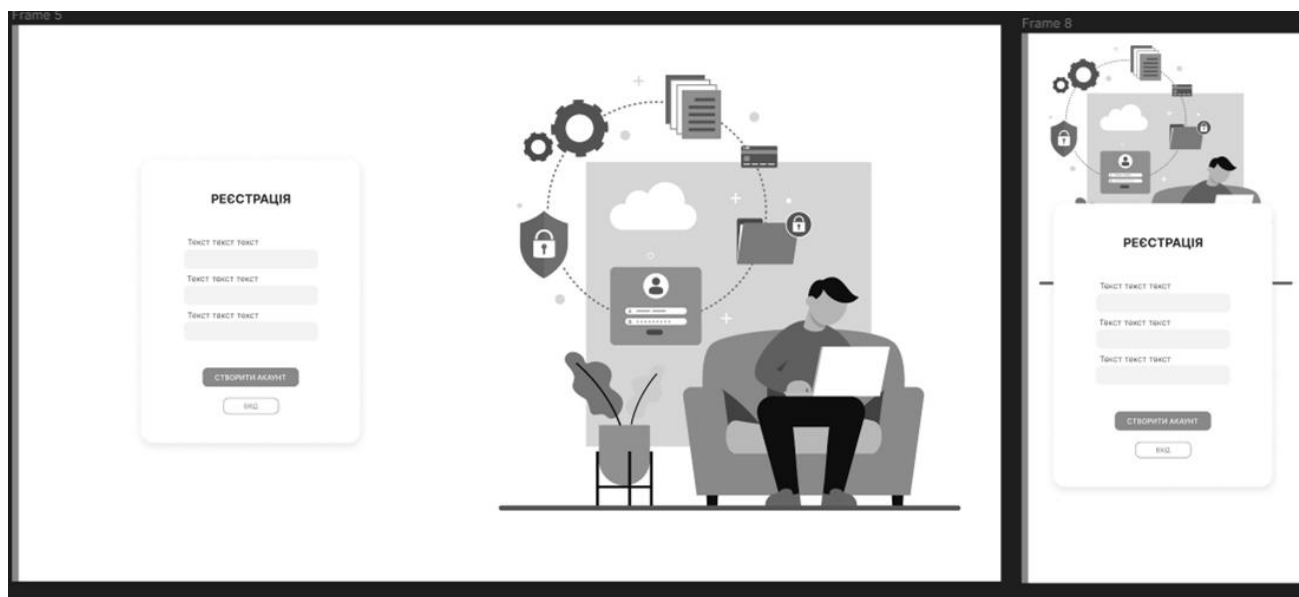


Рисунок 2.6 – Прототип сторінки реєстрації і мобільної адаптації до неї

Джерело: запропоноване автором

На рисунку 2.7 зображено прототип головної сторінки, меню навігації та модального вікна для заповнення інформації.

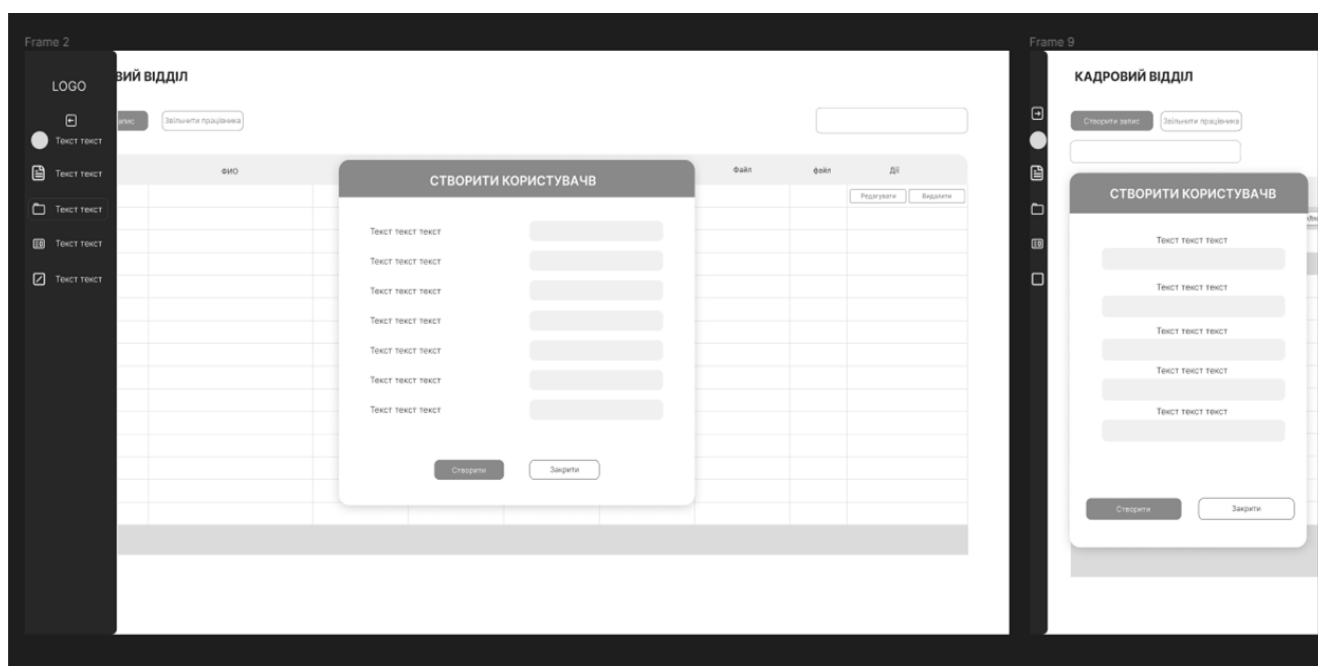


Рисунок 2.7 – Прототип головної сторінки для роботи з базою даних

Джерело: запропоноване автором

Коли прототип готовий, його можна протестувати фокус-групою або тестувати з користувачами. Це дозволяє отримати зворотний зв'язок від реальних користувачів та внести необхідні зміни в інтерфейс для покращення його зручності та функціональності.

Насамкінець можна сказати, що розробка прототипу інтерфейсу для сайту є важливим етапом у створенні якісного та зручного сайту. Процес розробки прототипу інтерфейсу має бути ретельно продуманий і ґрунтуватися на аналізі потреб користувачів. Також важливо враховувати вимоги до адаптивності та мобільності сайту, щоб він був зручним для використання на різних пристроях.

2.3 Проектування моделі бази даних

Проектування моделі бази даних є важливою частиною розробки будь-якої інформаційної системи. Основна мета проектування моделі бази даних полягає в створенні ефективної та ергономічної структури для зберігання даних

Етап проектування полягає в пошуку і визначенні якнайкращого способу реалізації та виконання вимог, сформульованих на етапі аналізу. При цьому має забезпечуватись належний рівень сервісу в умовах певного технологічного середовища, що відповідає ухваленим рішенням щодо рівня автоматизації.

Логічне проектування — це розроблення структур зберігання, методів доступу й логічної структури системи баз даних без прив'язки до конкретної СКБД.

Фізичне проектування — це проектування бази даних у конкретній СКБД

Створення бази даних слід починати з її проектування (розробки). У результаті проектування має бути визначена структура бази, тобто склад таблиць, їхня структура та логічні зв'язки. Структура реляційної таблиці визначається складом стовпців, їхньою послідовністю, типом даних кожного стовпця та їхнім розміром, а також ключем таблиці [5].

На рисунку 2.9 зображена концептуальна модель бази даних розроблюваної системи.

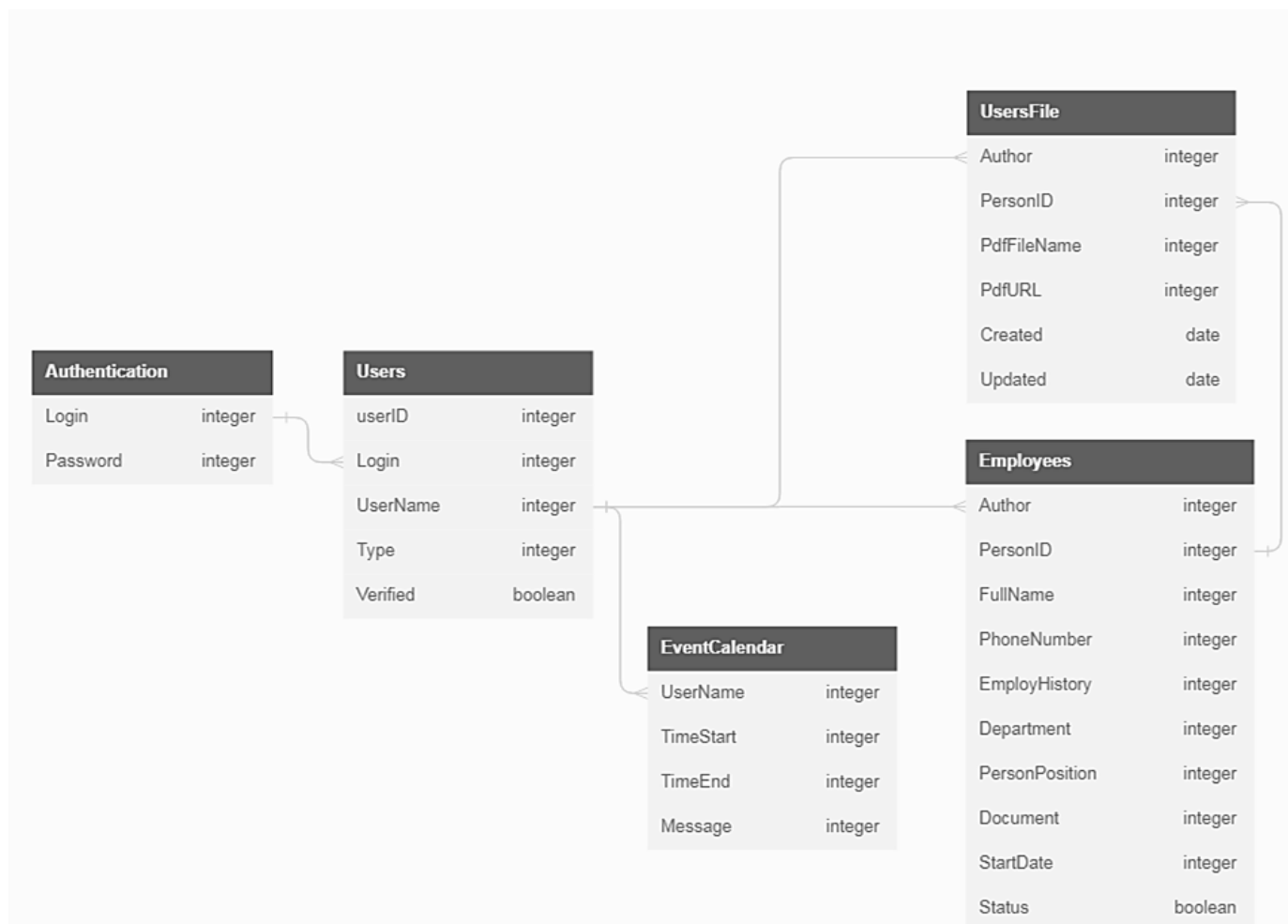


Рисунок 2.9 - Концептуальна модель бази даних системи

Джерело: запропоноване автором

База даних системи містить такі сутності:

- **Authentication**: так як Firebase для авторизації має окреме сховище, тож авторизацію було розбито на Authentication і Users. Authentication перевіряє лише правильність введених даних ;
- **Users**: містить інформацію про користувача таку як: унікальний ID, логін, ім'я, тип облікового запуску, статус верифікації користувача;
- **Employees**: містить основну інформацію про працівника;
- **UsersFile**: файлове сховище яке містить файли в форматі PDF для кожної картки працівника;
- **EventCalendar**: календар подій та нотаток для користувачів.

Зв'язки між сутностями:

Authentication – Users: перевіряє користувача на роль у базі даних та верифікацію профілю.

Users – Employees: якщо користувач має роль «Адміністратор» то він може створювати, редагувати, видаляти записи для карток працівників і його ім'я буде відображене у полі «Автор».

Users – UsersFile: якщо користувач має роль «Адміністратор» то він може створювати, редагувати, видаляти файли і його ім'я буде відображене у полі «Автор».

Employees – UsersFile: при створенні картки працівника файли які завантажуються передаються в окреме хмарне сховище, для створення зв'язку між двома сховищами було використано ID працівника.

Users – EventCalendar: містить ім'я автора що створив, текст події, дату початку і завершення події.

РОЗДІЛ 3

РОЗРОБКА ІНФОРМАЦІЙНОЇ ПІДСИСТЕМИ ОБЛІКУ КАДРОВОГО ЗАБЕЗПЕЧЕННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ КОЛЕДЖУ ВСП «ГАФК СНАУ»

3.1 Архітектура програмного забезпечення

Архітектура програмного забезпечення — спосіб структурування програмної або обчислювальної системи, абстракція елементів системи на певній фазі її роботи . Система може складатись з кількох рівнів абстракції і мати багато фаз роботи, кожна з яких може мати окрему архітектуру. [14].

Архітектура ПЗ є критичним елементом у процесі розробки ПП, оскільки вона визначає основні характеристики та якість ПП, такі як ефективність, безпека, масштабованість та модульність. Крім того, правильно спроектована архітектура дозволяє забезпечити повторне використання коду та підтримку ПП протягом його життєвого циклу.

У розробці буде використано архітектуру клієнт-сервер, яка зображена на рисунку 3.1. Архітектура ПЗ клієнт-сервер - це модель взаємодії між комп'ютерними програмами, в якій одна програма (клієнт) використовує послуги іншої програми (сервера). Клієнт і сервер можуть бути виконані на різних комп'ютерах в мережі або на одному комп'ютері.

Клієнтська програма зазвичай забезпечує інтерфейс користувача та взаємодіє з користувачем, а серверна програма забезпечує виконання певних завдань, які запитуються клієнтською програмою. Клієнт-сервер використовується коли потрібно об'єднати ресурси та сервіси, до яких потрібно забезпечити доступ великої кількості розподілених клієнтів, і при цьому необхідно контролювати доступ або якість обслуговування.

Однією з головних переваг архітектури клієнт-сервер є розділення функцій програми між клієнтом та сервером. Це дозволяє покращити масштабованість та продуктивність, оскільки сервер може обробляти більші обсяги даних та завдань, ніж

клієнтська програма. Крім того, ця архітектура дозволяє забезпечити безпеку, оскільки сервер може контролювати доступ до даних та ресурсів.

На рисунку 3.1 зображена архітектура програмного забезпечення для типу клієнт-сервер.

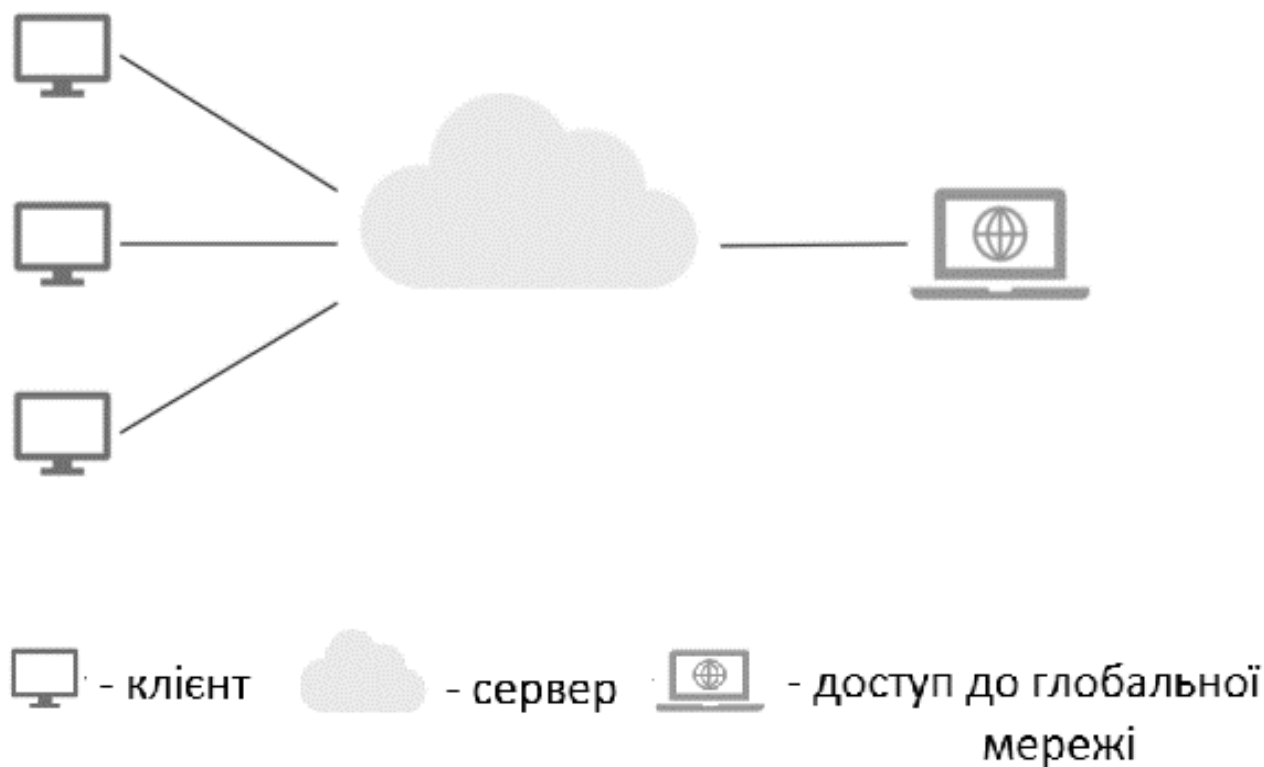


Рисунок 3.1 – Архітектура програмного забезпечення клієнт-сервер

Джерело: [14]

3.2 Програмна реалізація

Перед початком роботи було створено структуру проєкту, це дозволить визначити взаємозв'язок файлів та їх місцезнаходження. Структура проєкту складається з HTML, SCSS, JavaScript і CSS файлів. Структуру створеного проєкту показано на рисунку 3.2.

