Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет комп'ютерних наук
(повна назва)

Кафедра програмної інженерії
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти перший (бакалаврський)
Програмна система для закладів дошкільної освіти з керуванням режиму дня та харчування дітей. Front-end
(тема)

Виконав:
студент 4 курсу, групи ПЗПІ-20-3
Бондаренко А.А
(прізвище, ініціали)

Спеціальність 121 – Інженерія програмного забезпечення
(код і повна назва спеціальності)
Тип програми освітньо-професійна
Освітня програма Програмна інженерія
(повна назва освітньої програми)

Керівник доц. кафедри ПІ Побіженко І,О.
(посада, прізвище, ініціали)

Допускається до захисту
Зав. кафедри _____ З.В.Дудар
(підпис) (прізвище, ініціали)

2024 р.

Харківський національний університет радіоелектроніки

Факультет комп’ютерних наук
Кафедра програмної інженерії
Рівень вищої освіти перший (бакалаврський)
Спеціальність 121 – Інженерія програмного забезпечення
Тип програми Освітньо-професійна
Освітня програма Програмна Інженерія
(шифр і назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«___» _____ 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Бондаренко Анні Артемівні
(прізвище, ім’я, по батькові)

1. Тема роботи Програмна система для закладів дошкільної освіти з керуванням режиму дня та харчування дітей. Front-end
Затверджена наказом по університету від 20.05. 2024р. № 471 Ст
2. Термін подання студентом роботи до екзаменаційної комісії 14.06.2024
3. Вихідні дані до роботи Розробити програмну систему, котра надаватиме функції для закладів дошкільної освіти, які різняться в залежності від ролі, мовою програмування TypeScript з використанням бібліотеки React.js.
4. Перелік питань, що потрібно опрацювати в роботі
Вступ, аналіз предметної галузі, формування вимог до програмної системи, архітектура та проєктування програмного забезпечення, опис прийнятих програмних рішень, тестування, впровадження розробленого програмного забезпечення, висновки, додатки.

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Аналіз предметної галузі	08.04.2024	*виконано*
2	Створення специфікації ПЗ	12.04.2024	*виконано*
3	Проєктування ПЗ	15.04.2024	*виконано*
4	Розробка ПЗ	15.05.2024	*виконано*
5	Тестування ПЗ	17.05.2024	*виконано*
6	Оформлення пояснювальної записки	01.06.2024	*виконано*
7	Підготовка презентації та доповіді	06.06.2024	*виконано*
8	Попередній захист	16.06.2024	*виконано*
9	Нормоконтроль, рецензування	16.06.2024	*виконано*
10	Здача роботи у електронний архів	16.06.2024	*виконано*
11	Допуск до захисту у зав. кафедри	17.06.2024	*виконано*

Дата видачі завдання 8 квітня 2024р.

Студент (ка)_____ Бондаренко А.А.
(підпис)

Керівник роботи _____ доц. кафедри ПІ Побіженко І,О.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка до кваліфікаційної роботи бакалавра, 118 стор., 55 рис., 2 табл., 34 джерела.

АВТОМАТИЗАЦІЯ ПРОЦЕСІВ, ВЕБ-СИСТЕМА, ЗАКЛАД ДОШКІЛЬНОЇ ОСВІТИ, КАЛЕНДАР, КОНТРОЛЬ БЕЗПЕКИ, ПОДАЧА ЗАЯВОК, РАХУНКИ, РОЗКЛАД, УПРАВЛІННЯ, ШТУЧНИЙ ІНТЕЛЕКТ, UI, UX.

Об’єктом розробки є програмна система для закладів дошкільної освіти (ЗДО) «ChildWell».

Мета розробки – створення програмної системи для закладів дошкільної освіти, яка сприятиме ефективності організації освітнього процесу та розвитку особистості дитини дошкільного віку з використанням ШІ.

Стек технологій, використаний у процесі розробки для створення серверної частини – мова програмування C# і фреймворк ASP.NET. Клієнтські частини створено за допомогою мови TypeScript, бібліотеки React.js та фреймворку React Native .Дані зберігаються в базі Microsoft SQL Database, що хоститься на Azure.

У результаті розробки створено програмний продукт, котрий має фунціонал для автоматизації рутинних процесів у забезпеченні діяльності дошкільного закладу.

PROCESS AUTOMATION, WEB-BASED SYSTEM, PRESCHOOL EDUCATION INSTITUTION, CALENDAR, SECURITY CONTROL, APPLICATION SUBMISSION, INVOICES, TIMETABLE, MANAGEMENT, ARTIFICIAL INTELLIGENCE, UI, UX.

A software system for preschool educational institutions (PEI) ‘ChildWell’ is the object of development. The purpose of the development is to provide a software system for preschool education institutions that will contribute to the efficiency of the educational process and the development of the personality of a preschool child using AI. The

technology stack used in the development process to create the server side is the C# programming language and the ASP.NET framework. The client side was developed using TypeScript, the React.js library, and the React Native framework. The data is stored in the Microsoft SQL Database hosted on Azure. As a result of the development, a software product that has the functionality to automate routine processes in the activities of a preschool institution has been created.

Я, Бондаренко Анна Артемівна, студент гр. ПЗПІ-20-3, здобувач вищої освіти на першому (бакалаврському) рівні кафедри «Програмна інженерія», заявляю: моя кваліфікаційна робота на тему «Програмна система для закладів дошкільної освіти з керуванням режимом дня харчуванням дітей. Front-end частина», що буде представлена до екзаменаційної комісії для публічного захисту, виконана самостійно, в ній не містяться елементи плагіату і вона може бути опублікована в електронному архіві відкритого доступу ElAr KhNURE. Усі запозичення з друкованих та електронних джерел мають відповідні посилання.

Я ознайомлена з діючим положенням «Про протидію академічному плагіату в ХНУРЕ», згідно з яким виявлення плагіату є підставою для відмови до допуску кваліфікаційної роботи до захисту та застосування дисциплінарних заходів.

ЗМІСТ

ВСТУП

Сучасні ІТ-технології значно впливають на всі сфери життя людини, зокрема організацію освітнього процесу в дошкільних закладах. Створені програмні продукти активно залучаються в освітній простір, використовуються в управлінні дошкільною установою, організаційно-методичному супроводі методичної роботи, у взаємодії з дітьми і батьками, як повноправними учасниками освітнього процесу. Зростає зацікавленість фахівців до використання сучасних додатків і програмних систем з метою уникнути рутини у своїй професійній діяльності.

Однак, бажання активно залучати ІТ-технології стикається з низкою суперечностей, зокрема: обмеженість функціоналу програмних застосунків і наявність мовного бар'єру, недосконаліть сучасних додатків і ускладнений інтерфейс і т.д.

Ураховуючи реалії сьогодення, дошкільна освіта потребує сучасного програмного забезпечення для організації ефективного освітнього процесу відповідно до вимог «Державного стандарту дошкільної освіти» та міжнародних стандартів якості освіти.

Темою кваліфікаційної роботи є програмна система для закладів дошкільної освіти з керуванням режимом дня харчуванням дітей. Front-end частина. Основне завдання програмної системи – створення програмної системи для закладів дошкільної освіти, яка сприятиме ефективності організації освітнього процесу та розвитку особистості дітей дошкільного віку з використанням ШІ.

Метою роботи є розробка програмної системи, яка має фунціонал для оптимізації і автоматизації рутинних процесів в забезпеченні діяльності дошкільного закладу. Програмний продукт надає можливість фахівцям і батькам у додатку здійснювати перегляд розкладу, керування відвідуванням та інші операції, що автоматизують діяльність закладів дошкільної освіти.

Для розробки програмного продукту використано мови програмування C#, TypeSctipt і середовище розробки Microsoft Visual Studio Code.

1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ

1.1 Аналіз предметної галузі

Аналіз предметної галузі є невід'ємною частиною розробки складної програмної системи. В рамках розробки програмного продукту було виокремлено процеси та користувачів предметної області, а також визначено їхні інформаційні потреби.

Сучасний світ ставить багато викликів перед освітніми установами, зокрема фахівцями дошкільної освіти. Реалії сьогодення та особливості організації освітнього процесу значно ускладнюють реалізацію вимог до розвитку обов'язкових компетентностей та досягнення результатів освіти дитини дошкільного віку. Особливої уваги потребує процес створення умов, які забезпечать дотримання вимог, виокремлених у «Державному стандарті дошкільної освіти відповідно до міжнародних стандартів якості освіти».

В освітньому процесі дошкільного закладу важливим є моніторинг як академічних досягнень, так і розвитку дітей, впливу освітнього середовища на фізичне й ментальне здоров'я дошкільників, емоційний стан усіх учасників. Виходячи з цього, створення освітнього середовища, змістовне наповнення освітніх напрямів, організаційні питання реалізації освітнього процесу, зокрема розпорядку дня та ключових діяльностей дітей, стають актуальними аспектами, що впливають на здоров'я та розвиток дошкільнят.

Протягом останніх років фахівці докладають значних зусиль та приділяють особливу увагу процесу створення освітнього середовища дошкільного закладу, який враховує вимоги «Стандарту дошкільної освіти». Умовами реалізації «Стандарту дошкільної освіти» є: участь сім'ї в розвитку дитини; взаємодія закладів дошкільної освіти з сім'ями дітей як учасниками освітнього процесу; педагог як фахівець, який стимулює процес розвитку дитини; зміст та організація освітнього процесу; універсальний дизайн у закладі дошкільної освіти; наступність між дошкільною та початковою освітою в реалізації перспектив розвитку дитини; роль суспільства / громади в забезпеченні доступної та якісної дошкільної освіти [1].

Отже, на перший рівень освіти, яким є дошкільна освіта, покладено відповідальність стосовно закладення підґрунтя для розвиту особистості, формуючи її мотивації та цінності, які впливають на подальші можливості в житті [2].

Інтеграція сучасних технологій у сферу освіти постає як важлива необхідність у сучасному світі, особливо в контексті сучасних реалій дошкільної освіти, де створення програмних систем для керування організацією освітнього процесу (режимом дня та харчуванням дітей і т.д.) виявляється надзвичайно актуальним завданням і покликане автоматизувати вирішення певних організаційних питань реалізації освітнього процесу .

Особливого значення в сфері дошкільної освіти набуває використання штучного інтелекту (ШІ). Залучення ШІ сприятиме узагальненню найкращого досвіду та автоматизації планування розкладу дня та організації харчування дітей у дошкільних закладах. Штучний інтелект доречно використовувати для створення ефективного розкладу дня, що враховує доцільні види діяльності дошкільників, харчування та відпочинку через електронний календар. Запропонована система враховує вікові особливості та потреби груп дошкільників, щоб забезпечити оптимальний розподіл занять та сприяти гармонійному розвитку кожної дитини, економить час фахівців на вирішення організаційних питань, дає можливість більше уваги приділити взаємодії з учасниками освітнього процесу.