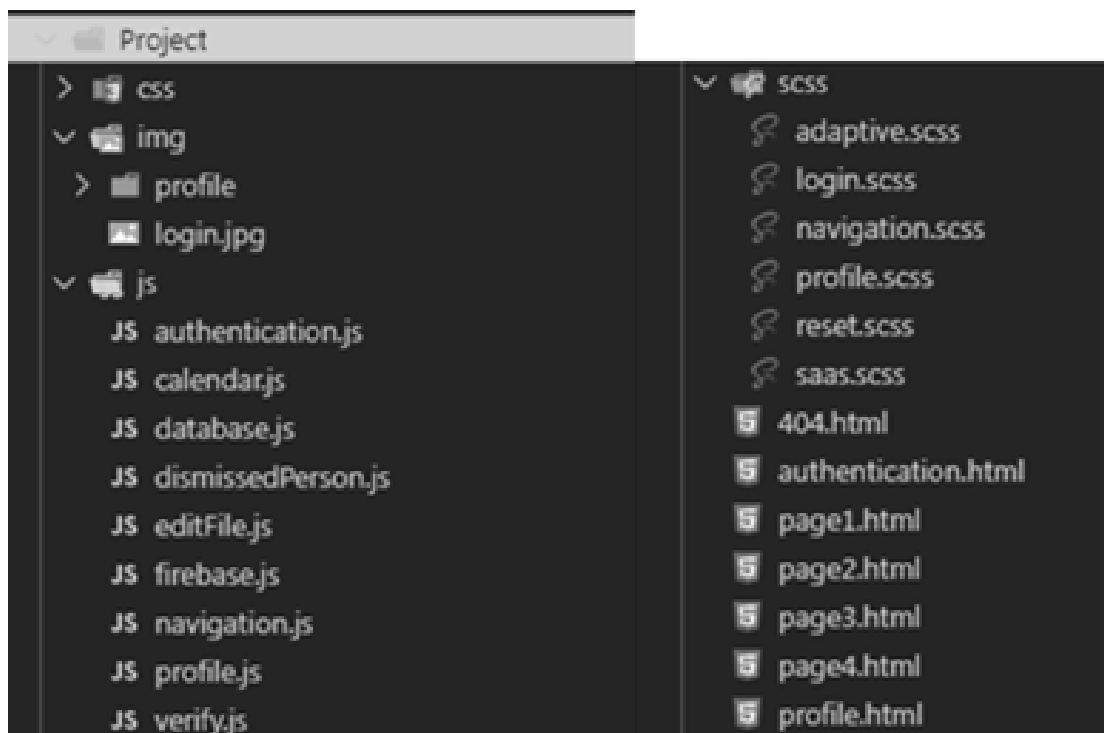


Рисунок 3.2. – Структура веб-сайту

Джерело: запропоноване автором

Перед початком розробки сайту потрібно підключити Firebase database, підключення було створено в окремому файлі – firebase.js (рисунок 3.5)

```
Project > js > JS firebase.js > ...
1  /// DATABASE CONNECT
2  export const firebaseConfig = {
3    apiKey: "AIzaSyASbriNsGigCr1_wvbqNSl9upAilGe1zZ4",
4    projectId: "danil-86157",
5    storageBucket: "danil-86157.appspot.com",
6    messagingSenderId: "400192790550",
7    appId: "1:400192790550:web:a0d1a58eb6533200a6d537"
8  };
9
10 import { getApp, initializeApp } from "https://www.gstatic.com/firebasejs/9.18.0/firebase-app.js";
11 import { getDatabase, ref } from "https://www.gstatic.com/firebasejs/9.18.0/firebase-database.js";
12 import { getStorage } from "https://www.gstatic.com/firebasejs/9.18.0/firebase-storage.js";
13
14 // Initialize Firebase + firebase Storage
15 export const app = initializeApp(firebaseConfig);
16 export const firebaseApp = getApp();
17 export const db = getDatabase();
18 export const dbRef = ref(db, `employees/`);
19 export const storage = getStorage(firebaseApp, "gs://danil-86157.appspot.com");
20
```

Рисунок 3.5. – Підключення Firebase database до веб-сайту

Джерело: запропоноване автором

Для ініціалізації функцій було використано імпортовано об'єкти `getApp`, `initializeApp`, `getDatabase`, `getStorage`, данні об'єкти дозволяють підвантажувати функції Firebase до веб-сайту, а `firebaseConfig` підключає особисту створену базу даних до веб-сайту. Для уникнення дублювання в коді було використано експорт класів і об'єктів, експорт функцій надає доступ до цих об'єктів в будь якому іншому js файлі.

Налаштування доступу до Firebase Realtime зображено на рисунку 3.3, в даному меню ми надаємо можливість роботи функцій читання і запису інформації за допомогою запитів до бази.

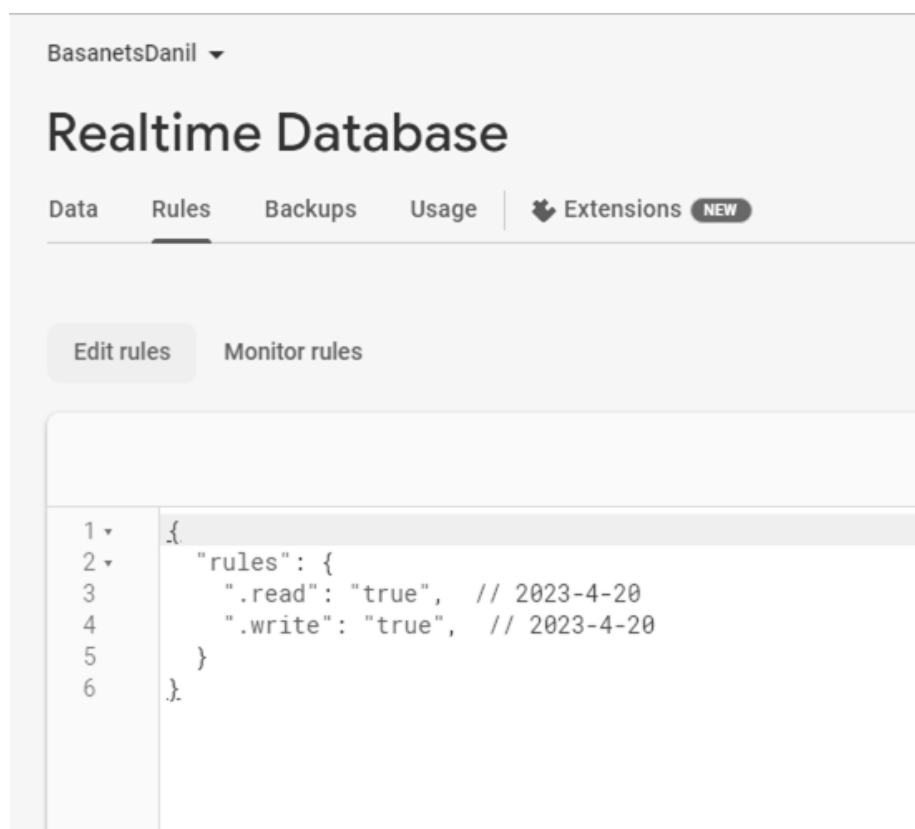


Рисунок 3.3. – Налаштування Firebase database

Джерело: запропоноване автором

Так як наша база даних повинна зберігати ще й файли, то в Firebase потрібно окремо налаштувати файлове сховище. Файлове сховище за даними параметрами має дозвіл на читання і запис інформації (рисунок 3.4).

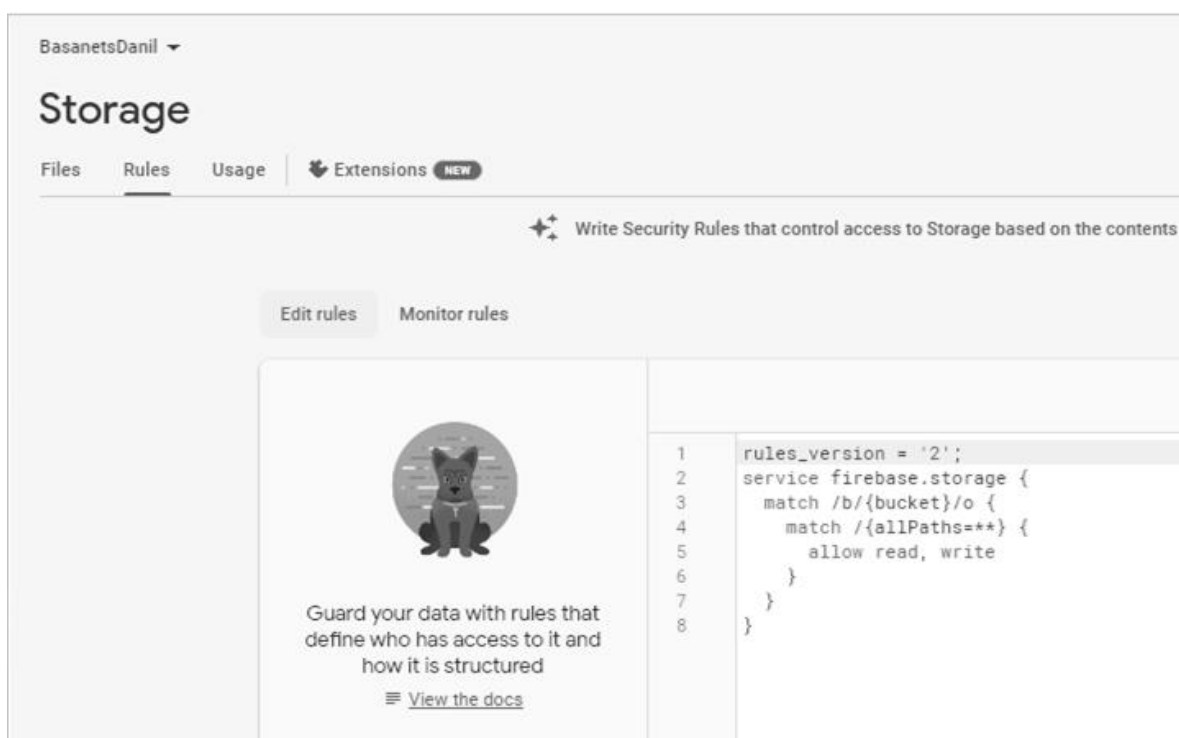


Рисунок 3.4. – Налаштування Firebase Storage

Джерело: запропоноване автором

Для забезпечення безпеки даних було створено сторінку авторизації, реєстрації та перевірки користувачів. Ця сторінка містить два файли - authentication.js та verify.js, які відповідають за функціонал авторизації та перевірки користувачів.

У файлі authentication.js знаходиться функція signUpFunc(), яка відповідає за створення нових користувачів і запис їх даних до бази даних. Параметри, які записуються, включають унікальний ID користувача, ім'я, електронну адресу та тип облікового запису. За замовчуванням, тип облікового запису встановлюється як "гість", а статус верифікації - "False" (рисунок 3.7).

Функція signInFunc() в тому ж файлі, authentication.js, виконує запит до бази даних і перевіряє правильність введених користувачем даних під час авторизації. Якщо введені дані невірні, то з'явиться повідомлення про помилку "Помилка авторизації" (рисунок 3.8).

Файл verify.js відповідає за перевірку того, чи користувач авторизований на сайті. Якщо користувач не авторизований, він буде автоматично перенаправлений на сторінку авторизації (рисунок 3.6).

Ці функції та файли були створені для забезпечення безпеки та контролю доступу до сайту, дозволяючи користувачам реєструватись, авторизуватись і перевіряти свій статус на сайті.

```
Project > js > JS verifyjs > ...
1  /// DATABASE CONNECT
2  import "https://www.gstatic.com/firebasejs/8.2.7/firebase-app.js"
3  import "https://www.gstatic.com/firebasejs/8.2.7/firebase-auth.js"
4  import { firebaseConfig } from "../firebase.js";
5  firebase.initializeApp(firebaseConfig);
6
7
8  firebase.auth().onAuthStateChanged(function (user) {
9    if (!user) {
10      window.location.href = "authentication.html";
11    }
12  });
```

Рисунок 3.6. – Код для перевірки авторизації користувачів Verify.js

Джерело: запропоноване автором

```
auth.createUserWithEmailAndPassword(email, password)
  .then((userCredential) => {
    const user = userCredential.user
    user.updateProfile({
      displayName: nameRegisterInp
    })
    set(ref(db, 'users/' + user.uid), {
      uid: user.uid,
      name: nameRegisterInp,
      email: email,
      profile: "guest",
      verified: false
    });
    menuRegister.style.display = "none"
    menuAuth.style.display = ""
    colorColumn.style.background = "#18A0FB"
  })
  .catch(function (error) {
    var errorMessage = error.message;
    alert(errorMessage);
  });
}
```

Рисунок 3.7. – Код Authentication.js для реєстрації користувачів

Джерело: запропоноване автором

```

function signInFunc() {
  var email = document.querySelector(".emailLoginInp").value
  var password = document.querySelector(".passwordLoginInp").value;

  firebase.auth().setPersistence(firebase.auth.Auth.Persistence.SESSION)
    .then(function () {
      firebase.auth().signInWithEmailAndPassword(email, password)
        .then(function (userCredential) {
          const currentUser = userCredential.user.uid

          get((ref(db, `users/${currentUser}`))).then(querySnapshot => {
            const verifiedAcc = querySnapshot.val().verified
            if (verifiedAcc) { window.location.href = "index.html"; }
            else alert("чекайте підтвердження облікового запису")
          })
        })
    })
}

```

Рисунок 3.8. – Код Authentication.js для авторизації користувачів
Джерело: запропоноване автором

Прототип веб-сайту передбачає наявність блоку навігації, для якого були створені окремі файли - navigation.js та navigation.html (рисунок 3.9). Цей код дозволяє забезпечити перехід між різними сторінками веб-сайту та надати інтерактивність цьому блоку.

```

12  imgRight.addEventListener("click", () => {
13    navLogoOpen.style.display = ""
14    navLogoClose.style.display = "none"
15    navContainer.style.width = "220px"
16    setTimeout(() => {
17      imgRight.style.display = "none"
18      imgLeft.style.display = ""
19    }, 50)
20    setTimeout(() => {
21      for (let i = 0; i < navP.length; i++) {
22        navP[i].style.display = "block";
23      }
24    }, 150)
25  })
26
27
28  imgLeft.addEventListener("click", () => {
29    imgLeft.style.display = "none"
30    imgRight.style.display = ""
31    navLogoOpen.style.display = "none"
32    navLogoClose.style.display = ""
33    navContainer.style.width = "45px"
34    for (let i = 0; i < navP.length; i++) {
35      navP[i].style.display = "none";
36    }
37  })
38

```

Рисунок 3.9. – Код navigation.js для навігації
Джерело: запропоноване автором

Розглянемо основний код database.js, який відповідає за створення таблиці для виведення даних. На рисунку 3.10 представлений код, який надсилає запит до Firebase для отримання даних, а отримані дані відображаються в таблиці. Цей фрагмент коду дозволяє отримати потрібну інформацію з бази даних та представити її у вигляді таблиці.

```
// Read database
function readDB() {
  const response = get(dbRef).then((
    querySnapshot) => {
    if (querySnapshot.exists()) {
      const arr = Object.values(querySnapshot.val())
      let data = arr.reduce((acc, curr) => {
        if (curr.status) {
          acc.push(curr);
        }
      }, []);
      return data;
    }
    return response;
  }).catch((error) => {
    console.error(error);
  });
}
```

Рисунок 3.10 – Запит до бази даних на отримання інформації за допомогою GET

Джерело: запропоноване автором

Крім того, на рисунку 3.11 зображений фрагмент коду, який використовується для побудови таблиць, їх заповнення, створення пагінації та пошуку даних на веб-сайті. Цей код включає в себе функціонал, що дозволяє створити таблиці з необхідною кількістю рядків та стовпців, заповнити їх відповідними даними, а також додати можливість пагінації для перегляду багатої кількості даних та пошуку конкретної інформації на веб-сайті.

Ці файли та фрагменти коду були створені з метою забезпечення зручного та ефективного виведення даних на веб-сайті, а також забезпечення користувачів інтерактивними можливостями для навігації та роботи з даними.

Оскільки веб-сайт має два типи користувачів, для користувача з типом «Гість» створення табличок не буде містити кнопок, які взаємодіють з базою даних, тому фрагмент коду на рядках 462-470 з рисунку 3.11 буде відсутній.

Для встановлення типу облікового запису користувача буде створено додаткову функцію, яка робить запит до бази даних авторизації, отримує унікальний ID користувача, за цим ID вже в RealtimeDB знаходить цього користувача, його ім'я та тип облікового запису. За рисунком 3.12 можемо побачити функцію, яка перевіряє тип облікового запису. Якщо тип облікового запису «Admin», то буде створено додаткові кнопки, такі як: 1) Створення картки працівника; 2) редагування картки працівника; 3) звільнення працівника; 4) видалення картки працівника.

```

firebase.auth().onAuthStateChanged(updateUser);
function updateUser(user) {
  const currentUser = user ? user.uid : null;
  get((ref(db, `users/${currentUser}`))).then(querySnapshot => {
    const typeProfile = querySnapshot.val().profile
    const nameProfile = querySnapshot.val().name
    if (typeProfile === "admin") {
      // render page
      renderTable(nameProfile)

      // HTML content
      const btnWrapper = document.querySelector('.btnWrapper');
      const table = document.querySelector(".table")
      const createPersonBtn = document.querySelector(".createPerson")
      const dismissalPerson = document.querySelector(".dismissalPerson")

      // Create new person
      createPersonBtn.onclick = () => { ...
    }

    // dismissal person
    dismissalPerson.addEventListener('click', () => { ...
  })

  // delete person btn
  table.addEventListener('click', (event) => {
    if (event.target.classList.contains('btnDell')) { ...
  }
})

  // edit persone btn
  table.addEventListener('click', (event) => { ...
})

  // render message to create person
  function BoxToCreatePerson(nameBtn, valueBtn) { ...
  }
}
// else typeprofile = guest
else {
  renderTableGuest()
}
})
}

```

Рисунок 3.12 – Код для відображення сторінки для різних типів користувачів

Джерело: запропоноване автором

Оскільки при реєстрації кожен користувач вказує своє ім'я, в коді реалізована функція, яка визначає, який саме користувач вніс зміни до бази даних. Це дозволяє зберігати інформацію про те, хто виконує певні дії та вносить зміни.

```

375 // Create table ADMIN
376 function renderTable() {
377     readDB().then((data) => {
378         grid = new gridjs.Grid({
379             resizable: true,
380             sort: true,
381             search: true,
382             pagination: { limit: 15 },
383             language: { ...
395         },
396         fixedHeader: true,
397         columns: [ ...
451     ],
452     data: data.map((row) => (
453         [
454             row.personID,
455             row.fullName,
456             row.department,
457             row.personPosition,
458             row.startDateStr,
459             row.emplHistoryStr,
460             row.commandStr,
461             row.phoneNumStr,
462             gridjs.html(`
463             <a href="${row.pdfURL}" target="_blank">${row.pdfFileName}</a>
464             `),
465             gridjs.html(`
466             <i class="fa-regular fa-pen-to-square fa-lg btnEdit" id="${row
467             data-action="edit" style="color: #555658;"></i>
468             <i class="fa-solid fa-trash-can btnDell" id="${row.personID}"
469             title="Видалити" style="color: #555658;"></i>
470             `),
471             row.avtor]))
472     })
473     grid.render(document.getElementById("table"));
474 })
475 }

```

Рисунок 3.11 – Код для побудови і виводу на сторінки таблиць

Джерело: запропоноване автором

Наведений у розділі код є лише невеликою частиною основної коду логіки, що використовується при створенні веб-сайту. Для більш повного розуміння функціональності сайту та його реалізації можна звернутися до ДОДАТКУ Д, де розміщений весь інший код веб-сайту.

ДОДАТОК Д включає в себе решту коду, що не був представлений у даному розділі. В цьому додатку можна знайти додаткові функції, модулі та інші складові частини веб-сайту, які допомагають забезпечити його повну функціональність.

Після створення веб-додатку потрібно його розмістити на хостингу, щоб інші користувачі підприємства також мали змогу користуватися цим веб-додатком. Для розміщення на хостингу було обрано Firebase Hosting, в нашому випадку це дуже зручно, так як база даних знаходиться теж в системі Firebase.

Перед початком роботи потрібно встановити додаткову бібліотеку Firebase CLI, для цього в терміналі середовища програмування або в командному рядку потрібно вписати команду «`npm install -g firebase-tools`». Данна команда встановлює додаткові бібліотеки які дозволяють налаштувати завантаження веб-сайта на хостинг.