Головними користувачами системи є педагогічний персонал, який використовує її для планування розкладу дня та організації харчування. Цільова аудиторія – дошкільники, які залучаються до різних видів діяльності, які позначено у розробленому розкладі дня та отримують освіту, відповідну їхнім потребам та віковим особливостям. Цей аналіз підкреслює важливість використання штучного інтелекту для оптимізації процесів в дошкільних закладах та покращення якості освітнього процесу.

Результати аналізу системи організації освітнього процесу в дошкільному закладі засвідчують, що існує 3 основні цільові групи користувачів, які в ній поєднуються: адміністрація, вихователі та батьки як учасники освітнього процесу.

Акцентуємо увагу на ключових інформаційних процесах, які відбуваються всередині закладу дошкільної освіти і актуальні для зазначених вище учасників освітнього процесу:

– формування розкладу дня для груп;
– подача заявок у садочок;
– ведення журналів відвідування;
– оплата рахунків.

Кожна цільова група користувачів має власні інформаційні потреби. Наприклад, для підвищення ефективності роботи керівника закладу дошкільної освіти, йому в нагоді стануть такі можливості:

– налаштування активностей садочка;
– обробка й прийом заяв на вступ до садочка;
– управління персоналом.

У певних ситуаціях, можуть виникати суперечності між бажанням вихователів мінімізувати навантаження та вимогами до професійного рівня фахівців, передбачених освітньою програмою.

Таким чином, для вихователів важливими аспектами є наступні:

– організація освітньої діяльності і розвитку дошкільників відповідно розкладу (керування розкладом вихованців);
– моніторинг відвідування дошкільного закладу дітьми;
– отримання статистичних даних, результатів моніторингу освітнього процесу щодо групи.

Батьки ж у свою чергу, потребують наступних інструментів:

– можливість подати заяву до садочку;
– моніторинг освітньої та ігрової діяльності дитини;
– безпека дитини в дошкільному закладі (передача дітей з рук в руки);
– можливість оплатити рахунки та переглянути їхні складові.

Отже, актуальність дослідження випливає з необхідності оптимізації і автоматизації організації освітнього процесу (зокрема розробки розкладу, способів

взаємодії з батьками, планування харчування) в дошкільних закладах з використанням сучасних технологічних рішень. Застосування ШІ, зокрема GPT (Generative Pre-trained Transformer) моделей, дозволяє реалізувати індивідуальний підхід до кожної вікової групи дошкільного закладу, сприяючи гармонійному розвитку дітей, їх безпеці. Використання ШІ для генерації розкладів дозволяє врахувати не тільки освітні, але й фізіологічні потреби дітей, забезпечуючи тим самим ефективне та здоровозберігаюче освітнє середовище.

1.2 Виявлення та вирішення проблем

Виявлення проблем та шляхів їх вирішення є ключовим етапом розробки програмного забезпечення, оскільки допомагає в розумінні потреб користувачів, дозволяє уникнути потенційних помилок та недоліків, визначити найбільш важливі аспекти для вирішення проблем та виокремити пріоритетні задачі.

1.2.1 Проблеми предметної галузі

У процесі роботи над даним програмним продуктом було виявлено низку наявних проблем в описуваній предметній галузі.

По-перше, присутня велика кількість рутинної роботи при плануванні розкладу дня та харчування (витрати часу, можливі помилки).

По-друге, сьогодні праця керівника ДНЗ в першу чергу пов’язана з надмірною бюрократизацією системи. Навала звітних документів різних галузей: охорона праці, охорона дитинства, пожежна безпека, цивільна оборона, санітарно-епідеміологічна служба, юстиція, харчування, фінансування тощо[3].

Окремим аспектом діяльності керівників ДНЗ залишається є звітування в органи місцевої влади.

По-третє, необхідне оновлення ролі батьків, як учасників освітнього процесу та усвідомлення ними ключових завдань дошкільного закладу.

Наостанок, система безпеки, особливо контролю за передачею дітей із дошкільного закладу в руки дорослих потребує удосконалення. Відсутність автоматизації ускладнює зазначений процес. Ураховуючи реалії сьогодення, такий

стан створює загрозу безпеці дітей та умов для небажаних ситуацій, зокрема передачі дитини небажаній особі.

На ринку освітніх додатків, вже існують рішення для часткового вирішення зазначених проблем. Після проведення аналізу IT-ринку було визначено 5 програмних продуктів, які мають подібний функціонал. Розглянемо клієнтські застосунки цих програмних систем. Відзначивши їхні переваги та слабкі сторони можливо створити програмний продукт, який сприятиме ефективному вирішенню зазначених потреб дошкільного закладу освіти.

1.2.2 Аналіз аналогів

Програмний продукт «Preschool2me» частково вирішує зазначені вище потреби. Він призначений саме для упорядкування діяльності дошкільних закладів (див. рис. 1.1).

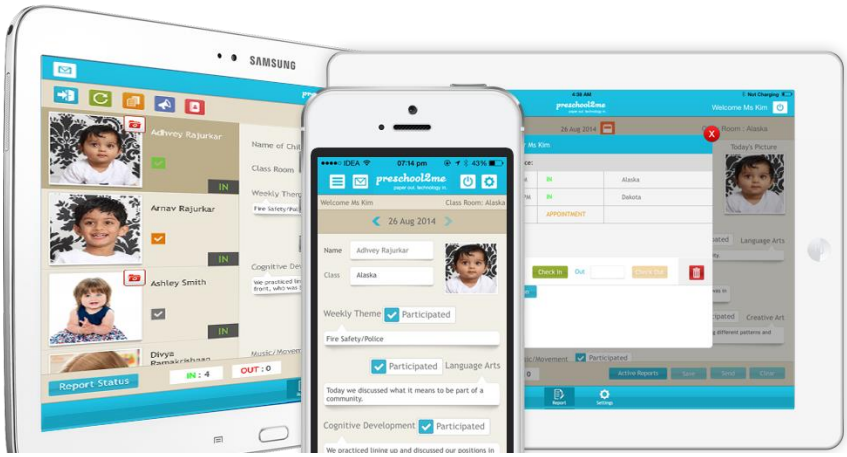


Рисунок 1.1 – Інтерфейс системи «Preschool2me» (за даними [4])

«Preschool2me» надає такі засоби:

– індивідуальні звіти з фотографіями;
– моніторинг відвідування дітей;
– розсилка повідомлень батькам завідувачем;
– створення планів занять для вихователя;
– дроп-оф форма, що містить додаткову інформацію (як дитина спала, коли востаннє їла і т.д.) та заповнюється батьками.

Як бачимо, ця система має ряд переваг, серед них зручний доступ до функцій через мобільний додаток для батьків та педагогів, а також функціонал для завідувача дошкільної установи. Окрім того, у програмному продукті є функція, яка пропонує імпорт даних вже наявних в інформаційному обігу закладу дошкільної освіти.

З одного боку, користувацький інтерфейс системи «Preschool2me» є зрозумілим, за виключенням певних кнопок, що містять лише іконки без супроводжуючого тексту, який уточнював би функцію того чи іншого елементу. А з іншого – є проблемним, бо інтерфейс доступний лише англійською мовою, що обмежує можливості доступу українського споживача.

Окрім зазначеного вище, система «Preschool2me» має низку характерних недоліків. Одним з найпомітніших є відсутність можливостей моніторингу. Запропонована система не надає інструментів для контролю за часом та особою, яка привела або забрала дитину. Попри можливість створити плани занять, генерацію розкладу не передбачено. Аспект оплати рахунків також не включено в даний програмний продукт.

Наступний програмний продукт, розроблений для сприяння ефективному управлінню дошкільними закладами, має назву «Procare». Він пропонує інструменти для директорів, вихователів та батьків (див. рис. 1.2).

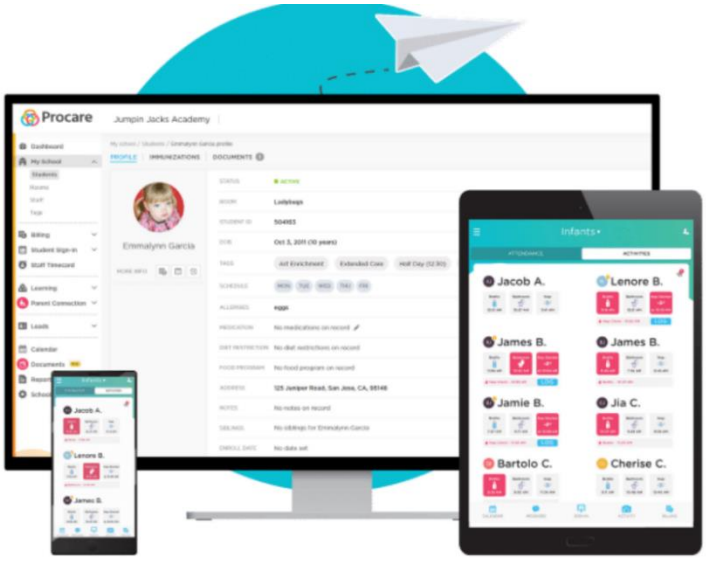


Рисунок 1.2 – Інтерфейс системи Procare (за даними [5])

«Procare» надає такі можливості:

– відстеження прогресу дітей;

– спілкування з родинами;

– управління закладом;

– щоденні звіти;

– сплата рахунків;

– відстеження діяльності вихованців.

Отже, додаток має спеціальні розділи для керівників, вчителів та батьків, дозволяючи кожній групі користувачів отримати необхідні функції. З приємних бонусів можна відзначити, що вони керуються підходом ціна «все в одному», незалежно від того, скільки дітей перебуває під піклуванням садочка. Легка навігація та швидкий доступ до важливої інформації через мобільний додаток роблять це рішення доволі привабливим.

Серед недоліків системи найвагомішими можна виділити те, що «Procare» не володіє вбудованими інструментами для планування розкладу та має малу кількість мов інтерфейсу.

«KidCheck» – це програмний продукт, спеціалізований на контролі за дітьми під час їхнього перебування у закладах, призначених для дітей (див. рис. 1.3).