Після успішного встановлення бібліотеки з'явиться меню (рисунок 3.13) в якому потрібно обрати меню з конфігурації і розгортання хостингу

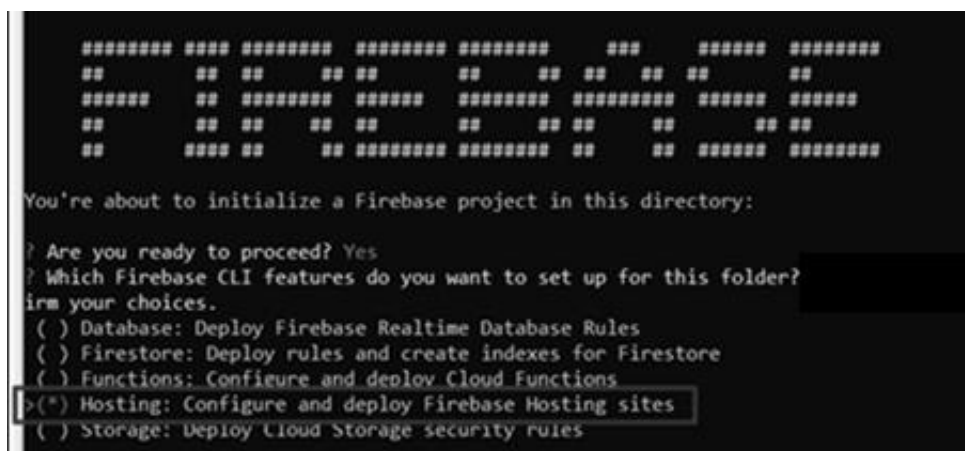


Рисунок 3.13 – Меню налаштування встановленої бібліотеки Firebase

Джерело: запропоноване автором

З рис 3.14 налаштовуємо параметри для завантаження на хостинг. Параметри мають наступні значення:

- Чи публічний веб-сайт? Так
- Чи початкова сторінка має назву `index.html`? Так
- Чи потрібно завантажувати веб-сайт на GitHub? Так

- Чи потрібно створити сторінку з помилкою? Так

```

=== Hosting Setup

Your public directory is the folder (relative to your project directory) that
will contain Hosting assets to be uploaded with firebase deploy. If you
have a build process for your assets, use your build's output directory.

? What do you want to use as your public directory? public
? Configure as a single-page app (rewrite all urls to /index.html)? No
? Set up automatic builds and deploys with Github? No
+ Wrote public/404.html
? File public/index.html already exists. Overwrite? (y/N) |I

```

Рисунок 3.14 – Налаштування параметрів проєкту

Джерело: запропоноване автором

Повідомлення на рис 3.15 означає що веб-сайт завантажений на хостинг і готовий до роботи. Посилання на сайт можна знайти в Firebase Hosting

```

Created alias firebase-react-deploy for fir-deploy-1fe86.
Now using alias firebase-react-deploy (fir-deploy-1fe86)
[The-Coding-Machine:firebase-deploy Rioux$ firebase deploy

=== Deploying to 'fir-deploy-1fe86'...

i deploying hosting
i hosting[fir-deploy-1fe86]: beginning deploy...
i hosting[fir-deploy-1fe86]: found 16 files in build
✓ hosting[fir-deploy-1fe86]: file upload complete
i hosting[fir-deploy-1fe86]: finalizing version...
✓ hosting[fir-deploy-1fe86]: version finalized
i hosting[fir-deploy-1fe86]: releasing new version...
✓ hosting[fir-deploy-1fe86]: release complete

✓ Deploy complete!

```

Рисунок 3.15 – Веб-сайт успішно завантажений на хостинг

Джерело: запропоноване автором

3.3 Використання інформаційної системи

Перша сторінка, з якої починається робота з веб-сайтом, є сторінка авторизації (рисунок 3.16), на цій сторінці необхідно ввести логін та пароль, який

використовувався під час реєстрації. Ця сторінка визначає роль користувача в системі: адміністратор або читач.

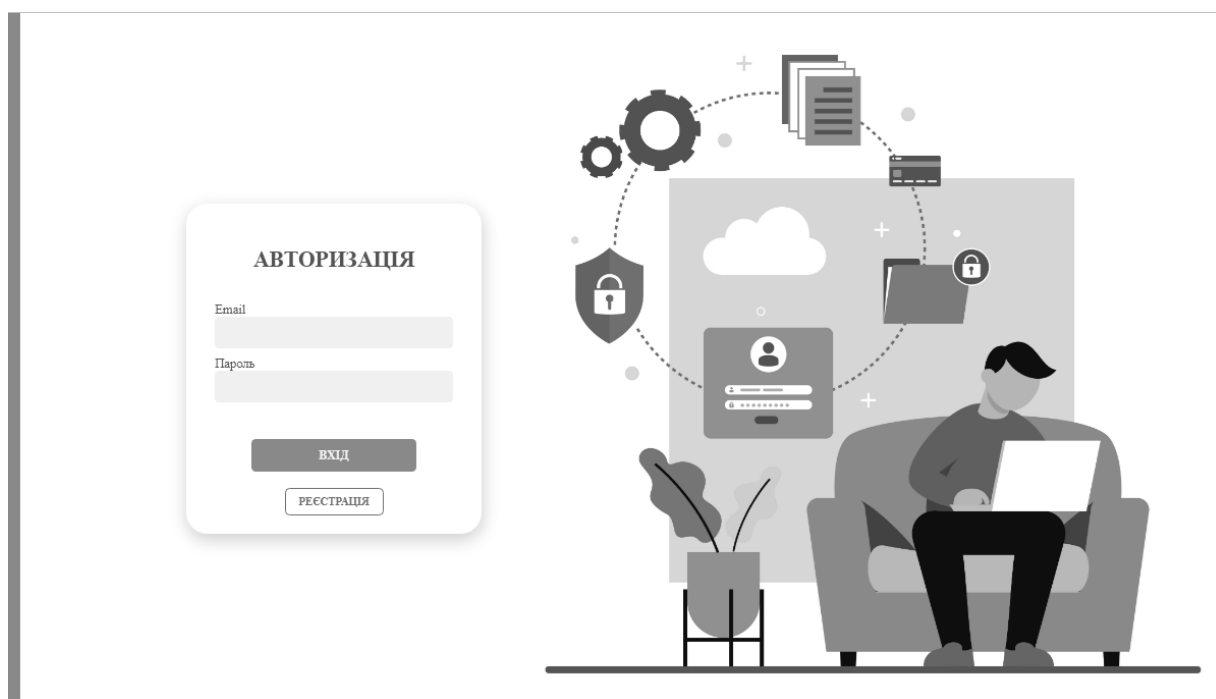


Рисунок 3.16 – Сторінка авторизації

Джерело: запропоноване автором

Якщо користувач вводить неправильну інформацію, то в діалоговому вікні з'являється повідомлення, яке вказує на помилку «Невірний логін або пароль» (рисунок 3.17). Це означає, що введені користувачем дані не співпадають з даними, які були використані під час реєстрації. У такому випадку користувачу рекомендується перевірити правильність введення логіна та пароля і спробувати знову.

У випадку, якщо користувач ввів правильні дані для авторизації, але його обліковий запис ще не було верифіковано адміністратором, він побачить повідомлення «Чекайте підтвердження облікового запису» (рисунок 3.18). Це означає, що користувач успішно надав правильні дані, але їх потрібно перевірити та схвалити адміністратором перед тим, як він зможе отримати повний доступ до функцій сайту. В такому випадку користувачеві рекомендується зачекати на підтвердження його облікового запису адміністратором.

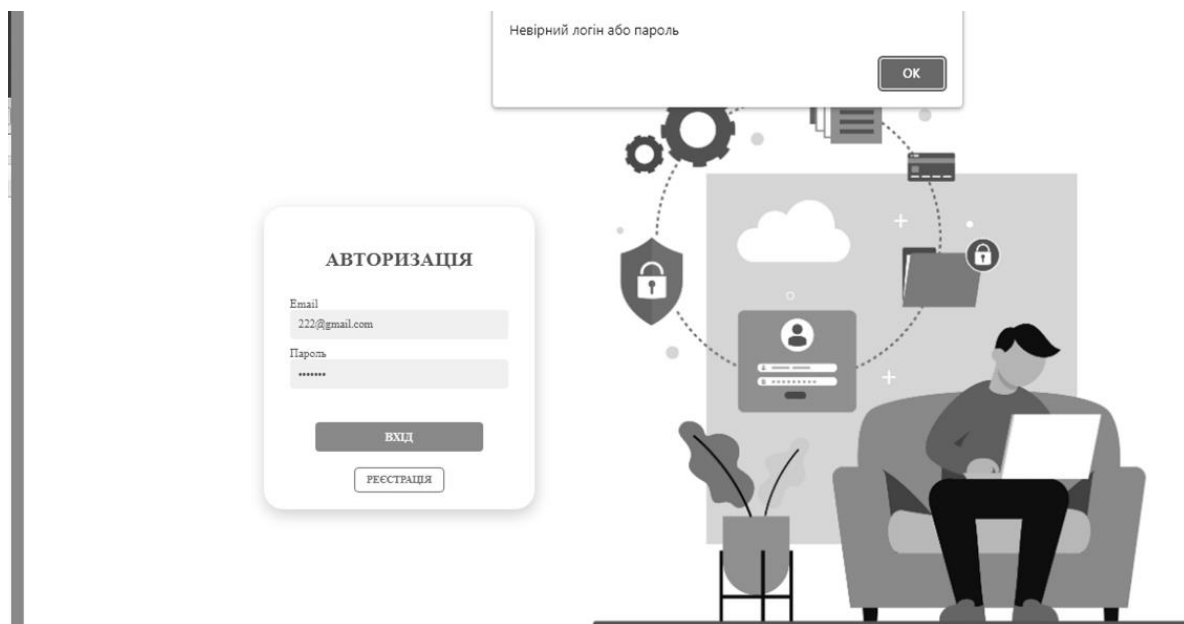


Рисунок 3.17 – Перевірка авторизації

Джерело: запропоноване автором

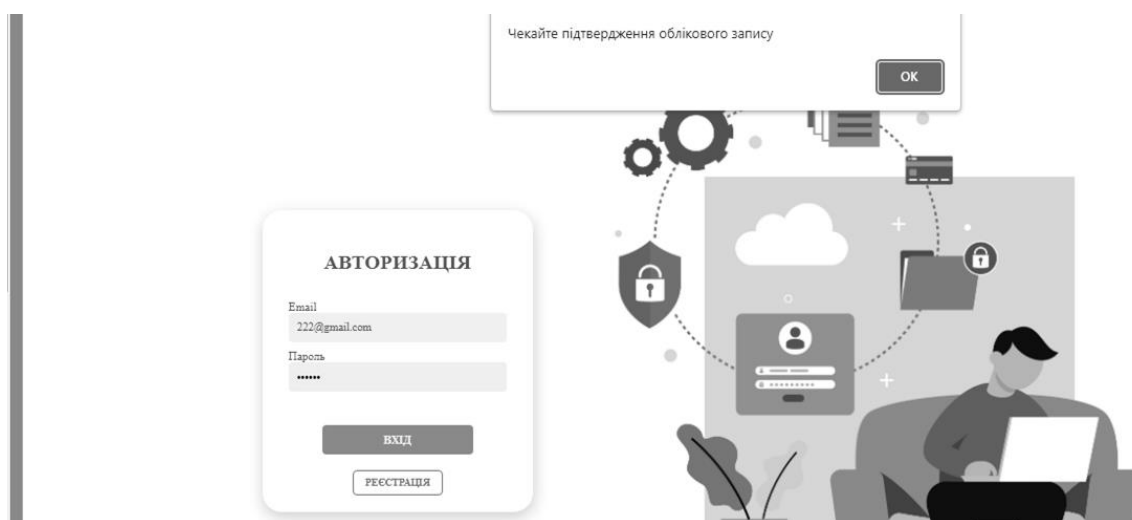


Рисунок 3.18 – Перевірка на верифікацію облікового запису

Джерело: запропоноване автором

Якщо користувач вперше хоче скористатися роботою з веб-сайтом, йому потрібно пройти реєстрації та верифікацію від адміністратора. При реєстрації здійснюється валідація формату електронної поштової скриньки, символів у паролі (кількість символів повинна бути 6 або більше), наявність заповненого ПІБ та

перевірка наявності вже зареєстрованого профілю. У разі невідповідності критеріям перевірки виникне помилка, яка зазначатиме, що саме потрібно виправити (рисунки 3.19, 3.20).



Рисунок 3.19 – Перевірка паролю при реєстрації

Джерело: запропоноване автором

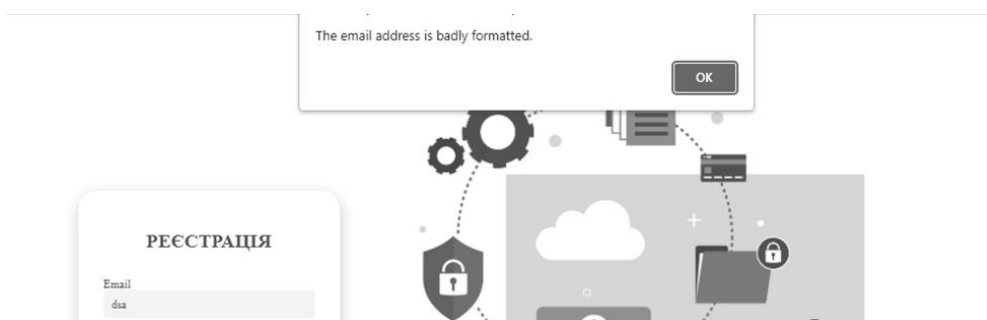


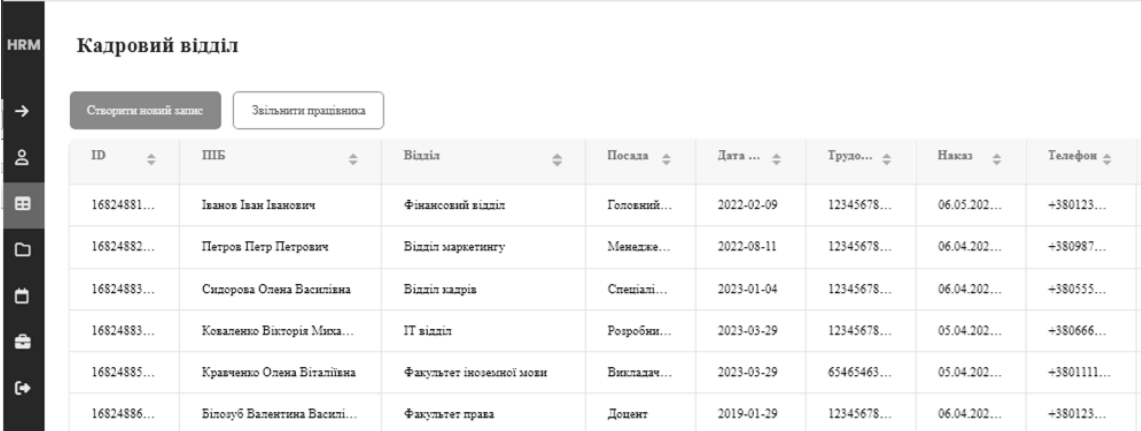
Рисунок 3.20 – Перевірка логіну при реєстрації

Джерело: запропоноване автором

Після реєстрації можна перейти до меню авторизації та ввести створений логін і пароль. При успішній авторизації до веб-сайту користувач буде переправлений на основну сторінку веб-сайту – сторінку кадрового відділу (рисунки 3.21). З даної сторінки користувач може за допомогою меню навігації обирати наступні сторінки, а також за потреби вийти з облікового запису.

З рисунку 3.21 можемо побачити основну сторінку для роботи з кадровим відділом для ролі адміністратор. Роль адміністратора включає в себе функції

створення нової картки працівника, редагування, видалення та звільнення працівника з місця роботи.



Кадровий відділ

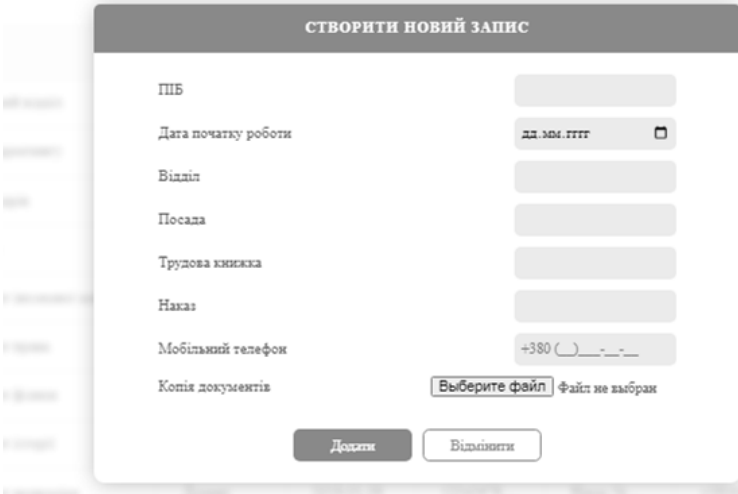
Створити новий запис Звільнити працівника

ID	ПІБ	Відділ	Посада	Дата ...	Трудо...	Наказ	Телефон
16824881...	Іванов Іван Іванович	Фінансовий відділ	Головний...	2022-02-09	12345678...	06.05.202...	+380123...
16824882...	Петров Петр Петрович	Відділ маркетингу	Менеджер...	2022-08-11	12345678...	06.04.202...	+380987...
16824883...	Сидорова Олена Василівна	Відділ кадрів	Спеціаліст...	2023-01-04	12345678...	06.04.202...	+380555...
16824883...	Коваленко Вікторія Мико...	ІТ відділ	Розробник...	2023-03-29	12345678...	05.04.202...	+380666...
16824885...	Кравченко Олена Віталіївна	Факультет іноземної мови	Викладач...	2023-03-29	65465463...	05.04.202...	+3801111...
16824886...	Білозуб Валентина Василі...	Факультет права	Доцент	2019-01-29	12345678...	06.04.202...	+380123...

Рисунок 3.21 – Головна сторінка веб-сайту

Джерело: запропоноване автором

На рисунку 3.22 зображено діалогове вікно, яке створюється при натисканні кнопки «Створити запис». Діалогове вікно включає в себе перевірку на заповнення всіх форм, завантаження файлів, наявність даного користувача вже в базі даних. Якщо користувач вже створений та перебуває в статусі «Працює» або «Звільнено», буде виведено помилку (рисунок 3.23).



СТВОРИТИ НОВИЙ ЗАПИС

ПІБ

Дата початку роботи

Відділ

Посада

Трудова книжка

Наказ

Мобільний телефон

Копія документів

Рисунок 3.22 – Форма для заповнення особової картки працівника

Джерело: запропоноване автором



Рисунок 3.23 – Перевірка на наявність вже створеної картки працівника для цієї особи.

Джерело: запропоноване автором

На рисунку 3.24 зображено календар подій та нотатків. Данна сторінка слугує для планування робочого графіка, створення нотатків при роботі з картками працівників, а також в календарі подій можна відмічати відпустки працівників.

За допомогою комп'ютерної миші ми можемо виділити потрібні дні, до яких ми хочемо додати подію та натиснути на кнопку «Додати подію».

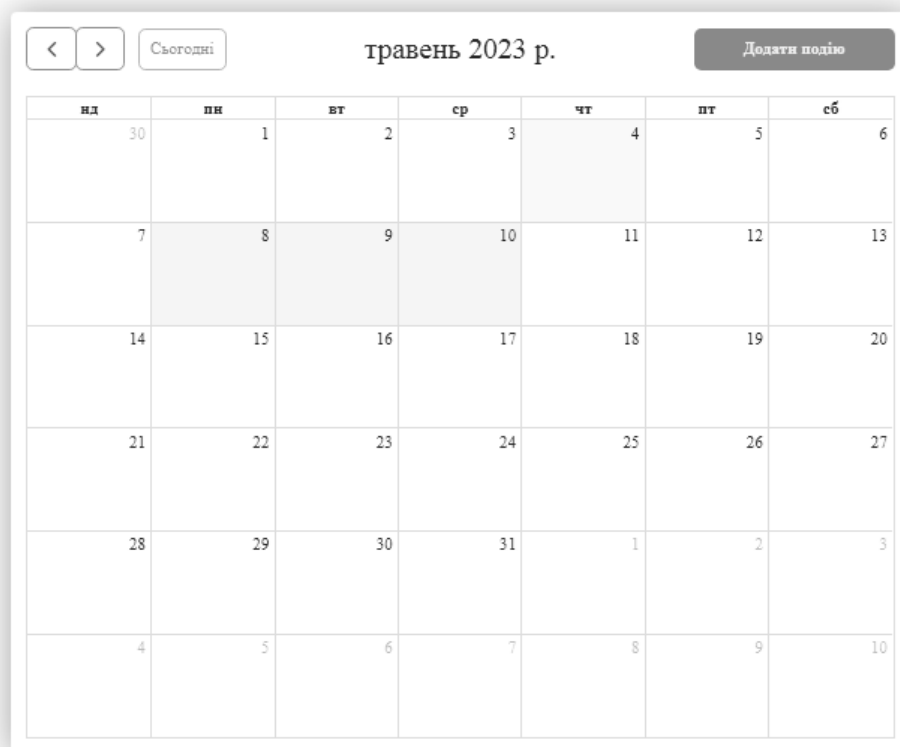


Рисунок 3.24 – Календар подій та нотатків

Джерело: запропоноване автором

Після натискання «Додати подію». з’явиться діалогове вікно в якому потрібно задати назву для події (рисунок 3. 25).

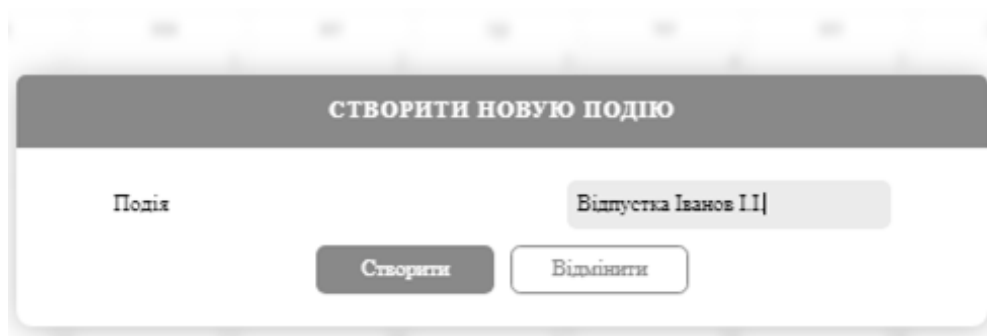


Рисунок 3.25 – Створення назви для події

Джерело: запропоноване автором

У випадку, якщо користувач не обере дату для події, то при заповненні діалогового вікна з’явиться помилка про те, що спочатку потрібно обрати дату (рисунок 3.26)

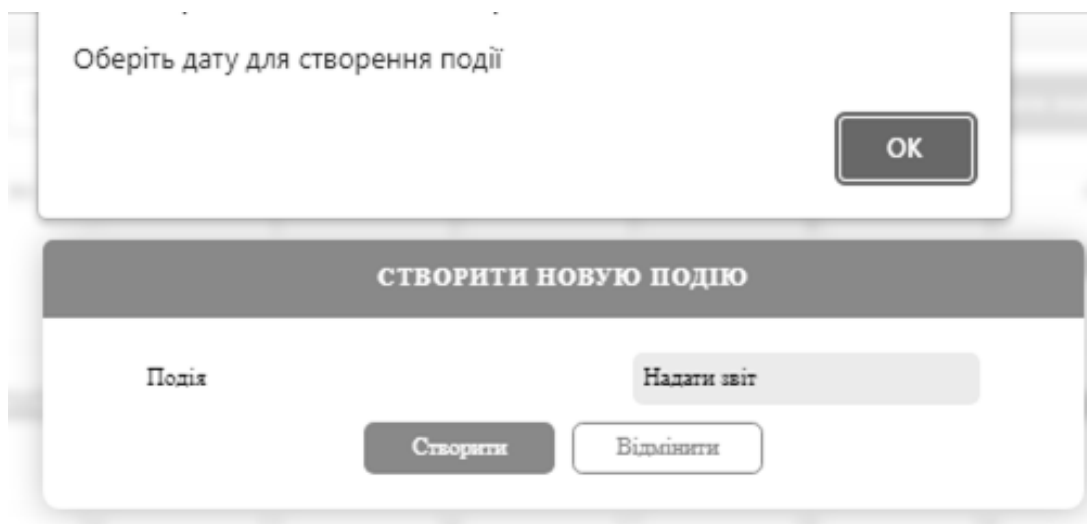


Рисунок 3.26 – Перевірка вибору дати

Джерело: запропоноване автором

Кожна подія створюється різного кольору, який задається автоматично. Приклад заповненого календаря подій наведено на рисунку 3.27

<

>

Сьогодні

травень 2023 р.

Додати подію

нд	пн	вт	ср	чт	пт	сб
30	1	2	3	4	5	6
				Відпустка Сергієва Сергія Сергійовича		
7	8	9	10	11	12	13
Відпустка Сергієва Сергія Сергійовича						
	Відпустка Іванова Івановича			Відпуск Сидоренко Віталій Іванович		
14	15	16	17	18	19	20
Відпустка Сергієва Сергія Сергійовича						
		● Подати звіт 2В				
21	22	23	24	25	26	27
Відпустка Сергієва Сергія Сергійовича						
28	29	30	31	1	2	3
4	5	6	7	8	9	10

Рисунок 3.27 – Заповнений календар подій

Джерело: запропоноване автором

На сторінці «Профіль» для користувача, у якого роль адміністратор створено функцію для верифікації та зміни ролі користувачів (рисунок 3.29).

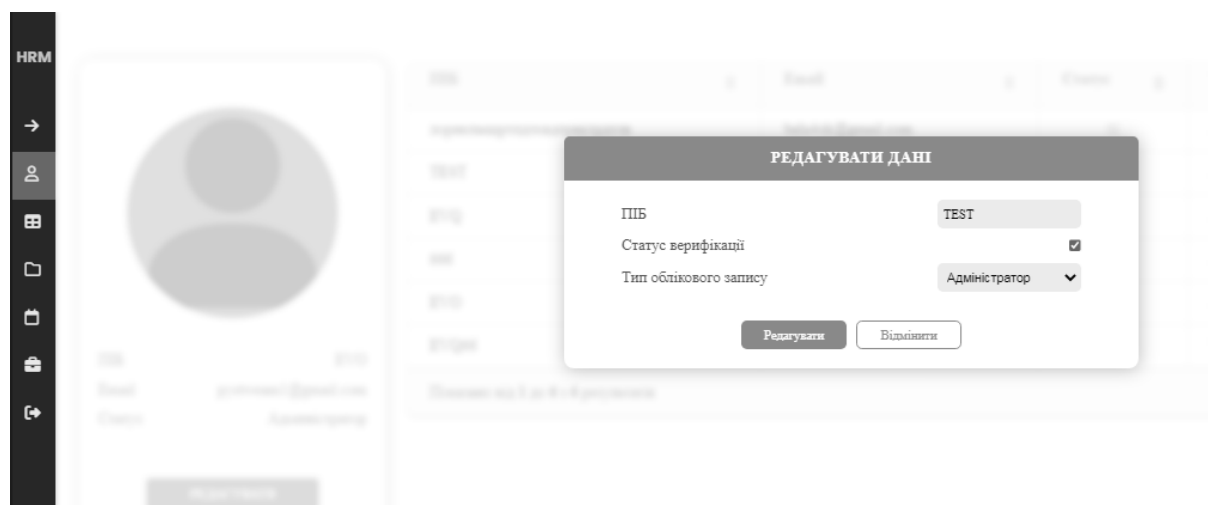


Рисунок 3.28 – Надання статусу та верифікації користувача

Джерело: запропоноване автором

Якщо потрібно видалити працівника або відновити йому пароль, це робить головний адміністратор бази даних за допомогою веб-сайту Firebase. Firebase надає можливість відправки повідомлень на поштову скриньку для верифікації облікових записів, відновлення паролю, зміну поштової скриньки для облікового запису і т.д, приклад шаблону для верифікації облікового запису наведено на рисунку 3.28

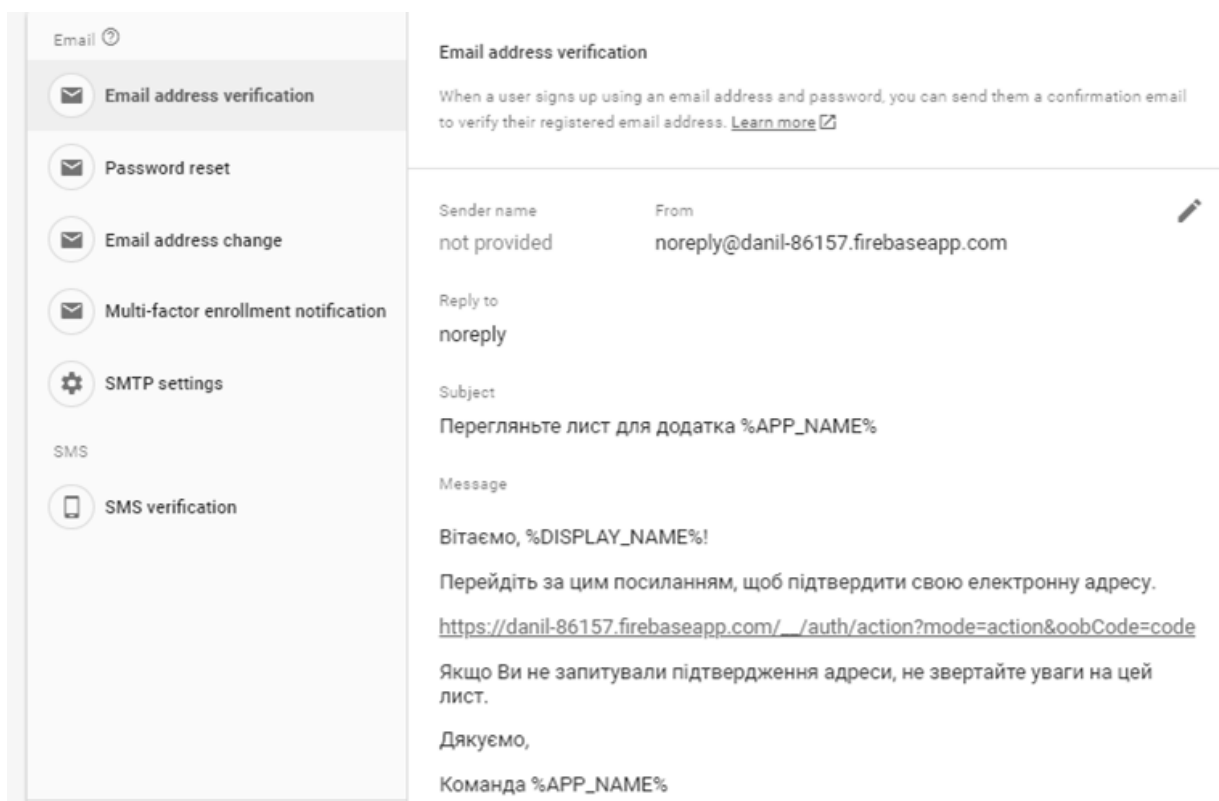


Рисунок 3.28 – Шаблон відправлення повідомлення для відновлення паролю

Джерело: запропоноване автором

ВИСНОВКИ

В результаті виконання кваліфікаційної роботи бакалавра було проведено розробку програмного продукту для інформаційної підсистеми для ВСП «Глухівський агротехнічний коледж СНАУ».

Проведено аналіз альтернативного програмного забезпечення до теми, зроблено порівняння даних і прораховано затрати на місячну підписку на дане ПП. З аналізу витрат на підписку і недоліків наявного ПП було вирішено розробити власний продукт, головною метою було розробити ПП який би мав менше витрат на місяць, аніж продукти аналоги.

Визначено проблему, головну мету та задачі для проєкту. Виходячи з поставленої задачі и проблеми було оглянуто літературні джерела за темами: розробка зручного і сучасного дизайну веб-сайту, розробка логіки для веб-сайту, проектування бази даних.

Для створення ПП було використано базову мову для розмітки сторінки – HTML, мова стилів – CSS, препроцесор для мови стилів – SASS. Препроцесор надає можливість зручніше працювати зі стилями веб-сторінки, особливо коли ми маємо веб-сайт який складається з декількох сторінок та має об'єкти які мають однакові стилі. Для вибору мови логіки було проаналізовано популярні мови програмування на 2023 рік які використовуються для створення FRONT-END і BACK-END додатків. Мовою логіки для веб-сайту було обрано JavaScript, вона є однією з найпопулярніших мов програмування для створення веб-сайтів та веб-додатків.

Базою даних веб-сайту було обрано NoSQL базу даних Firebase від компанії Google. Так як Firebase задовольняє основні потреби для роботи створюваного веб-сайту, то було обрано саме цю базу даних. Firebase надає можливість безкоштовного збереження текстової інформації, файлової інформація, а також містить власну захищену систему авторизації, чим спрощує роботу програміста і підвищує безпеку даних. Firebase використовується у таких програмних продуктах як: Instagram, WhatsApp, Twitter та інших, зазвичай ці ПП мають власні сервери для збереження інформації, а Firebase використовують, як захищену базу даних для авторизації

користувачів. Також однією з головних переваг вибору було те, що Firebase надає можливість безкоштовного хостингу, хоч і безкоштовний хостинг має обмеження, але для роботи кадрового відділу ВСП «Глухівський агротехнічний коледж СНАУ» їх повинно вистачити, у разі чого можна налаштувати тариф підписки під власні потреби.

Для більш детального розуміння зв'язків було побудовано схему бази даних, визначено ключові поля, та зв'язки між таблицями. Окрім схеми бази даних було побудовано діаграму використання IDEF0 яка описує процес створення картки працівника на сайті.

В програмному продукті Figma розроблено макет для розроблюваного веб-сайту. Макет дозволяє програмісту зручніше створювати веб-сайт, знаючи точне положення кожного елемента та його стиль.

За схемою даних було побудовано структуру в Firebase, за створеним макетом було створено структуру проєкту та веб-сторінки. За структурою сайт має на кожну сторінку власний файл HTML, JS та декілька спільних файлів для кожної сторінки, такими файлами є: `firebase.js` – відповідає за підключення бази даних, `authentication.js` – відповідає за перевірку авторизації користувача.

Після розробки веб-сайт було «поставлено» на хостинг від Firebase та протестовано на виявлення помилок.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Minimum system requirements to cast from Chrome URL: <https://support.google.com/chromecast/answer/3209990?hl=en> (дата звернення: 25.05.2023).
2. Мова програмування. URL: https://uk.wikipedia.org/wiki/Мова_програмування (дата звернення: 25.05.2023).
3. Рейтинг мов програмування 2023. URL: <https://dou.ua/lenta/articles/language-rating-2023/> (дата звернення: 25.05.2023).
4. Технічна документація Firebase Database URL: <https://firebase.google.com/docs/database/web/start> (дата звернення: 25.05.2023).
5. Технічна документація Firebase Storage. URL: <https://firebase.google.com/docs/storage> (дата звернення: 25.05.2023).
6. Технічна документація Firebase Hosting. URL: <https://firebase.google.com/docs/hosting> (дата звернення: 25.05.2023).
7. Технічна документація Figma. URL: <https://help.figma.com/hc/en-us> (дата звернення: 25.05.2023).
8. Технічна документація SASS. URL: <https://sass-lang.com/documentation/> (дата звернення: 25.05.2023).
9. Мова програмування JavaScript. URL: <https://developer.mozilla.org/en-US/docs/Web/JavaScript> (дата звернення: 25.05.2023).
10. CSS Documentation. URL: <https://devdocs.io/css/> (дата звернення: 25.05.2023).
11. Редактор коду Visual Studio Code. URL: <https://code.visualstudio.com/docs>
12. Огляд продукту HurmaSystem. URL: <https://hurma.work/capabilities/hr/> (дата звернення: 25.05.2023).
13. Огляд продукту PeopleForce URL: <https://peopleforce.io/uk/products/peoplehr> (дата звернення: 25.05.2023).
14. Архітектура програмного забезпечення URL: https://uk.wikipedia.org/wiki/Архітектура_програмного_забезпечення

15. Басанець Д.В. ІНФОРМАЦІЙНА ПІДСИСТЕМА ОБЛІКУ КАДРОГО ЗАБЕЗПЕЧЕННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ КОЛЕДЖУ ВСП «ГАФК СНАУ» / Технології ХХІ сторіччя: Збірник тез за матеріалами 28-ої міжнародної науково-практичної конференції (23-25 листопада 2022 р.). Ч.2. – Суми: СНАУ, 2022 - с.7.

16. Басанець Д.В. ІНФОРМАЦІЙНА ПІДСИСТЕМА ОБЛІКУ КАДРОГО ЗАБЕЗПЕЧЕННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ КОЛЕДЖУ ВСП «ГАФК СНАУ» / Матеріали науково-практичної конференції викладачів, аспірантів та студентів Сумського НАУ (25-28 квітня 2023 р.). – Суми, 2023. – с.356.

ДОДАТКИ

ДОДАТОК А

ТЕХНІЧНЕ ЗАВДАННЯ

**На розробку інформаційної підсистеми обліку кадрового забезпечення
інформаційної системи коледжу ВСП «ГАФК СНАУ»**

Погоджено:

Кандидат технічних наук, доцент кафедри
кібернетики та інформатики

Студент групи ІСТ 2101 с.т.

Басанець Данило Віталійович

СУМИ 2023

1. Призначення й мета створення веб-додатку

1.1 Призначення веб-додатку

Інформаційна система має автоматизувати роботу кадрового відділу ВСП «Глухівський агротехнічний фаховий коледж СНАУ» та підвищити взаємозв'язок між відділами підприємства.

1.2 Мета створення веб-додатку

Автоматизація роботи кадрового відділу ВСП «Глухівський агротехнічний фаховий коледж СНАУ».

1.2.1 Цільова аудиторія

До цільової аудиторії веб-додатку можна віднести працівників підприємства ВСП «Глухівський агротехнічний фаховий коледж СНАУ», а саме: працівників кадрового відділу підприємства, начальників інших відділів, працівників Єдиної державної електронної бази освіти, завідувачів відділень.

2 Вимоги до веб-додатку

2.1 Вимоги до веб-додатку в цілому

2.1.1 Вимоги до структури й функціонування веб-додатку

Веб-додаток повинен складатися із взаємозалежних розділів із чітко розділеними функціями, мати єдиний інтерфейс оформлення та має бути доступним в мережі Інтернет.

2.1.2 Вимоги до персоналу

Від персоналу не має вимагатися особливих технічних навичок для підтримки й експлуатації веб-додатку, окрім загальних навичок роботи з персональним комп'ютером і стандартним веб-браузером.