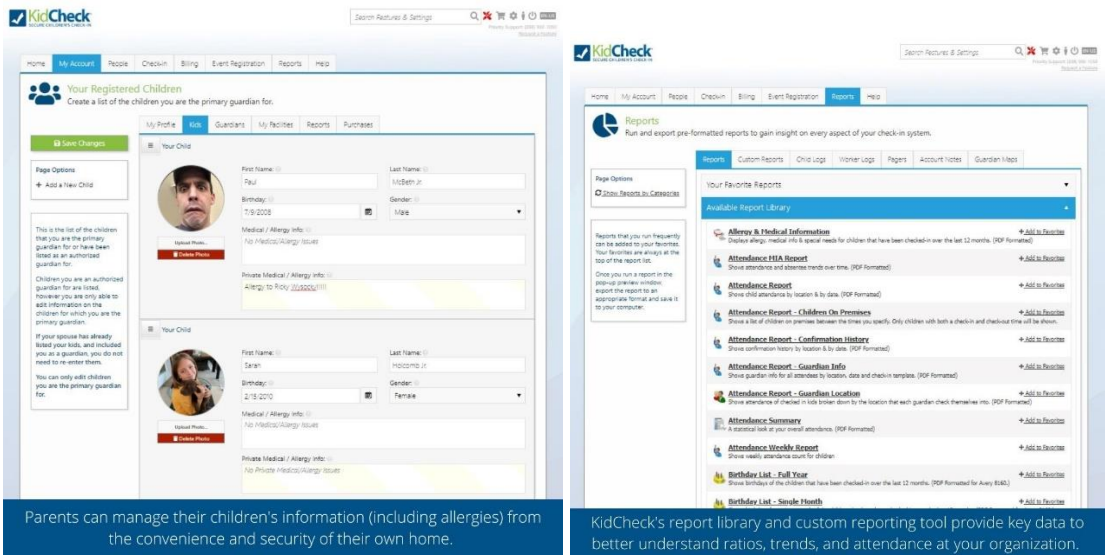


Рисунок 1.3 – Інтерфейс системи «KidCheck» (за даними [6])

Функціонал включає такі можливості:

– розширені функції безпеки;

– точне відстеження відвідування;

– мобільна експрес-реєстрація присутності;

– екстрені текстові повідомлення.

Основною перевагою «KidCheck» є мобільний чек-ін (можливість відмічати присутність дитини за допомогою мобільних пристроїв). Також вони забезпечують можливість інтеграції з зовнішнім програмним забезпеченням. Новим клієнтам надається безкоштовне налаштування, навчання та підтримка. Це може бути корисним, оскільки на початку користування програмою, інтерфейс може бути важким для розуміння.

Можна зробити висновок, що «KidCheck» пропонує широкий спектр функцій для ведення списків відвідувачів, та інтегрує програмне забезпечення (ПЗ) для керування справами закладу освіти.

Окрім вище зазначених переваг, для KidCheck характерні недоліки, перелік яких наведено далі. Мала кількість мов інтерфейсу. Відсутність функцій, спрямованих на планування розкладу та харчування, запису до закладу дошкільної освіти та неможливість оплатити рахунки роблять цю систему менш конкурентноспроможною.

«Teach 'n Go» – система управління школами для навчальних центрів і освітніх підприємств (див. рис. 1.4).

«Teach 'n Go» пропонує такі функціональні можливості:

– самостійно скласти розклад занять;

– імпорт розкладу з Excel;

– перегляд особистого календаря;

– моніторинг присутності на заняттях;

– перегляд квитанцій;

– здійснення платежів.

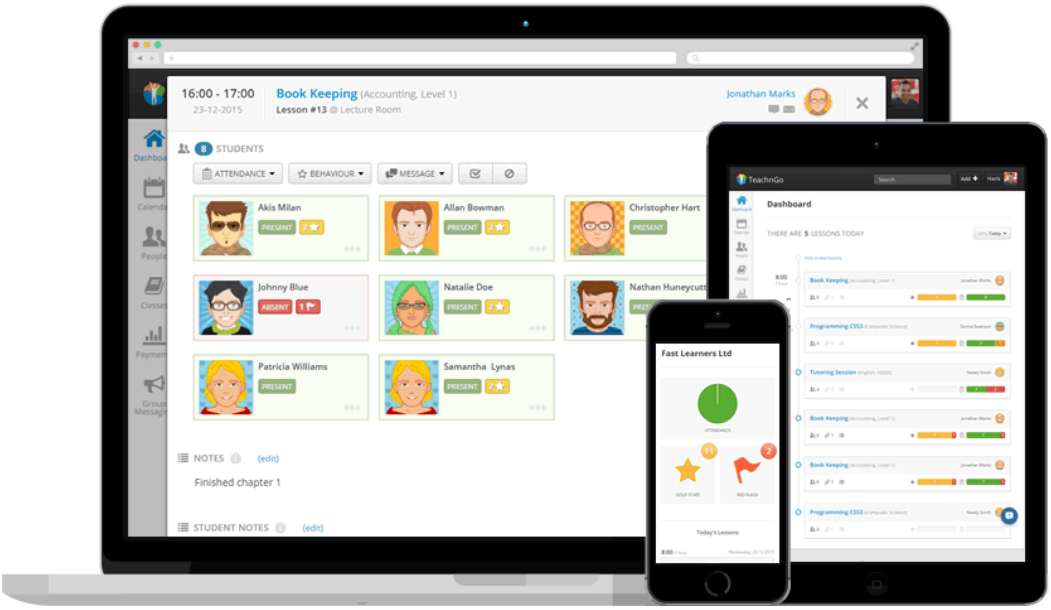


Рисунок 1.4 – Інтерфейс системи «Teach 'n Go» (за даними [7])

Може знадобитись час, аби розібратися з функціями, доступними завдяки користувацькому інтерфейсу, проте загалом додаток можна назвати простим у використанні. Корисним є перегляд календаря в щоденному, тижневому або місячному режимі та фільтрація, щоб переглядати календар за учнем, учителем, класом або класною кімнатою. Перевагою є можливість переглядати зміни в розкладі в реальному часі.

Приємним аспектом є наявність україномовного інтерфейсу.

Проте, не слід забувати і про недоліки системи. Враховуючи те, що додаток призначений швидше для здобувачів освіти старшого шкільного віку, функції моніторингу часу, коли приводять та забирають дитину відсутні. Розклад можна створити вручну або імпортувати з Excel, але генерація розкладу відсутня.