2.1.3 Вимоги до збереження інформації

Уся інформація надана у веб-додатку буде зберігатися у хмарній базі даних Firebase.

2.1.4 Вимоги до розмежування доступу

Веб-додаток, який розроблюється, не повинен бути загальнодоступним.

Залежно від прав доступу до інформації в веб-додатку, користувачі можуть бути розділені на відвідувачів та адміністраторів.

Відвідувачі мають змогу переглядати всі сторінки веб-додатку, але не можуть створювати, редагувати, видаляти будь які дані.

Адміністратор має повний доступ до всіх функцій веб-додатку. Він може редагувати, додавати та видаляти картки працівників, надавати статус верифікації іншим користувачам та призначати рівень доступу.

2.2 Структура веб-додатку

2.2.1 Загальна інформація про структуру веб-додатку

Структура веб-додатку являє собою набір сторінок, які також є пунктами головного меню.

Такими розділами є:

Профіль - це сторінка, на якій можна знайти інформацію про користувача. Для адміністратора веб-додатку доступний список користувачів.

Кадровий відділ – головна сторінка на якій відображена інформація про кадровий відділ підприємства.

Редагування файлів – це сторінка, яка дозволяє редагувати файли вже створених карток працівників.

Календар подій – це сторінка, яка використовується для відмічання відпусток та інших нотаток працівників кадрового відділу.

Звільнені працівники – інформація про працівників, які були звільнені, а також причина їх звільнення.

2.2.2 Навігація

Відповідно до бажаного замовником дизайну веб-додатку, для навігації, у шапці буде створена система контент меню. Меню необхідне для швидкого переміщення користувача по усім доступним сторінкам. Меню буде відображатися на всіх сторінках, щоб користувач міг в будь-який момент часу перейти на будь-яку сторінку веб-додатку.

2.2.3 Наповнення веб-додатку (контент)

Заповнення та редагування контенту веб-додатку має робити користувач з типом доступу адміністратора.

2.2.4 Дизайн та структура додатку

Стиль веб-додатку має бути сучасним, приємним для сприйняття, у якості основних кольорів було запропоновано використати сині та помаранчеві кольори.

Розташування елементів на головній сторінці веб-додатку схематично показано на рисунку А.1.

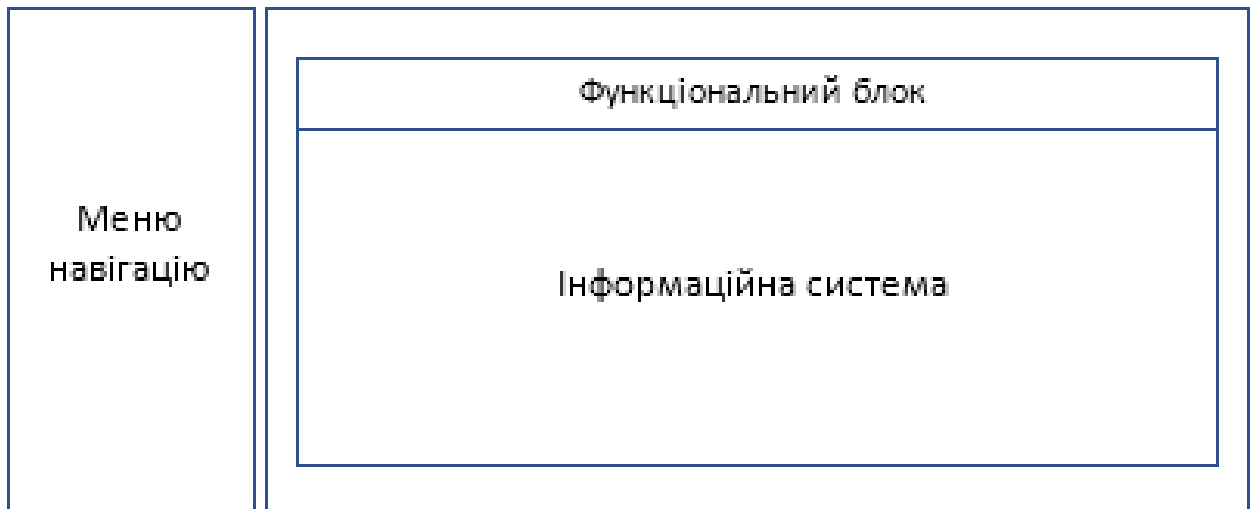


Рисунок А.1 – Схема головної сторінки

2.2.5 Система навігації (карта веб-додатку)

Карта веб-додатку зображена на рисунку А.2.

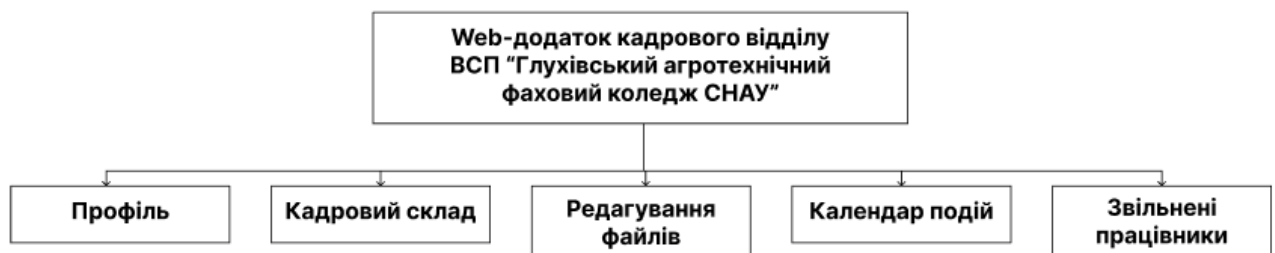


Рисунок А.2 – Карта веб-додатку

2.3 Вимоги до функціонування системи

2.3.1 Потреби користувача

Потреби користувача, визначені на основі рішення замовника, представлені у таблиці А.1.

Таблиця А.1 – Потреби користувача

ID	Потреби користувача	Джерело
UN-01	Можливість створення списку карток працівників	Адміністратор
UN-02	Можливість редагування картки працівника	Адміністратор
UN-03	Видалення картки працівника	Адміністратор
UN-04	Перегляд звільнених працівників	Адміністратор
UN-05	Редагування файлів	Адміністратор, відвідувач
UN-06	Пошук карток працівників	Адміністратор, відвідувач
UN-07	Можливість редагування типу доступу для користувачів	Адміністратор
UN-08	Можливість надання статусу верифікація користувачів	Адміністратор
UN-09	Можливість переглядати хто останній зробив зміни до запису	Адміністратор

2.3.2 Функціональні вимоги

На основі потреб користувача були визначені такі функціональні вимоги:

- реєстрація та авторизація користувачів;
- пошук карток працівників;
- відображення результатів пошуку на веб-сторінці;
- можливість редагування файлів;
- створення карток працівників;
- редагування карток працівників;
- видалення карток працівників;
- створення календаря подій та нотаток;
- адміністрування користувачів.

2.3.3 Системні вимоги

Даний розділ визначає, розподіляє та вказує на системні вимоги, визначені розробником. Їх перелік наведений в таблиці А.2.

Таблиця А.2 – Системні вимоги

ID	Системні вимоги	Пріоритет	Опис
SR-01	Наявність модуля з картками працівників	М	Надає можливість переглядати і створювати картки працівників.
SR-02	Меню для редагування карток працівників	М	Формує модальне вікно для редагування.
SR-03	Меню для видалення картки працівника	М	Формує модальне вікно для видалення.
SR-04	Наявність модуля у якому відображаються усі звільнені працівники	S	Відповідає за відображення списку звільнених працівників і відображає причину звільнення .
SR-05	Модуль для редагування файлів	S	Надає можливість редагувати файл який вже підв'язаний до картки працівника.
SR-06	Меню пошуку карток працівників	C	Надає можливість шукати за ПІБ працівника в базі даних.
SR-07	Модуль роботи з користувачами	М	Надає можливість для верифікації користувачів та надання прав.
SR-08	Відображення останніх змін	C	Модуль який вказує хто з користувачів останній вніс зміни до запису.

Умовні позначення в таблиці А.2:

Must have (M) – вимоги, які повинні бути реалізовані в системі;

Should have (S) – вимоги, які мають бути виконані, але вони можуть почекати своєї черги;

Could have (C) – вимоги, які можуть бути реалізовані, але вони не є центральною ціллю проекту.

2.4 Вимоги до видів забезпечення

2.4.1 Вимоги до інформаційного забезпечення

Реалізація веб-додатку відбувається з використанням:

- JavaScript
- Firebase

2.4.2 Вимоги до лінгвістичного забезпечення

Веб-додаток має бути виконаний українською мовою.

2.4.3 Вимоги до програмного забезпечення

Програмне забезпечення клієнтської частини повинне задовольняти наступним вимогам: Веб-браузер: Internet Explorer 7.0 і вище, або Firefox 3.5 і вище, або Opera 9.5 і вище, або Safari 3.2.1 і вище, або Chrome 2 і вище.

3 Склад і зміст робіт зі створення веб-додатку

Докладний опис етапів роботи зі створення веб-додатку наведено в таблиці А.3.

Таблиця А.3 – Етапи створення веб-додатку

№	Склад і зміст робіт	Строк розробки (у робочих днях)
1	Обґрунтування проекту	1 день
3	Розробка договору на створення сайту	1 день
4	Створення макету для веб-ресурсу	3 дні
5	Узгодження макету	1 день

Продовження таблиці А.3

6	Розробка програмного коду HTML, SASS	4 дні
7	Створення логіки вводу-виводу	6 днів
8	Створення логіки видалення записів	4 дні
9	Створення логіки редагування записів	5 днів
11	Оптимізація коду	5 днів
	Загальна тривалість робіт	46 днів

4. Вимоги до складу й змісту робіт із введення веб-додатку в експлуатацію

Для того, щоб веб-додатком могли користуватися користуватися усі користувачі підприємства які мають на це доступ, необхідно розмістити його у мережі Інтернет. На хостинг переноситься веб-додаток і наповнення бази даних з подальшою їх доробкою. Для коректного переносу веб-додатку на хостинг необхідно, щоб параметри хостинга відповідали вимогам, зазначеним у ТЗ.

ДОДАТОК Б

ПЛАНУВАННЯ РОБІТ

Б.1 Ідентифікація ідеї проекту

Метою даного проекту є покращення ефективності кадрового відділу шляхом автоматизації його процесів. Це дозволить зменшити ручну роботу, спростити багато рутинних операцій і забезпечити більш точний та швидкий доступ до необхідної інформації. Крім того, використання програмного продукту дозволить забезпечити збереження всієї необхідної інформації в електронному вигляді, що сприятиме зручності її зберігання, пошуку та аналізу.

Програмний продукт буде реалізований у формі веб-сайту, який буде доступний через мережу Інтернет. Це дозволить користувачам зручно працювати з системою з будь-якого місця, де є доступ до Інтернету. Веб-сайт буде мати інтуїтивно зрозумілий і легкий у використанні інтерфейс, що спростить роботу з ним навіть для непрофесійних користувачів.

Одним із важливих аспектів розробки буде забезпечення високого рівня безпеки. Програмний продукт повинен мати захист від несанкціонованого доступу до конфіденційної інформації про співробітників та інші важливі дані. Будуть застосовані відповідні механізми аутентифікації та авторизації, шифрування даних та інші заходи для забезпечення безпеки інформації.

Б.2 Деталізація мети проекту методом SMART

SMART - це аббревіатура, що означає конкретні, вимірювані, досяжні, релевантні та часові обмеження для поставлених цілей. Розглянемо результати деталізації методом.

Продуктом дипломного проекту є створена ІС для збереження інформації про працівників ВСП «Глухівський агротехнічний коледж СНАУ». Результати деталізації мети методом SMART розміщені у таблиці Б.1.

Таблиця Б.1 – Деталізація мети методом SMART

Specific (конкретна)	Створити якісний програмний продукт для кадрового відділу, використовуючи мінімум ресурсів.
Measurable (вимірювана)	Щонайменше вдвічі підвищити ефективність роботи з системою (оцінка проводиться користувачами системи)
Achievable (досяжна)	Реалізація веб-сайту для кадрового відділу можлива за допомогою використання HTML, CSS, JS.
Relevant (актуальна)	У наявності є всі необхідні технічні та програмні засоби. Розробники достатньо кваліфіковані для виконання поставлених задач.
Time-bound (обмежена у часі)	Ціль має часові обмеження. Робота повинна бути виконана у терміни, що були оговорені замовником проекту. Проект повинен бути виконаний згідно з календарним планом.

Б.3 Планування змісту структури робіт ІТ-проекту

Ієрархічна структура робіт (Work Breakdown Structure) є інструментом, що дозволяє розбити проект на складові частини. Вона встановлює ієрархічну структуру поділу праці для всіх працівників, які беруть участь у проекті.

Під час створення WBS, проект послідовно розбивається на підпроекти, роботи різних рівнів та детальні робочі пакети. Розподіл включає розподіл результатів проекту на менші, керовані компоненти пакетів. Зазвичай пакети робіт відповідають найнижчому рівню деталізації і складаються з окремих завдань. Декомпозиція повинна бути коректною, тобто елементи на будь-якому рівні WBS повинні бути необхідними та достатніми для створення відповідного елемента на вищому рівні.

Ієрархічна структура робіт, в основному, є переліком завдань проекту. Вона може бути представлена у графічній формі або у вигляді опису

Виконаємо побудову WBS структури, у якій зазначимо всі виконувані роботи в залежності від головних етапів. Діаграма WBS зображена на рисунку Б.1.

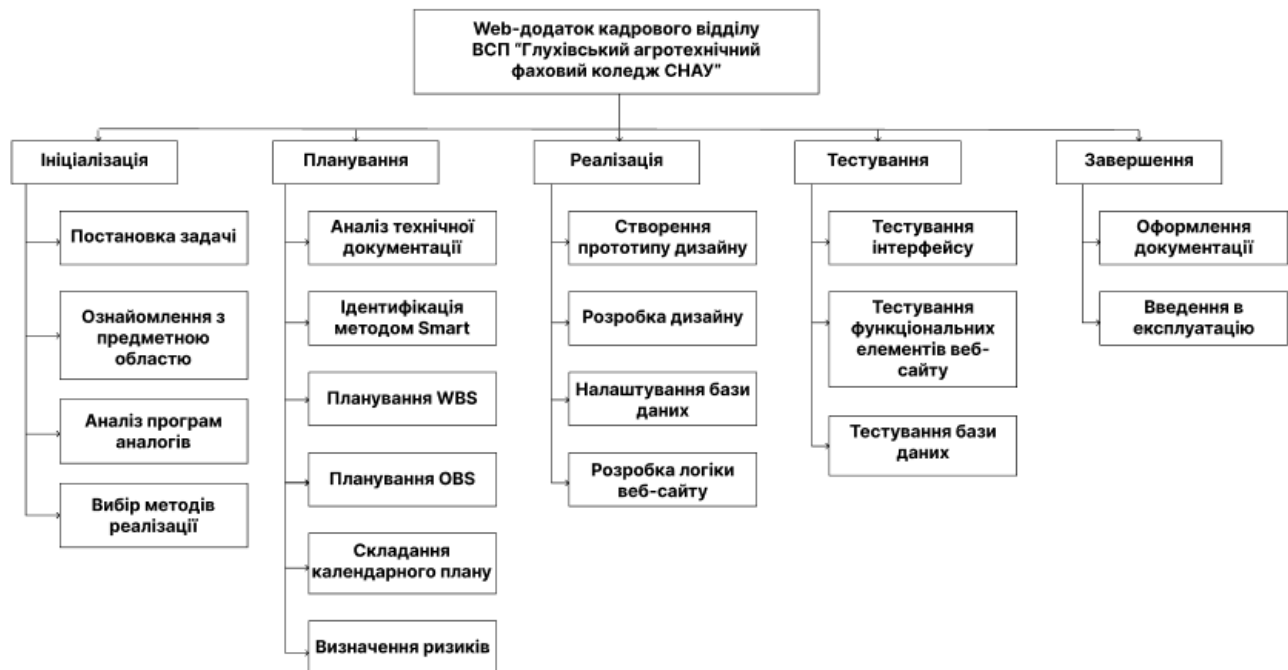


Рисунок Б.1 – Таблиця WBS

Б.4 Планування структури організації готового проекту

Організаційна структура проекту (OBS) - це структура виконавців (організацій), яка визначається на основі переліку пакетів робіт нижнього рівня структури розбиття робіт (WBS). Вона представляється відповідальними особами за виконання цих пакетів робіт. Відповідальні особи - це люди, які безпосередньо організують виконання робіт, і це не обов'язково керівники організацій або відділів.

Організаційна структура представляє собою графічне відображення учасників проекту та їх відповідальних осіб, які беруть участь у реалізації проекту. На верхньому рівні OBS знаходиться команда проекту. На наступному рівні вказуються виконавці, такі як організації або відділи. На рівні нижче для кожного виконавця вказуються конкретні особи, які будуть відповідальні за виконання елементарних робіт WBS. Діаграма OBS зображена на рисунку Б.2.

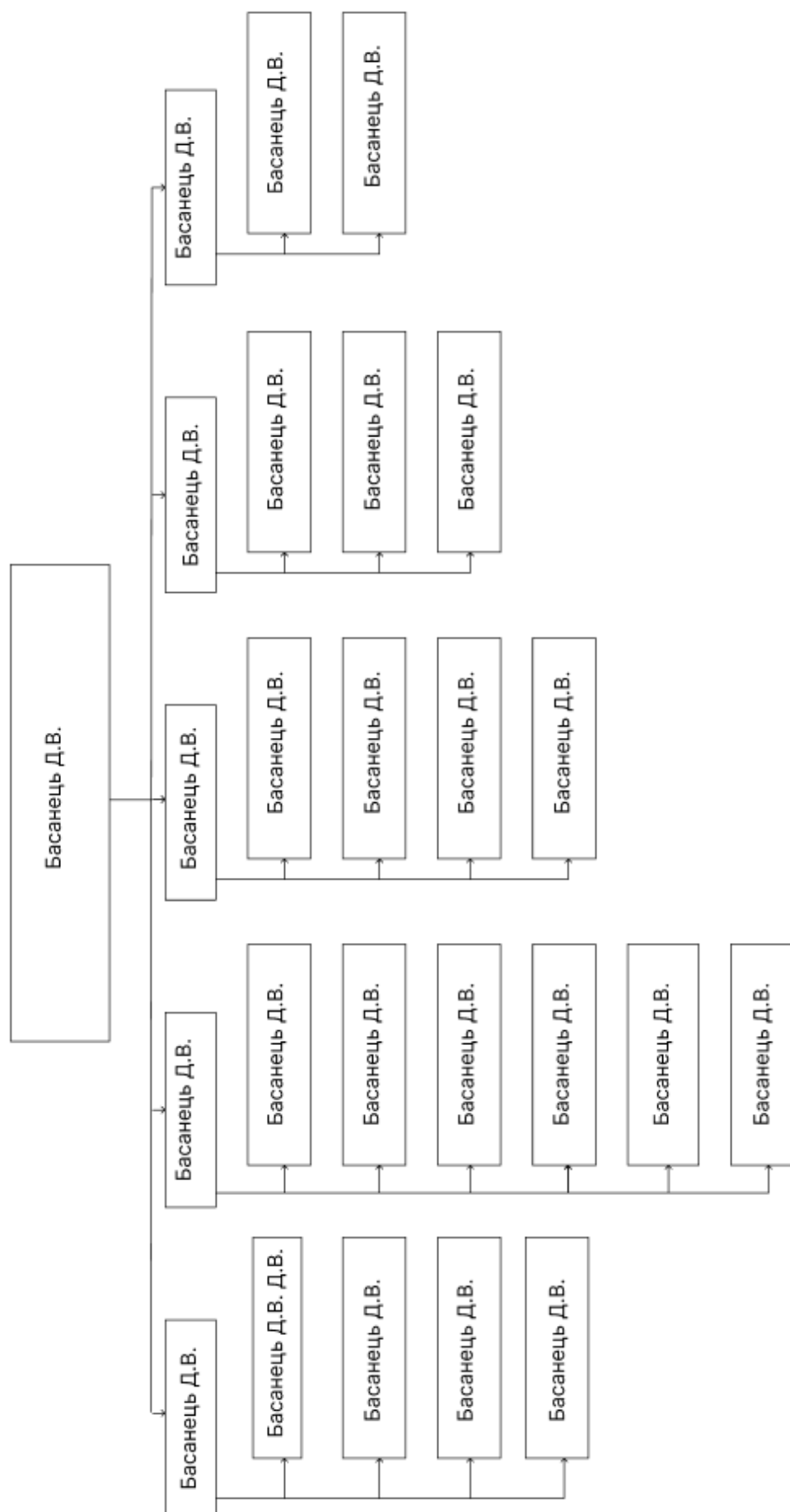


Рисунок Б.2 – Таблица OBS

Б.5 Побудована календарного графіку виконання ІТ-проекту

Для формування розкладу проекту використовується діаграма Ганта, яка є надійним інструментом управління проектами. Ця діаграма допомагає в плануванні та створенні розкладу проектів будь-якого масштабу. На рисунку Б.3 показана діаграма Ганта, яка була створена за допомогою програми Microsoft Project.