«LineLeader» – платформа, спеціально розроблена для управління дитячими закладами (рис. 1.5).

Функціонал:

– спілкування з сім’ями та персоналом;
– оптимізація виставлення рахунків і платежів;
– звіт відвідуваності;

– безконтактна реєстрація відвідування.

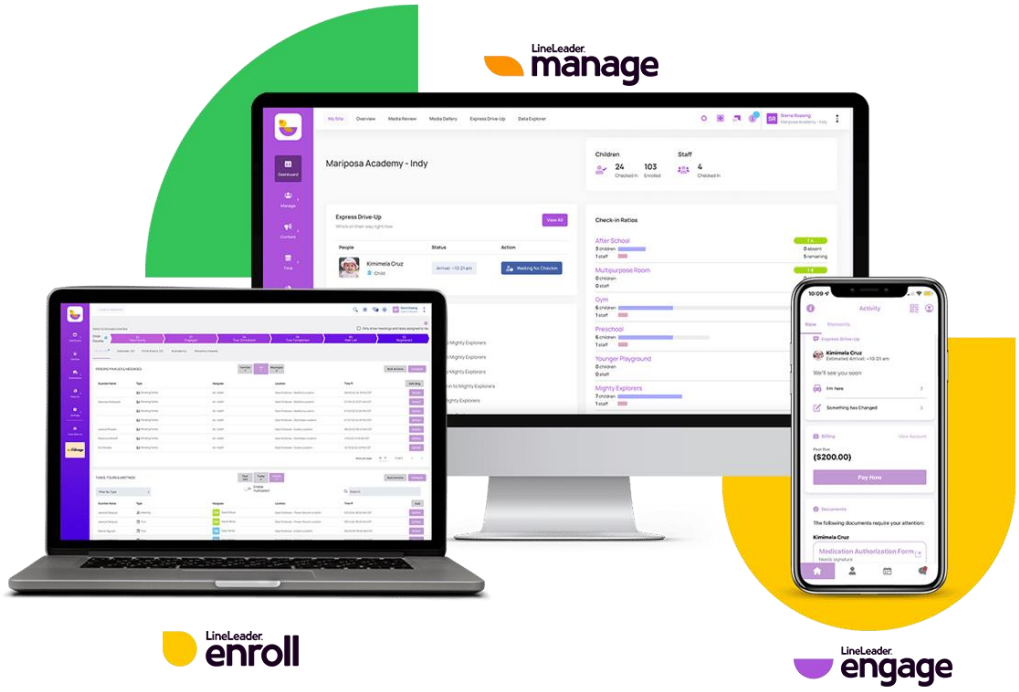


Рисунок 1.5 – Інтерфейс системи LineLeader (за даними [8])

Було виокремлено такі переваги, що позитивно впливають на заклад освіти та навчальний процес в ньому. Інтерфейс виконаний у приємних кольорах та є більш зручним для керівників закладами дошкільної освіти. Батьки можуть отримувати сповіщення через додаток для батьків і виконувати реєстрацію присутності дитини самостійно.

Було виокремлено також і суттєві недоліки які можуть створити умови для пошуку альтернативних рішень. Наприклад, платформа зосереджена більше на управлінні закладом освіти, аніж на плануванні режиму дня та харчування дітей. Таким чином відсутнє відстеження режиму дня здобувачів освіти, а відмітка про присутність здійснюється батьками самостійно, за допомогою відповідних кнопок у додатку, що не виключає помилкових відміток. Україномовний інтерфейс відсутній.

1.2.3 Висновки з порівняльного аналізу

Після проведення аналізу функціональних можливостей застосунків-конкурентів, створено порівняльну таблицю (див. додаток Г), що демонструє наочно наявність різного функціоналу для кожного із програмних продуктів, що розглядаються.

Із наведеної таблиці чітко видно, що кожен із конкурентів реалізує лише частину того основного функціоналу, який міститиметься у системі, що розробляється, тому вона стане достатньо конкурентоспроможним гравцем на ринку, ефективно вирішуючи велике коло проблем користувачів.

При цьому, вона буде ефективним рішенням, що дозволить комбінувати як аспекти контролю режиму дня та харчування, подачі заявок, сплати рахунків так і видачі дітей. Особливою перевагою є можливість користуватися українською мовою. Обмежена кількість конкурентів наразі надають цей функціонал частково.

1.3 Постановка задачі

Проаналізувавши наявні програмні рішення та їхні недоліки, було зроблено висновок, що вони пропонують корисний функціонал у частковому чи обмеженому об’ємі. З цього випливає необхідність створення програмної системи, яка б поєднувала можливості, що зараз пропонують розглянуті додатки. Доречно також, при проєктуванні зосередити увагу на недоліках конкурентів з метою створення ефективного програмного продукту, який забезпечить потреби організації освітнього середовища дошкільного закладу.

Для здійснення цієї задачі наша програмна система повинна містити такий функціонал:

- реєстрація та авторизація;
- реєстрація та авторизація за допомогою Google акаунта;
- генерація розкладу для дітей, з урахуванням фізіологічних особливостей;
- зміни вихователями режимів дня;
- контроль батьків над процедурою передачі дітей довіреним особам;

– автоматичне сповіщення вихователів про відсутність дитини;

– електронний журнал відвідуваності з часовими рамками;

– перегляд докладної інформації про режим дня та активності дитини;

– оплата рахунків (батьками);

– подача заявок до закладу освіти;

– керування заявками до закладу освіти;

– налаштування завідувачем певних аспектів закладу освіти.

Для зручності використання системи зареєстровані користувачі поділятимуться на завідуючих, вихователів та батьків й матимуть доступ до різних функцій, залежно від ролі. Завідуючі отримають функції для управлянні закладом освіти, в той час як вихователі – групами. Батьки ж у свою чергу зможуть переглядати розклади дня дітей, але лише вихователі зможуть їх редагувати. Також батьки зможуть здійснювати оплату рахунків та подачу заявок до дитсадків.

Для розробки системи використовуються мови програмування C#, TypeScript, ASP.NET (фреймворк для створення веб-додатків на серверному боці), React.js (бібліотека багаторазового використання компонентів JavaScript) та React Native (фреймворк для розробки кросплатформених програм).

Для роботи з даними було обрано Microsoft SQL Database, що хоститься на Azure.

2 ФОРМУВАННЯ ВИМОГ ДО ПРОГРАМНОЇ СИСТЕМИ

2.1 Виявлення та аналіз вимог

Вже неодноразово перевірено й широко визнано, що якісна інженерія вимог сприяє підвищенню рівня якості програмного забезпечення і суттєво зменшує ризик невдачі чи надмірного витрачання бюджету на розробку програмного забезпечення [9, 10].

Сукупність характеристик властивостей, якості та функцій програмного забезпечення, яке знаходиться на етапі розробки або модернізації називають вимогами до ПЗ. Виявлення вимог є першим етапом у процесі визначення вимог. На ньому проводиться збір інформації про побажання замовника та межі застосування майбутнього програмного продукту [11].

Для проведення якісної інженерії вимог важливим фактором є визначення й аналіз цільової аудиторії. Це дозволить сформувати вимоги, які б враховували потреби майбутніх користувачів, що в майбутньому може підвищити затребуваність продукту на ринку. Результати аналізу, для якого було використано методом 5W, наведено в таблиці 2.1.

Таблиця 2.1. Результати сегментації цільової аудиторії.

Запитання	Група 1	Група 2	Група 3
Хто? (Who?)	Керівники дошкільних закладів освіти	Педагогічний колектив закладів дошкільної освіти	Батьки або опікуни дітей дошкільнят
Що? (What?)	Додаток, що надає функції автоматизації обробки заявок	Додаток, що надає функції генерації розкладу та відстеження відвідування	Додаток, призначений для відстеження активностей дітей, відвідування
Де? (Where?)	В ЗДО	В ЗДО	В ЗДО
Коли? (When?)	Використовується щодня в робочий час	Використовується протягом усього навчального року	Використовується щодня
Чому? (Why?)	Для підвищення ефективності управління та адміністративної діяльності	Для підвищення якості навчального процесу та гарантування безпеки дітей	Для забезпечення участі в навчальному процесі, зручного отримання інформації, убезпечення дітей

Після успішного командного аналізу цільової аудиторії необхідно переходити до документування вимог до програмного продукту, етапу специфікації вимог [11].

2.2 Функціональні вимоги до програмної системи

Функціональні вимоги описують внутрішні процеси програмного забезпечення, його поведінку, включаючи обчислення, проведення маніпуляцій з даними, обробка даних й інші специфічні функції [12].

Загалом, програмний продукт має задовольняти всі затверджені функціональні вимоги, які наведено нижче.

2.2.1 FR-001 Реєстрація користувачів

Нові користувачі повинні мати можливість створювати акаунти в системі та реєструватися за допомогою Google акаунта.

Для цього вони мають заповнити усі обов’язкові поля для реєстрації. Сервер повинен створити нового користувача. В разі помилки виводиться повідомлення про це.

2.2.2 FR-002 Авторизація користувачів

Зареєстровані користувачі повинні мати можливість використовувати створені акаунти для входу.

Для цього вони мають бути заповнені усі обов’язкові поля для входу. Сервер повинен перевірити чи існує такий користувач і в разі його відсутності необхідно вивести відповідне повідомлення. Повідомлення про помилку також має бути виведено в разі використання неправильного пароля.

2.2.3 FR-003 Генерація розкладу

Вихователі повинні мати можливість згенерувати розклад дня дітей, обравши активності, які необхідно включити. При чому, обов’язкові активності не можна буде прибрати з переліку обраних. В результаті цього на екрані має відобразитись

згенерований розклад та кнопка для його повторної генерації. У випадку виключної ситуації має бути виведено повідомлення про помилку.

2.2.4 FR-004 Зміна розкладу

Вихователі повинні мати можливість змінити розклад дня дітей, використовуючи вже наявні в розкладі активності. В результаті цього на екрані має відобразитись оновлений розклад. У виняткових випадках система має показати повідомлення про помилку.

2.2.5 FR-005 Перегляд журналу відвідування групи

Вихователі повинні мати можливість переглядати журнал відвідуваності групи дітьми за будь-який обраний день, якщо такі відомості за цей день існують. В протилежному випадку (наприклад, якщо обрано день в майбутньому) на екрані відображається повідомлення про відсутність запитуваних даних. Записи в журналі мають відображати присутність чи відсутність дитини в закладі освіти та містити часові рамки в разі присутності.

2.2.6 FR-006 Перегляд інформації про розклад дитини

Батьки повинні мати можливість переглядати розклад групи їхньої дитини за будь-який обраний період, в одному з режимів (місяць, тиждень, день, табличний вигляд), якщо такий розклад існує і був попередньо згенерований вихователем. В протилежному випадку (розклад не було згенеровано, або дитина ще не належить до групи) на екрані з’являється повідомлення про відсутність розкладу для цієї групи.