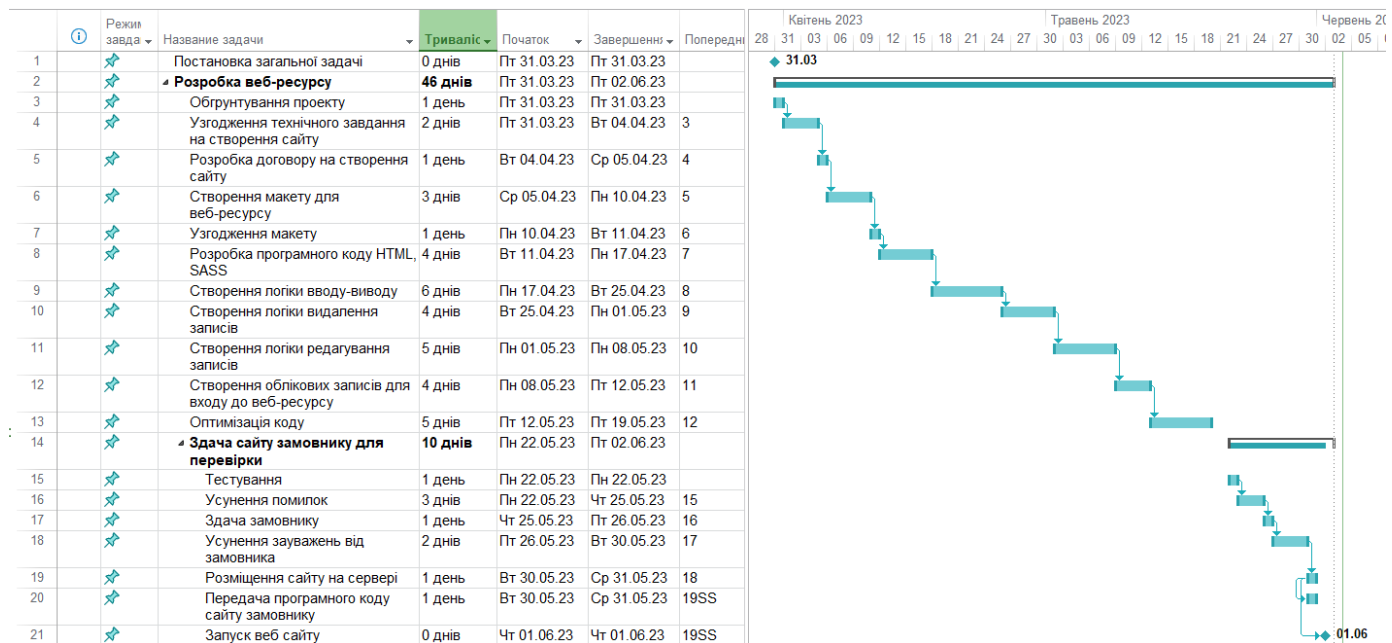


Рисунок А.3. – Діаграма Ганта

А.6 Ідентифікація ризиків

Проведемо оцінку ризиків роботи з точки зору якості та кількості. Під час якісної оцінки виявимо ті ризики, на які потрібно негайно реагувати. Ця оцінка визначить важливість ризику та допоможе обрати відповідний метод реагування. Кількісна оцінка ризиків буде здійснена для більш повного визначення ризиків та їх впливу на виконання проекту. Кількісна та якісна оцінка ризиків можуть використовуватися окремо або разом, залежно від доступного часу та бюджету, а також від потреби у кількісній або якісній оцінці ризиків.

Ризик – це ймовірнісна подія, яка у випадку своєї появи негативно або позитивно впливає на проект.

Управління ризиком – це процес реагування на події та зміни ризиків у процесі виконання проекту. При цьому важливим є проведення моніторингу ризиків.

Перелічимо можливі ризики проекту, визначимо відносну критичність, а також можливі способи їх уникнення. Введемо умовну шкалу рівнів критичності ризиків (табл. Б.2):

Таблиця Б.2 – Рівні критичності ризиків

1	Незначний
2	Низький
3	Середній
4	Високий
5	Критичний

Ризики проекту, визначені в результаті ідентифікації, а також їх потенційний вплив та ймовірність виникнення представлені у таблиці Б.3.

Таблиця Б.3 – Ризики проекту

№	Опис	Ймовірність	Вплив
1	Втрата актуальності проекту	1	4
2	Втрата фінансування проекту	3	5
3	Недостатній рівень кваліфікації розробників	1	4
4	Вимушене зміщення термінів виконання завдань або паралельне виконання цих завдань	3	3
5	Невідповідність вимогам замовника	3	4
6	Пошкодження або відсутність обладнання	1	5
7	Проблеми зі сумісністю на різних пристроях	2	4
8	Невідповідність вимогам замовника	2	4

З отриманих даних в таблиці Б.3, складемо матрицю залежності впливу від ймовірності виникнення кожного із зазначених ризиків (табл. Б.4).

Таблиця Б.4 – Матриця «Ймовірність – Влив»

Ймовірність \ Вплив	1	2	3	4	5
1				1,3	6
2				7,8	
3			4	5	2
4					
5					

ДОДАТОК В

ДОВІДКА ПРО ВИКОРИСТАННЯ РЕЗУЛЬТАТІВ ДИПЛОМНОЇ РОБОТИ

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СУМСЬКИЙ НАЦІОНАЛЬНИЙ АГРАРНИЙ
УНІВЕРСИТЕТ

Відокремлений структурний підрозділ
«Глухівський агротехнічний фаховий
коледж Сумського національного
аграрного університету»

вул. Терешенків, 36, м. Глухів, Сумська область
Україна, 41400, Тел./факс +38(05444) 23552
e-mail: hlukhiv_ksnau@ukr.net, СДРПОУ 44267623



MINISTRY OF EDUCATION AND SCIENCE OF UKRAINE
SUMY NATIONAL AGRARIAN
UNIVERSITY

Separate Structural Subdivision
"Hlukhiv Agrotechnical Professional
College of Sumy National
Agrarian University"

Tereshchenkiv Street, 36, Hlukhiv, Sumy Oblast
Ukraine, 41400, Tel./fax +38(05444) 23552
e-mail: hlukhiv_ksnau@ukr.net, СДРПОУ 44267623

Від 22.06.2023 № 40.1-12/359

На № _____ від _____

ДОВІДКА

про впровадження результатів
дипломної роботи бакалавра
Басанця Данили Віталійовича

Виконана студентом факультету економіки та управління 126 «Інформаційні системи та технології» Сумського національного аграрного університету Басанець Д.В. дипломна робота на тему "Розробка інформаційної підсистеми обліку кадрового забезпечення інформаційної системи коледжу ВСП «ГАФК СНАУ»" має практичну значущість і розглянута щодо впровадження у роботу кадрового відділу ВСП «Глухівський агротехнічний фаховий коледж Сумського національного аграрного університету».

Директор ВСП «Глухівський
агротехнічний фаховий
коледж Сумського
національного аграрного
університету»



Анатолій ЛИТВИНЕНКО

ДОДАТОК Г
КОПІЇ ПУБЛІКАЦІЙ



Сумський національний аграрний університет



Національний технічний університет «ХП»



Політехніка Свентокржинська в Кельцах (Польща)



ТОВ «ТРІЗ»



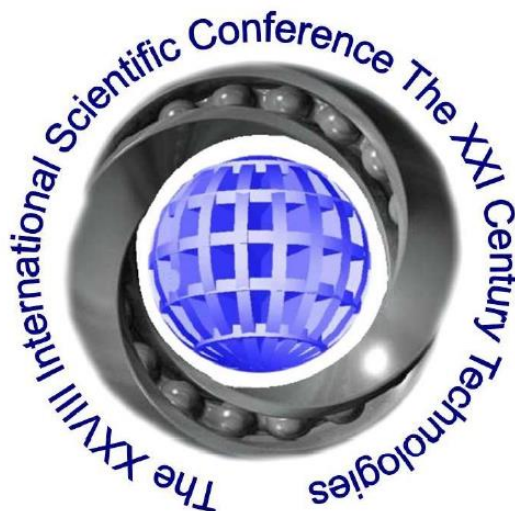
Сумський державний університет



Державний біотехнологічний університет



Українська технологічна академія



ТЕХНОЛОГІЇ ХХІ СТОРІЧЧЯ

Збірник тез за матеріалами 28^{ої} міжнародної науково-практичної конференції
(23-25 листопада 2022 р.)

Частина 2

Секції: «Прогресивні технології в економіці»,
«Прогресивні технології в екології та енергоефективність»,
«Прогресивні технології у педагогіці»

Суми – 2022

ЗМІСТ

СЕКЦІЯ «ПРОГРЕСИВНІ ТЕХНОЛОГІЇ В ЕКОНОМІЦЬ»

<i>Самошкіна І.Д., доцент, СНАУ, м. Суми, Україна</i> МІЖНАРОДНИЙ ДОСВІД РОЗВИТКУ АГРАРНОГО СТРАХУВАННЯ В УКРАЇНІ.....	3
<i>Омельчук А.І., магістрант, Агаджанова С.В., к.т.н., доцент, Суми, СНАУ, Україна</i> ЗАКОНОДАВЧІ ЗАСАДИ ФУНКЦІОНУВАННЯ СИСТЕМИ НАДАННЯ СОЦІАЛЬНИХ ПОСЛУГ ДЛЯ СОЦІАЛЬНИХ ГРУП НАСЕЛЕННЯ.....	5
<i>Омельчук А.І., магістрант, Агаджанова С.В., к.т.н., доцент, Суми, СНАУ, Україна</i> ВДОСКОНАЛЕННЯ СИСТЕМИ НАДАННЯ СОЦІАЛЬНИХ ПОСЛУГ СПЕЦІАЛЬНИМ ГРУПАМ НАСЕЛЕННЯ ЗАСОБАМИ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ.....	6
<i>Басанець Д.В., студент, СНАУ, Суми, Україна</i> ІНФОРМАЦІЙНА ПІДСИСТЕМА ОБЛІКУ КАДРОВОГО ЗАБЕЗПЕЧЕННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ КОЛЕДЖУ ВСП «ГАФК СНАУ».....	7
<i>Нечипоренко В.В., к.с.-г.н., доцент, СНАУ</i> СТРАТЕГІЧНІ НАПРЯМИ СТАЛОГО РОЗВИТКУ АГРАРНОГО СЕКТОРУ ЕКОНОМІКИ.....	8
<i>Шакоотько А.М., студент, Толбатов А.В., к.т.н., доцент, СНАУ, Суми, Україна</i> ІНФОРМАЦІЙНА ПІДСИСТЕМА МАТЕРІАЛЬНОГО ЗАБЕЗПЕЧЕННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ ВСП «ГАФК СНАУ».....	9
<i>Шакоотько Є.С., молодший спеціаліст, Сумський національний аграрний університет</i> СУЧАСНІ НАПРЯМКИ СТРАХУВАННЯ КІБЕРРИЗИКІВ.....	11
<i>Білик Г.М., магістрант, Сумський національний аграрний університет, Україна</i> ОСОБЛИВОСТІ ЗЕМЛЕКОРИСТУВАННЯ У РОЗВИНЕНИХ КРАЇНАХ СВІТУ В ЧАСТИНІ ЕКСПЛУАТАЦІЇ СІЛЬСЬКОГОСПОДАРСЬКИХ УГІДЬ.....	12
<i>Грошовик Р.Ю. студент, Пасько Н.Б., доцент, Сумський НАУ</i> АВТОМАТИЗАЦІЯ ПОГОСПОДАРСЬКОГО ОБЛІКУ ТЕРИТОРІАЛЬНИХ ГРОМАД.....	15
<i>Кальченко Я., магістрант, СНАУ, Суми, Україна</i> ОЦІНКА ЕФЕКТИВНОСТІ ЗБУТОВОЇ ДІЯЛЬНОСТІ АГРАРНОГО ПІДПРИЄМСТВА.....	19
<i>Колеснікова Ю.С., магістрант, Сумський національний аграрний університет, Україна</i> ОСОБЛИВОСТІ ФОРМУВАННЯ МАРКЕТИНГОВОЇ СТРАТЕГІЇ ПЕРСОНАЛУ У ВЕТЕРИНАРНИХ КЛІНІКАХ.....	21
<i>Колосов Р.В., магістрант, Сумський національний аграрний університет, Україна</i> ПРОБЛЕМИ ПРОГНОЗУВАННЯ ВИКОРИСТАННЯ ЗЕМЕЛЬНИХ РЕСУРСІВ В УКРАЇНІ.....	24
<i>Колпаков В.В., магістрант, Сумський національний аграрний університет, Україна</i> МАРКЕТИНГ ВІДНОСИН ЯК ВАЖЛИВИЙ ЕЛЕМЕНТ КОНКУРЕНТНОЇ МАРКЕТИНГОВОЇ СТРАТЕГІЇ.....	26
<i>Лисенко К., магістрант, Сумський національний аграрний університет, Україна</i> МЕТОДОЛОГІЯ ТА АЛГОРИТМ ПРОВЕДЕННЯ ЗЕМЕЛЬНИХ ТОРГІВ.....	29
<i>Макарова В.В., д.е.н., доцент, Сумський національний аграрний університет, Україна</i> МЕТОДОЛОГІЧНІ ПІДХОДИ ДО ОЦІНКИ ЕФЕКТИВНОСТІ МАРКЕТИНГОВОЇ ДІЯЛЬНОСТІ В СОЦІАЛЬНИХ МЕРЕЖАХ.....	31
<i>Могилка А., магістрант, Сумський національний аграрний університет, Україна</i> ЕКОЛОГО-ЕКОНОМІЧНІ ТЕНДЕНЦІЇ ВИКОРИСТАННЯ СІЛЬСЬКОГОСПОДАРСЬКИХ ЗЕМЕЛЬ В УКРАЇНІ.....	34
<i>Селезень О.М. ст.викладач, Мотузний Ю.М. магістрант, СНАУ, Суми, Україна</i> СУТНІСТЬ ТА СУЧАСНІ МЕТОДИ ПРОСУВАННЯ ПРОДУКЦІЇ В СИСТЕМІ МАРКЕТИНГУ.....	36

Басанець Д.В., студент, СНАУ, Суми, Україна

ІНФОРМАЦІЙНА ПІДСИСТЕМА ОБЛІКУ КАДРОВОГО ЗАБЕЗПЕЧЕННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ КОЛЕДЖУ ВСІ «ГАФК СНАУ»

На даний момент важко уявити людину, яка не має мобільного телефону або якогось іншого гаджета. Навіть у маленьких дітей часом помітиш таку річ. Інформаційні технології настільки глибоко проникли в наше повсякденне життя, що часом ми відмовляємося від зовнішнього світу, замінюючи його на технічні пристрої. Мандруючи містом, мимоволі стикаємося з тим, що безліч людей йдуть поруч один з одним мовчки, «уткнувшись» у свої телефони, бо вони насправді замінюють їм реальне спілкування.

З чим це може бути пов'язане? Насамперед, з тим, що наше суспільство вийшло на рівень постіндустріального розвитку. Створюються нові й нові технології, завдяки яким наше життя стає більш комфортним та простим. Переведення економіки України на новий рівень відповідно до умов сьогодення неможливе без формування відповідного інформаційного забезпечення. Для того, щоб конкурувати й мати можливість підвищити ефективність діяльності підприємства [1].

Завдяки стрімкому розвитку інформаційних технологій спостерігається розширення сфери їхнього застосування. Якщо раніше ледве чи не єдиною областю, в якій застосовувалися інформаційні системи, була автоматизація бухгалтерського обліку, то на сьогодні спостерігається впровадження інформаційних технологій у безліч інших сфер, наприклад, аудит, маркетинг і т.ін. Ефективне використання корпоративних інформаційних систем дозволяє робити більш точні прогнози й уникати можливих помилок в управлінні підприємством в умовах динамічного ринку та глобалізації економічного простору [2].

Кожна організація, незалежно від її розмірів, має відділ кадрів, який, своєю чергою, може бути представлений однією, двома чи великою кількістю фахівців. Відділ кадрів є тією ланкою, яка з'єднує рядових співробітників та вищих керівників.

В діяльності відділу кадрів можна назвати такі важливі складові: укладання трудового договору із співробітниками, ведення особистої картки, складання штатного розкладу,

графіка відпусток, створення різних наказів. Кадрове діловодство передбачає складання та оформлення документів за певними кадровими завданнями: прийом на роботу, звільнення, переведення, сумісництво, тимчасове переведення для заміщення відсутнього працівника, заохочення, стягнення, надання відпустки, відрядження, атестація тощо.

Відділ кадрів, будучи точкою, з якої все починається, змушений працювати з великою кількістю людей. Це, в свою чергу, призводить до того, що відбувається документообіг. У фахівців зберігається величезна кількість наказів, особових справ, анкет та інших документів, які є необхідними при такій роботі. Вміння працювати з документами, правильно будувати процеси діловодства та документальних потоків в установі надають значний вплив на результати діяльності, а точніше – на її ефективність. Можливість виконання та збереження всіх цих документів в електронному вигляді є важливим та актуальним завданням.

Усі документи є результатами виконання будь-яких кадрових операцій або вихідними даними для них. Таким чином, у фахівця з'явиться можливість виконання всіх кадрових операцій в електронному вигляді, що ефективніше, швидше та зручніше, ніж у паперовому.

Ось тільки деякі з переваг, які надаються при використанні обчислювальної техніки під час роботи відділу кадрів:

- Можливість оперативного контролю за достовірністю інформації;
- Зменшення числа можливих помилок при роботі;
- Можливість швидкого доступу до будь-яких даних;
- Можливість швидкого формування звітів;
- економія трудовитрат та витрат часу на обробку інформації.

Виконання всіх функцій абсолютно точно за інструкцією і без будь-якої програмної підтримки призводить до природних помилок, неточності, що є абсолютно неприпустимим. Саме тому створення інформаційної системи відділу кадрів є необхідним і виключно правильним заходом для будь-якого підприємства, така автоматизація діяльності знижує процент помилок у роботі, пов'язаних з людським фактором, скорочення часу пошуку необхідної інформації, а також збільшення ефективності роботи співробітників.

Особливо сьогодні створення такої інформаційної системи є важливим аспектом у діяльності кожного підприємства, оскільки ситуація в країні не є стабільною, важливо зберігати інформацію не лише у паперовій формі, а й забезпечити можливість збереження інформації в електронному вигляді.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Кучеркова С. О. Використання інформаційних технологій для просування малого бізнесу: зарубіжний досвід. Облік і фінанси. – 2017. – №1. – С. 161-167
2. Пономаренко В. С. Журавльова І. В. Латишева І. Л. Інформаційні системи в управлінні персоналом. Харків. Вид. ХНЕУ, 2008р. – с 5-6

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СУМСЬКИЙ НАЦІОНАЛЬНИЙ АГРАРНИЙ УНІВЕРСИТЕТ

МАТЕРІАЛИ

науково-практичної конференції
викладачів, аспірантів та студентів
Сумського НАУ

(25-28 квітня 2023 р.)

Рудакова Д.О. РОЛЬ ЕКОЛОГІЧНОГО МАРКЕТИНГУ В ФОРМУВАННІ КОНКУРЕНТНИХ ПЕРЕВАГ ДЕРЖАВНОГО ПІДПРИЄМСТВА	339
Базарний С.О. УПРАВЛІННЯ ТОВАРНОЮ ПОЛІТИКОЮ НА ПІДПРИЄМСТВАХ МАЛОГО БІЗНЕСУ	340
Сергієнко В.Г. КАДРОВА БЕЗПЕКА ЯК ОСНОВНИЙ ЕЛЕМЕНТ ЕКОНОМІЧНОЇ БЕЗПЕКИ ПІДПРИЄМСТВА	341
Бельська А.О. РЕКЛАМНА ДІЯЛЬНІСТЬ НА ПІДПРИЄМСТВАХ РОЗДРІБНОЇ ТОРГІВЛІ ТА СПОСОБИ ЇЇ ВДОСКОНАЛЕННЯ	342
Дундук Д.С. РОЛЬ І ЗНАЧЕННЯ КРЕДИТУ В РОЗВИТКУ ЕКОНОМІКИ УКРАЇНИ	343
В'юненко О.Б. СУЧАСНІ ПРОБЛЕМИ ІНФОРМАЦІЙНОЇ КІБЕРГІГІЄНИ	344
Баталова А.Б. ГЕОІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ ЯК ІНСТРУМЕНТ ЕКОЛОГІЧНИХ ДОСЛІДЖЕНЬ	345
Маяк А.О. ЦИФРОВІ ІНСТРУМЕНТИ ДЛЯ ПУБЛІКАЦІЇ ТА ПРОДАЖУ МУЗИКИ	346
Бусенко І.В. РОЗРОБКА ВІДЕОІГОР НА БАЗІ ТЕХНОЛОГІЇ P2E	347
Крамський А.С. ChatGPT – ТВІЙ МЕНТОР У НАВЧАННІ	348
Гервольський М.Д. АНАЛІТИЧНИЙ АНАЛІЗ РІЗНИХ ВИДИ ХОСТИНГІВ	349
Потапенко М.Т. АВТОМАТИЗОВАНА ІНФОРМАЦІЙНА СИСТЕМА ДЛЯ СПОРТИВНОГО КОМПЛЕКСУ	350
Єременко О.В. РОЗРОБКА ІНФОРМАЦІЙНОЇ СИСТЕМИ ПЕРЕВІРКИ ПРАКТИЧНИХ РОБІТ ІЗ ВИБІРКИ ДАНИХ: ШЛЯХИ І ТЕХНОЛОГІЇ	351
Мотлюк М.В. А/В ТЕСТУВАННЯ ЯК МЕТОД МАРКЕТИНГОВИХ ДОСЛІДЖЕНЬ В ІТ	352
Олефіренко Д.В. ПРОГНОЗУВАННЯ КІЛЬКОСТІ ВЕЛИКОЇ РОГАТОЇ ХУДОБИ В УКРАЇНІ З ВИКОРИСТАННЯМ ЕКОНОМЕТРИЧНИХ МЕТОДІВ	353
Бабуров Д.В. ПРОГНОЗУВАННЯ КІЛЬКОСТІ СІЛЬСЬКОГОСПОДАРСЬКОЇ ПТИЦІ В УКРАЇНІ З ВИКОРИСТАННЯМ ЕКОНОМЕТРИЧНИХ МЕТОДІВ	354
Шакотько А.М. ІНФОРМАЦІЙНА ПІДСИСТЕМА МАТЕРІАЛЬНОГО ЗАБЕЗПЕЧЕННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ ВСП «ГАФК СНАУ»	355
Басанець Д.В. ІНФОРМАЦІЙНА ПІДСИСТЕМА ОБЛІКУ КАДРОВОГО ЗАБЕЗПЕЧЕННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ КОЛЕДЖУ ВСП «ГАФК СНАУ»	356
Хіманова А.О. ПРОБЛЕМИ УПРАВЛІННЯ ІТ ПРОЄКТАМИ СУЧАСНИХ НАВЧАЛЬНИХ СИСТЕМ	357
Сіденко А.С. ПРОБЛЕМИ ПОБУДОВИ САЙТІВ ДЛЯ СИСТЕМ ДИСТАНЦІЙНОГО НАВЧАННЯ ...	358
Шакотько Є.С. СУЧАСНІ АЛГОРИТМИ АНАЛІЗУ ПРОДУКТИВНОСТІ ТА МОНІТОРИНГУ КОМП'ЮТЕРНИХ МЕРЕЖ	359
Грошовик Р.Ю. ОБГРУНТУВАННЯ НЕОБХІДНОСТІ АВТОМАТИЗАЦІЇ ВЕДЕННЯ ПОГОСПОДАРСЬКОГО ОБЛІКУ ТЕРИТОРІАЛЬНИХ ГРОМАД	360
Наконечний О.Б. АНАЛІЗ ПРОБЛЕМИ АВТОМАТИЗАЦІЇ РОЗРАХУНКУ БАЛАНСУ ГУМУСУ І ПОЖИВНИХ РЕЧОВИН В ҐРУНТІ ТА ПРОПОЗИЦІЇ ЩОДО ЇЇ ВИРІШЕННЯ	361
Мартинченко С.В. ОБРОБКА ВЕЛИКИХ ДАНИХ: ПРОБЛЕМИ І ШЛЯХИ ЇХ ВИРІШЕННЯ	362
Zhang Yuanyi THE KEY FACTORS IN PERSONAL DEVELOPMENT FROM THE PERSPECTIVE OF ENTERPRISES	363
Qin Fan IMPROVEMENT OF THE RISK MANAGEMENT SYSTEM AT THE ENTERPRISE	364
Jiang Maoli ORGANIZE THE CONCEPT AND METHOD OF BUSINESS PROCESS IMPROVEMENT	365
Zhang Buyao DISCUSSION ON THE MANAGEMENT OF CAREER PROMOTION OF ENTERPRISE EMPLOYEES	366
Cui Xiaoping THE IMPACT OF INNOVATIVE TRANSFORMATIONS IN THE AGRO-INDUSTRIAL SPHERE ON THE ACHIEVEMENT OF THE SUSTAINABLE DEVELOPMENT GOALS	367
Zhao Jing CONCEPTUAL APPROACHES TO MANAGING THE BEHAVIOR OF THE COMPANY'S PERSONNEL ON THE BASIS OF CORPORATE CULTURE	368
Гулько Б. В. МЕХАНІЗМ ДЕРЖАВНОГО РЕГУЛЮВАННЯ В СФЕРІ СТАЛОГО РОЗВИТКУ	369
Маслак О.М., Санін Р.С. ІНСТРУМЕНТИ ДЕРЖАВНОГО РЕГУЛЮВАННЯ ФЕРМЕРСЬКИХ ГОСПОДАРСТВ УКРАЇНИ В УМОВАХ ВОЄННОГО СТАНУ	370
Ніколаєнко Т.С. НАПРЯМИ РОЗВИТКУ ПУБЛІЧНОГО АДМІНІСТРУВАННЯ ОСВІТИ НА РІВНІ ТЕРИТОРІАЛЬНОЇ ГРОМАДИ	371
Гребченко Т.А., Стеценко Б.М. КОМУНІКАЦІЙНИЙ ЗВ'ЯЗОК ОРГАНІВ ДЕРЖАВНОГО УПРАВЛІННЯ ІЗ СУСПІЛЬСТВОМ	372
Майданевич С.В. АДМІНІСТРАТИВНО-ПРОЦЕДУРНІ ЗАСАДИ РЕГУЛЮВАННЯ В БАНКІВСЬКІЙ СФЕРІ	373
Chonglong Bi MANAGEMENT OF ORGANIZATIONAL CULTURE AS A FACTOR OF INCREASING COMPETITIVENESS	374
Тимошенко М.І. ПРИНЦИПИ СТВОРЕННЯ ТА УПРАВЛІННЯ КЛАСТЕРНИМИ ТА НАДКЛАСТЕРНИМИ ЛІКАРНЯМИ	375

ІНФОРМАЦІЙНА ПІДСИСТЕМА ОБЛІКУ КАДРОВОГО ЗАБЕЗПЕЧЕННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ КОЛЕДЖУ ВСП «ГАФК СНАУ»

Басанець Д.В., студ. 2 курсу ФЕІМ, спец. «Інформаційні системи та технології»
 Науковий керівник: к.т.н., доцент Толбатов А.В.
 Сумський НАУ

Комп'ютер став невід'ємною частиною управлінської системи підприємств. Однак сучасний підхід до управління припускає ще і вкладання грошей в інформаційні технології. Причому чим більше підприємство, тим більше повинні бути подібні вкладання. Ефективне використання корпоративних інформаційних систем дозволяє робити більш точні прогнози й уникати можливих помилок в управлінні підприємством в умовах динамічного ринку та глобалізації економічного простору.

З будь-яких даних і звітів про роботу підприємства можна отримати масу корисних зведень. І саме інформаційні системи дозволяють одержати максимум користі з усієї наявної інформації. Саме цим фактом і пояснюється життєздатність й бурхливий розвиток інформаційних систем – сучасний бізнес украй чутливий до помилок в управлінні, і для прийняття грамотного управлінського рішення в умовах невизначеності ринку необхідно постійно тримати під контролем різні аспекти фінансово-господарської діяльності підприємства.

Впровадження інформаційних систем у кадровий відділ відіграє важливу роль у підвищенні ефективності управління персоналом. Кадровий відділ є ключовим елементом в обробці різноманітної інформації, пов'язаної з відпустками, звільненнями, заробітною платою та іншими аспектами управління персоналом.

Системи управління кадрами є важливим інструментом для зберігання та обробки даних про співробітників. Вони дозволяють уникнути помилок, пов'язаних з веденням документації вручну. Крім того, такі системи забезпечують швидкий та легкий доступ до інформації про співробітників, що забезпечує більш ефективний кадровий облік.

Централізоване зберігання даних дозволяє більш ефективно керувати персоналом, зокрема, забезпечити рівномірне розподілення завдань та ресурсів між співробітниками. Це підвищує продуктивність та ефективність роботи відділу та підприємства в цілому.

Крім того, такі системи дозволяють здійснювати аналіз даних про персонал, що є корисним інструментом для прийняття рішень про планування кадрів, оцінки продуктивності та забезпечення правильної заробітної плати. Всі ці фактори в своєму комплексі забезпечують підвищення ефективності та ефективності роботи кадрового відділу, що має прямий вплив на успішність підприємства в цілому. Інформаційні системи не тільки полегшують взаємодію між співробітниками та керівництвом, але й сприяють покращенню самої комунікації. Завдяки використанню інформаційних систем, співробітники можуть взаємодіяти між собою безпосередньо, обмінюватися інформацією та спільно працювати над проектами, не втрачаючи часу на зайві зустрічі та переговори. Крім того, інформаційні системи забезпечують покращення координації роботи відділу та підприємства в цілому, що сприяє збільшенню ефективності та якості роботи.

У результаті використання інформаційних систем в роботі кадрового відділу можна очікувати покращення не лише взаємодії між співробітниками та керівництвом, а й рівня координації та ефективності роботи відділу в цілому. Таким чином, впровадження інформаційних систем є необхідним елементом сучасної організації роботи кадрового відділу та підприємства в цілому.

Таким чином, ІС у кадровому відділі дозволяють суттєво підвищити ефективність управління персоналом, скоротити час на рутинні завдання та підвищити точність та якість роботи. Однак, при виборі та впровадженні ІВ у кадровому відділі необхідно враховувати низку факторів.

- ІС повинні відповідати специфіці та потребам кадрового відділу компанії;
- при виборі ІС необхідно враховувати можливості та обмеження ІТ-інфраструктури компанії, а також рівень підготовленості персоналу до використання нових технологій;
- необхідно враховувати питання безпеки та конфіденційності даних;
- для успішного впровадження ІС у кадровому відділі необхідна підготовка персоналу, включаючи навчання та підтримку у процесі використання.

Отже інформаційні системи є необхідною складовою роботи кадрового відділу, що дозволяє оптимізувати та покращити якість роботи з персоналом. При виборі та впровадженні ІС у кадровому відділі необхідно враховувати специфіку компанії та ІТ-інфраструктури, а також забезпечувати безпеку та конфіденційність даних та підготовку персоналу до використання нових технологій.

Література

1. Пономаренко В. С. Журавльова І. В. Латишева І. Л. Інформаційні системи в управлінні персоналом. Харків. Вид. ХНЕУ, 2008р. с 5-7

ДОДАТОК Д ЛІСТИНГ ФАЙЛІВ ПРОГРАМНОГО КОДУ

Page1.html

```

<!DOCTYPE html>
<html lang="en">

<head>
  <title>Кадровий відділ</title>
  <script type="module" src="./js/check.js"></script>
  <meta http-equiv="Content" content="HTML/text" charset="UTF-8">
  <script src="https://kit.fontawesome.com/d514fe878e.js" crossorigin="anonymous"></script>
  <meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=1">
  <script async type="module" src="js/navigation.js"></script>
  <link href="https://unpkg.com/gridjs/dist/theme/mermaid.min.css" rel="stylesheet" />
  <link rel="stylesheet" href="css/saas.css">
</head>

<body>
  <div class="contaiter">
    <div class="nav">
      <ul>
        <li class="navLogo">
          <p class="navLogoOpen" style="display: none; color: #ffc043;">HRM SYSTEM</p>
          <p class="navLogoClose" style="color: #ffc043;">HRM </p>
        </li>

        <li class="navImg navA">
          <i class=" fa-solid fa-arrow-right navImg_L" style="color: #E8E8E8;"></i>
          <i class=" fa-solid fa-arrow-left navImg_R" style="display: none; color: #E8E8E8;"></i>
        </li>

        <li><a class="navA" href="/profile.html">
          <i class="fa-regular fa-user" style="color: #E8E8E8;"></i>
          <p class="navP">Профіль</p>
        </a></li>

        <li><a class="navA active" href="/index.html">
          <i class="fa-solid fa-table" style="color: #E8E8E8;"></i>
          <p class="navP">Кадровий відділ</p>
        </a></li>

        <li><a class="navA" href="/page3.html">
          <i class="fa-regular fa-calendar" style="color: #E8E8E8;"></i>
          <p class="navP">Календар подій</p>
        </a></li>

        <li><a class="navA" href="/page4.html">
          <i class="fa-solid fa-briefcase" style="color: #E8E8E8;"></i>
          <p class="navP">Звільнені працівники</p>
        </li>
      </ul>
    </div>
  </div>

```

```

    </a></li>
    <li><a class="navA" onclick="firebase.auth().signOut()">
      <i class="fa-solid fa-right-from-bracket" style="color: #E8E8E8;"></i>
      <p class="navP">Вихід</p>
    </a></li>
  </ul>
</div>
<script src="https://unpkg.com/gridjs/dist/gridjs.umd.js"></script>
<script type="module" src="js/db.js"></script>
</div>
</body>
</html>

```

Database.js

```

import { storage, dbRef, db } from "./firebase.js";
import { ref, set, get, remove, update } from "https://www.gstatic.com/firebasejs/9.18.0/firebase-database.js";
import { ref as sRef, uploadBytesResumable, getDownloadURL, deleteObject } from
"https://www.gstatic.com/firebasejs/9.18.0/firebase-storage.js";
import("https://unpkg.com/gridjs/dist/gridjs.umd.js")

```

```

const dismissalPerson = document.querySelector(".dismissalPerson")

// Create new person
createPersonBtn.onclick = () => {
  BoxToCreatePerson("addBtn", "Додати")
  const fullNameStr = document.querySelector(".fullNameStr")
  const startDateStr = document.querySelector(".startDateStr")
  const emplHistoryStr = document.querySelector(".emplHistoryStr")

  addBtn.addEventListener("click", e => {
    const file = document.getElementById("pdfFileInput").files[0];
    get(dbRef).then((snapshot) => {
      const arr = Object.values(snapshot.val())
      let count = 0
      arr.forEach((name) => {
        if (name.fullName === fullNameStr.value) { return count++ }
      })
      if (count === 0) {
        if (fullNameStr.value && startDateStr.value
          && emplHistoryStr.value && commandStr.value
          && phoneNumbStr.value && file.name) {
          const personID = Date.now()
          const imagesRef = sRef(storage, `${personID}/` + file.name);
          uploadBytesResumable(imagesRef, file)
            .then((snapshot) => {
              getDownloadURL(snapshot.ref).then((url) => {
                set(ref(db, 'employees/' + personID), {
                  personID: personID,
                  fullName: fullNameStr.value,

```

```

        startDateStr: startDateStr.value,
        emplHistoryStr: emplHistoryStr.value,
        department: departmentStr.value,
        status: true,
        author: nameProfile
    });
    // Reload table
    messege.remove()
    updtDbTable()
    });
    }).catch((error) => {
        console.error('Upload failed', error);
    });
    }
    else alert("Заповніть всі поля")
    }
    else alert("Даний користувач вже створений")
    })
    })
}

// delete person btn
table.addEventListener('click', (event) => {
    if (event.target.classList.contains('btnDell')) {
        let personID = event.target.id
        get(ref(db, 'employees/' + personID)).then((snapshot) => {
            const arr = Object(snapshot.val())
            const fileRef = sRef(storage, `${personID}/${arr.pdfFileName}`);
            deleteObject(fileRef)
                .catch((error) => {
                    console.error('Error deleting file', error);
                });
        })
        remove(ref(db, `employees/${personID}`))
        updtDbTable()
    }
})

// edit persone btn
table.addEventListener('click', (event) => {
    if (event.target.classList.contains('btnEdit')) {
        BoxToCreatePerson("editBtn", "Редагувати")
        get(arrPerson).then((snapshot) => {
            const values = Object(snapshot.val())
            fullNameStr.value = values.fullName
            startDateStr.value = values.startDateStr        })
        editBtn.addEventListener("click", () => {
            const messege = document.querySelector(".messege")
            const arrPerson = (ref(db, `employees/${personID}/`))
            update(arrPerson, {

```

```

        fullName: fullNameStr.value,
        department: departmentStr.value,
        personPosition: personPositionStr.value,
        startDateStr: startDateStr.value,
        emplHistoryStr: emplHistoryStr.value,
        commandStr: commandStr.value,
        phoneNumbStr: phoneNumbStr.value,
        author: nameProfile
    })
    messege.remove()
    document.body.style.overflow = "";
    updtDbTable()
  })
}
})

```

```

// render input message to create person / edit person / firePerson
function BoxToCreatePerson(nameBtn, valueBtn) {
  const newDiv = document.createElement("div");
  newDiv.classList.add("messege");
  if (nameBtn === "addBtn") {
    newDiv.innerHTML = `
      <div class="messege_content">
        <h3>Створити новий запис</h3>
        <div class="input_container">
          <div class="fullName input_container_flex">
            <p>ПІБ</p>
            <input class="fullNameStr messege_input" type="text">
          </div>
          <div class="startDate input_container_flex">
            <p>Дата початку роботи</p>
            <input class="startDateStr messege_input" type="date">
          </div>
          <div class="department input_container_flex">
            <p>Відділ</p>
            <input class="departmentStr messege_input" type="">
          </div>
          <div class="personPosition input_container_flex">
            <p>Посада</p>
            <input class="personPositionStr messege_input" type="">
          </div>
          <div class="emplHistory input_container_flex">
            <p>Трудова книжка</p>
            <input class="emplHistoryStr messege_input" type="text">
          </div>
          <div class="command input_container_flex">
            <p>Наказ</p>
            <input class="commandStr messege_input" type="text">
          </div>
          <div class="phoneNumb input_container_flex">

```

```

<p>Мобільний телефон</p>
<input class="phoneNumbStr messege_input" type="tel"
placeholder="+380 (__)__-__-__">
</div>
<div class="inpFile input_container_flex">
  <p>Копія документів</p>
  <input type="file" id="pdfFileInput" accept="application/pdf" class="inpFileStr">
</div>
</div>
<div class="inputBtnMenu">
  <input class="{nameBtn}" type="button" value="{valueBtn}">
  <input class="closeMessage" type="button" value="Відмінити">
</div></div> `;
  }
  if (nameBtn === "editBtn") {
    newDiv.innerHTML = `
      <div class="messege_content">
        <h3>Редагувати дані</h3>
        <div class="input_container">
          <div class="fullName input_container_flex">
            <p>ПІБ</p>
            <input class="fullNameStr messege_input" type="text">
          </div>
          <div class="department input_container_flex">
            <p>Відділ</p>
            <input class="departmentStr messege_input" type="">
          </div>
          <div class="personPosition input_container_flex">
            <p>Посада</p>
            <input class="personPositionStr messege_input" type="">
          </div>
          <div class="startDate input_container_flex">
            <p>Дата початку роботи</p>
            <input class="startDateStr messege_input" type="date">
          </div>
          <div class="emplHistory input_container_flex">
            <p>Трудова книжка</p>
            <input class="emplHistoryStr messege_input" type="text">
          </div>
          <div class="command input_container_flex">
            <p>Наказ</p>
            <input class="commandStr messege_input" type="text">
          </div>
          <div class="phoneNumb input_container_flex">
            <p>Мобільний телефон</p>
            <input class="phoneNumbStr messege_input" type="tel"
placeholder="+380 (__)__-__-__">
          </div>
        </div>
      </div>
    `;
    if (nameBtn === "dismissalBtn") {

```

```

<div class="inputBtnMenu">
<input class="{nameBtn}" type="button" value="{valueBtn}">
<input class="closeMessage" type="button" value="Відмінити">      }
    btnWrapper.insertAdjacentElement("afterend", newDiv);
    document.body.style.overflow = "hidden";
    const closeMessage = document.querySelector(".closeMessage")
    closeMessage.addEventListener("click", () => {
        document.body.style.overflow = "";
        newDiv.remove()
    })
  }
}
// else typeprofile = guest
else {
  createHTMLGuest()
  renderTableGuest()
}
style()
})
}

// Update db
function updtDbTable() {
  readDB().then((querySnapshot) => {
    let data = querySnapshot.map((val) => { return val })
    if (data.length > 0) {
      grid.updateConfig({
        data: data.map((row) => ([
          row.personID, row.fullName, row.department,
          row.personPosition, row.startDateStr,
          row.emplHistoryStr, row.commandStr, row.phoneNumbStr,
          gridjs.html(`
<a href="{row.pdfURL}" target="_blank">{row.pdfFileName}</a>`),
          gridjs.html(`
<i class="fa-regular fa-pen-to-square fa-lg btnEdit" id="{row.personID}" title="Редагувати" data-action="edit"
style="color: #555658;"></i>
          <i class="fa-solid fa-trash-can btnDell" id="{row.personID}" data-action="delete" title="Видалити" style="color:
#555658;"></i>
        `), row.author]))
      }).forceRender();
    }
    else location.reload()
  });
}

```

EditFile.js

```

import { storage, dbRef, db } from "./firebase.js";
import { ref, get, update } from "https://www.gstatic.com/firebasejs/9.18.0/firebase-database.js";
import { ref as sRef, uploadBytesResumable, getDownloadURL, deleteObject } from
"https://www.gstatic.com/firebasejs/9.18.0/firebase-storage.js";
import("https://unpkg.com/gridjs/dist/gridjs.umd.js")

```

```

import("https://cdn.jsdelivr.net/npm/exceljs@1.13.0/dist/exceljs.min.js")

firebase.auth().onAuthStateChanged(updateUser);
function updateUser(user) {
  const currentUser = user ? user.uid : null;
  get((ref(db, `users/${currentUser}`))).then(querySnapshot => {
    const typeProfile = querySnapshot.val().profile
    createHTML()
    if (typeProfile === "admin") {
      renderTable()
      const table = document.querySelector(".table")
      // edit file btn
      table.addEventListener('click', (event) => {
        if (event.target.classList.contains('btnEditFile')) {
          BoxToCreatePerson("editBtn", "Замінити файл")
          editBtn.addEventListener('click', () => {
            const file = document.getElementById('pdfFileInput').files[0];
            const imagesRef = sRef(storage, `${personID}/${file.name}`);
            get(ref(db, 'employees/' + personID)).then((snapshot) => {
              const arr = Object(snapshot.val())
              const fileRef = sRef(storage, `${personID}/${arr.pdfFileName}`);
              deleteObject(fileRef)
                .catch((error) => {
                  console.error('Error deleting file', error);
                });
            })
            uploadBytesResumable(imagesRef, file)
              .then((snapshot) => {
                getDownloadURL(snapshot.ref).then((url) => {
                  update(ref(db, 'employees/' + personID), {
                    pdfFileName: file.name,
                    pdfURL: url,
                  });
                  messege.remove()
                  updtDbTable()
                }
              )
            })
          })
        }
      })
    }
    else {
      renderTableGuest()
    }
  })
}

// Read database
function readDB() {
  const response = get(dbRef).then((

```

```

querySnapshot) => {
  if (querySnapshot.exists()) {
    const arr = Object.values(querySnapshot.val())
    let data = arr.reduce((acc, curr) => {
      if (curr.status) {
        acc.push(curr);
      }
      return acc;
    }, []);
    return data
  }
  else {
    console.log("No data available");
  }
})
.catch((error) => {
  console.error(error);
});
return response
}

// Create table ADMIN
function renderTable() {
  readDB().then((data) => {
    grid = new gridjs.Grid({
      resizable: true,
      sort: true,
      search: true,
      fixedHeader: true,
      pagination: { limit: 10 },
      language: {
        'search': {
          'placeholder': 'Пошук...'
        },
        'pagination': {
          'previous': 'Попередня',
          'next': 'Наступна',
          "showing": "Показано від",
          "of": "з",
          "to": "до",
          'results': () => 'результатів'
        }
      },
      columns: ["№", "ПІБ",
        "Файл", "Дії", "Користувач"],
      data: data.map((row) => (
        [
          row.personID, row.fullName,
          gridjs.html(
            <a href="${row.pdfURL}" target="_blank">${row.pdfFileName}</a>

```

```

    `),
    gridjs.html(`
      <i class="fa-regular fa-pen-to-square fa-lg btnEditFile" id="${row.personID}" title="Редагувати" data-action="edit
style="color: #555658;"></i>
    `), row. author]))
  }).render(document.getElementById("table"));
})
}
// Create table GUEST
function renderTableGuest() {
  readDB().then((data) => {
    grid = new gridjs.Grid({
      resizable: true,
      sort: true,
      search: true,
      fixedHeader: true,
      pagination: { limit: 10 },
      language: {
        'search': {
          'placeholder': 'Пошук...'
        },
        'pagination': {
          'previous': 'Попередня',
          'next': 'Наступна',
          "showing": "Показано від",
          "of": "з",
          "to": "до",
          'results': () => 'результатів'
        }
      },
      columns: ["№", "ППБ",
        "Файл", "Користувач"],
      data: data.map((row) => (
        [
          row.personID, row.fullName,
          gridjs.html(`
            <a href="${row.pdfURL}" target="_blank">${row.pdfFileName}</a>
          `), row. author]))
    }).render(document.getElementById("table"));
  })
}

// Update db
function updtDbTable() {
  readDB().then((querySnapshot) => {
    let data = querySnapshot.map((val) => { return val })
    if (data.length > 0) {
      grid.updateConfig({
        data: data.map((row) => ([
          row.personID, row.fullName,

```

```

        gridjs.html(`
        <a href="${row.pdfURL}" target="_blank">${row.pdfFileName}</a>`),
        gridjs.html(`
        <i class="fa-regular fa-pen-to-square fa-lg btnEditFile" id="${row.personID}" title="Редагувати" data-action="edit
style="color: #555658;""></i>
        `)))
    }).forceRender();
  }
  else location.reload()

});
}
function BoxToCreatePerson(nameBtn, valueBtn) {
  const newDiv = document.createElement('div');
  newDiv.classList.add('messege');
  newDiv.innerHTML = `
  <div class="messege_content">
  <h3>Оновити файл</h3>
  <div class="input_container">
  <div class="inpFile input_container_flex">
  <p>Копія документів</p>
  <input type="file" id="pdfFileInput" accept="application/pdf" class="inpFileStr">
  </div>
  </div>
  <div class="inputBtnMenu">
  <input class="${nameBtn}" type="button" value="${valueBtn}">
  <input class="closeMessage" type="button" value="Відмінити">
  </div></div>`;
  const navContainer = document.querySelector('.nav');
  navContainer.insertAdjacentElement('afterend', newDiv);
  const closeMessage = document.querySelector(".closeMessage")
  closeMessage.addEventListener("click", () => {
    newDiv.remove()
  })
}

```

Calendar.js

```

let newEvent;
let calendar
const wrapperCalendar = document.querySelector(".wrapperCalendar")
document.addEventListener('DOMContentLoaded', function () {
  readDB()
  calendar = new FullCalendar.Calendar(calendarEl, {
    locale: "uk",
    selectable: true, // вибір дат
    displayEventTime: false, // показ часу при події
    buttonText: {
      today: "Сьогодні"
    },
    headerToolbar: {

```

```

    left: "prev,next today",
    center: "title",
    right: "addEventButton"
  },
  customButtons: {
    addEventButton: {
      text: "Додати подію",
    }
  },
  select: function (info) {
    if (selectedDates = 2) {
      selectedDates = [];
    }
    selectedDates.push(info.start);
    selectedDates.push(info.end);
  },
  eventClick: function (info) {
    if (confirm("Ви впевнені що хочете видалити запис '" + info.event.title + "'?")) {
      remove(ref(db, `calendar/${info.event.title}`))
      info.event.remove();
      calendar.removeAllEvents();
      readDB()
    }
  },
});
calendar.render();
const addEventButton = document.querySelector(".fc-addEventButton-button")
addEventButton.onclick = function () {
  BoxToCreatePerson()
  const addEvent = document.querySelector(".addEvent");
  const eventTitleStr = document.querySelector(".eventTitleStr")
  const messege = document.querySelector(".messege")
  addEvent.onclick = function () {
    if (selectedDates.length > 0) {
      if (eventTitleStr.value) {
        const color = getRandomColor()
        const eventObj = {
          title: eventTitleStr.value,
          start: selectedDates[0].toISOString(),
          end: selectedDates[1].toISOString(),
          backgroundColor: color,
          borderColor: color,
        };
        set(ref(db, 'calendar/' + eventTitleStr.value), eventObj)
          .catch((error) => console.log("Error adding event to database: ", error));
        calendar.removeAllEvents();
        selectedDates = [];
        readDB()
      }
    } else {

```

```

        alert("Оберіть дату для створення події");
    }
    messege.remove()
}
}
});
function getRandomColor() {
    const red = Math.floor(Math.random() * 150) + 80;
    const green = Math.floor(Math.random() * 150) + 80;
    const blue = Math.floor(Math.random() * 150) + 80;
    return `rgb(${red}, ${green}, ${blue})`;
}
function readDB() {
    const calendarRef = ref(db, "calendar");
    onValue(calendarRef, (snapshot) => {
        const data = snapshot.val();
        for (const id in data) {
            const eventDataStart = data[id];
            const eventObj = {
                title: eventDataStart.title,
                start: eventDataStart.start,
                end: eventDataStart.end,
                backgroundColor: eventDataStart.backgroundColor,
                borderColor: eventDataStart.borderColor,
            };
            newEvent = calendar.addEvent(eventObj);
        }
    });
}
function BoxToCreatePerson() {
    const newDiv = document.createElement("div");
    newDiv.classList.add("messege");
    newDiv.innerHTML = `
        <div class="messege_content">
        <h3>Створити нову подію</h3>
        <div class="input_container">
        <div class="eventTitle input_container_flex">
        <p>Подія</p>
        <input class="eventTitleStr messege_input" type="text">
        </div>
        <div class="inputBtnMenu">
        <input class="addEvent" type="button" value="Створити">
        <input class="closeMessage" type="button" value="Відмінити">
        </div>
        </div> `;
    wrapperCalendar.insertAdjacentElement("afterend", newDiv);
    const closeMessage = document.querySelector(".closeMessage")
    closeMessage.addEventListener("click", () => {
        newDiv.remove()
    })
}

```

```
}
```

STYLE.CSS

```
@import "reset.css";
@import url('https://fonts.googleapis.com/css2?family=Poppins:wght@300;400;500;600;700&display=swap');
input:focus {
  outline: none;}
/// messege box
.messege {
  padding: 0px 15px 0px 60px;
  background: #ffffffb4;
  backdrop-filter: blur(3px);
  position: fixed;
  z-index: 2;
  width: 100%;
  height: 100%;
  top: 0%;
  left: 0%;
  overflow: auto;
  &_content {
    margin: 8% auto;
    box-shadow: 1px 1px 18px #00000042;
    background: #ffffff;
    max-width: 600px;
    min-width: 380px;
    border-radius: 10px; }
  &_input p {
    margin: 0px 0px 10px 20px;
  }
  &_input {
    padding: 0px 6px;
    width: 150px;
    height: 30px;
    border-radius: 6px;
    background: #d8efff; }
}
// messege box button
.addBtn,
.dismissalBtn,
.editBtn,
.addEvent {
  width: 110px;
  height: 30px;
  border-radius: 6px;
  color: #ffffff;
  background: #18A0FB;
  margin-right: 10px;
  transition: 0.6s;}
.addBtn:hover,
.dismissalBtn:hover,
```

```

.editBtn:hover {
  color: #18A0FB;
  background: #ffffff;
  border: 1px solid #18A0FB;
  transition: 0.6s;}
.closeMessage {
  width: 110px;
  height: 30px;
  border-radius: 6px;
  border: 1px solid #858585;
  color: #858585;
  background: white;
  transition: 0.4s;}
.closeMessage:hover {
  color: #18A0FB;
  border: 1px solid #18A0FB;
  transition: 0.4s;}
.eventTitleStr {
  width: 200px;
// messege box button end
// page 1
.wrapper {
  width: 98%;}
.pageTitle {
  top: 35px;
  left: 80px;
  position: absolute;
  font-weight: 700;
  font-size: 24px;
  color: #232830;}
// page 1 btn dissma, edit persone, search form
.contaiter {
  min-height: 100%;
  position: relative;}
.btnWrapper {
  top: 97px;
  left: 72px;
  position: absolute;
  z-index: 1;}
.createPerson {
  width: 165px;
  height: 41.5px;
  border-radius: 6px;
  color: #EEF1F6;
  background: #18A0FB;
  transition: 0.6s;
  margin-right: 10px;}
.createPerson:active {
  border: 4px solid #18A0FB;
  transition: 0.2s;}

```

```

// table head and footer
//head
th.gridjs-th {
  background: #f0f1f381;
  padding-top: 10px;}
// footer
.gridjs-footer {
  background: #f0f1f381;}
.gridjs-pagination .gridjs-pages button:first-child:hover {
  color: #18A0FB;
  border: 1px solid #18A0FB;
  transition: 0.4s;}
.gridjs-pagination .gridjs-pages button.gridjs-currentPage {
  border-radius: 6px;
  border: 1px solid #18A0FB;
  transition: 0.4s;}
.gridjs-pagination .gridjs-pages button:last-child:hover {
  color: #18A0FB;
  border: 1px solid #18A0FB;
  transition: 0.4s;}
.gridjs-pages button {
  color: #555658;
  border-radius: 6px;
  border: 1px solid #8C8E96;
  transition: 0.4s;}
// table body
.gridjs-td {
  white-space: nowrap;
  overflow: hidden;
  text-overflow: ellipsis;}
// btns table body
.gridjs-td span {
  display: flex;
  justify-content: center;
  align-items: center;
  height: 20px;}
.btnEditFile,
.btnDell,
.btnEdit {
  padding: 14px 8px;
  border-radius: 10px;
  transition: 0.5s;}
.btnEdit {
  margin-right: 10px;}
.btnDell {
  padding: 8px 8px;}
.btnEditFile:hover,
.btnDell:hover,
.btnEdit:hover {
  background: #c6e8ff;

```

```

    transition: 0.5s;}
.gridjs-container {
    max-height: 180%;
    margin-bottom: 20px}
// btns table body end
// page 3 CALENDAR
.wrapperCalendar {
    position: relative;
    left: 0%;
    top: 0px;
    width: 100%;
    height: 100%;
    padding: 0px 20px 0px 60px;
    overflow: auto;}
.calendar {
    max-width: 750px;
    margin: 20px auto 50px;
    padding: 13px;
    border-radius: 5px;
    box-shadow: 1px 2px 30px #0a2e687a;}
.fc-addEventButton-button {
    width: 167px !important;
    height: 35px !important;
    border-radius: 6px !important;
    color: #EEF1F6 !important;
    background: #18A0FB !important;
    font-weight: 600 !important;
    border: 0px !important;
    transition: 0.6s !important;}
.fc-addEventButton-button:hover {
    color: #18A0FB !important;
    background: #ffffff !important;
    border: 1px solid #18A0FB !important;
    transition: 0.6s !important;}
.fc-today-button {
    background: #FFFFFF !important;
    color: #515353 !important;
    border: 1px solid #8C8E96 !important;
    transition: 0.6s !important;
    border-radius: 6px !important;}
.fc-prev-button:hover,
.fc-next-button:hover,
.fc-today-button:hover {
    color: #18A0FB !important;
    border: 1px solid #18A0FB !important;
    transition: 0.4s !important;}

```