2.2.7 FR-007 Перегляд інформації про відвідування дитини

Батьки повинні мати можливість переглядати інформацію про відвідування групи їхньої дитиною за будь-який обраний період, якщо така інформація існує. В протилежному випадку (дитина ще не належить до групи) на екрані з’являється повідомлення про відсутність запитуваних даних.

2.2.8 FR-008 Перегляд рахунків

Батьки повинні мати можливість переглядати інформацію про рахунки, згенеровані системою, що базуються на відвідуванні групи їхньою дитиною, якщо така інформація існує. В протилежному випадку (дитина ще не належить до групи, або не відвідує заклад освіти) рахунки не генеруються та не відображаються.

2.2.9 FR-009 Оплата рахунків

Батьки повинні мати можливість сплачувати рахунки, згенеровані системою. Для цього вони мають натиснути відповідну кнопку та бути переадресованими на сайт платіжної системи. У випадку помилки, виводиться відповідне повідомлення.

2.2.10 FR-0010 Створення профілю дитини

Батьки повинні мати можливість створити профіль дитини, прив’язаний до їхнього акаунта. Для цього вони мають надати всю необхідну інформацію системі. У випадку помилки, виводиться відповідне повідомлення.

2.2.11 FR-0011 Подача заявки до закладу освіти

Батьки повинні мати можливість подати заявку на вступ до закладу дошкільної освіти, обравши відповідно дитину, на ім’я якої й буде створено заявку. Для цього вони попередньо мають обрати дитячий садок. У випадку, якщо набір у освітній заклад закрито, не обрано дитину, чи відсутні діти, прив’язані до батьківського акаунта кнопка для надсилання заявки неактивна. В разі успіху система має проінформувати про успішне створення заявки батьків.

2.2.12 FR-0012 Керування заявкою до закладу освіти

Батьки повинні мати можливість скасувати надсилання, прийняти або відхилити оброблену керівником ЗДО заявку. У випадку виключної ситуації має бути виведено повідомлення про помилку.

Керівники ЗДО повинні мати функціонал для перегляду деталей заявки, прийому або відхилення чи коментування заявки. У випадку виключної ситуації має бути виведено повідомлення про помилку.

2.2.13 FR-0013 Додавання дитини в групу

Керівники ЗДО повинні мати можливість додати дитину в групу, обравши одну з-поміж переліку груп садочка, відповідних до віку дитини. Фактичне додавання здійснюється після підтвердження прийому заявки батьками. У випадку помилки, виводиться відповідне повідомлення.

2.2.14 FR-0014 Перегляд заявок

Батьки повинні мати можливість перегляду всіх поданих заявок. У випадку відсутності заявок має бути повідомлення про це.

Керівники ЗДО повинні мати змогу переглядати всі заявки, подані в обраний ЗДО. У випадку відсутності заявок має бути повідомлення про це.

2.2.15 FR-0015 Перегляд груп закладу освіти

Керівники ЗДО повинні мати змогу переглядати всі групи, що належать до обраного ЗДО. В разі відсутності груп має бути повідомлення про це.

2.2.16 FR-0016 Перегляд активностей закладу освіти

Керівники ЗДО повинні мати можливість перегляду всіх активностей, доступних до включення в розклад груп. В разі відсутності власних активностей закладу освіти будуть відображатися лише базові активності, надані системою.

2.2.17 FR-0017 Перегляд налаштувань активностей закладу освіти

Керівники ЗДО повинні мати змогу переглядати налаштування активностей, що належать до обраного ЗДО.

2.2.18 FR-0018 Оновлення налаштувань активностей закладу освіти

Керівники ЗДО повинні мати можливість оновлення налаштування активностей, що належать до обраного ЗДО. В разі утворення виключної ситуації бути відображене повідомлення про це.

2.2.19 FR-0019 Перегляд списку дітей

Батьки повинні мати можливість перегляду всіх дітей, прив’язаних до їх акаунту. У випадку відсутності дітей має бути відображене повідомлення про це та запропоновано додати дитину до акаунта.

У той час, як функціональні вимоги установлюють, що програмний проєкт повинен робити, нефункціональні визначають, яким він має бути. Це є не менш важливим аспектом, тому перейдемо до специфікації нефункціональних вимог.

2.3 Нефункціональні вимоги до програмної системи

Нефункціональні вимоги задають атрибути якості функціонування програмного продукту [13]. Загалом, вони спрямовані на поліпшення безпеки, надійності, швидкодії та зручності використання програмного забезпечення, а також на вдосконалення його властивостей, таких як масштабованість і відновлюваність [11].

В ході роботи над розроблюваним програмним забезпеченням, було сформульовано такий перелік нефункціональних вимог:

- продуктивність і масштабованість (основна сторінка має завантажуватись не довше, ніж протягом 5 секунд в браузері Google Chrome та відображати вміст у відповідний спосіб);
- надійність (при аварійному завершенні роботи додаток повинен зберегти дані);
- ремонтопридатність (ймовірність виправлення помилки в компоненті за 24 години має становити не менше 80%);
- портативність та сумісність (система має підтримуватись у браузерах

Google Chrome (останнє оновлення), Microsoft Edge (останнє оновлення), Safari (останнє оновлення) та операційних системах Windows, Mac OS X, Android 6.0+ та iOS 13.4+);

– доступність (програмний продукт повинен бути доступний для користувачів з України 98,8 % часу в будь-який час доби);
– безпека (веб-система має гарантувати, що введені користувачем дані будуть приватними);
– локалізація (формат дати має відповідати наступному «ДД.ММ.РРРР» (наприклад 28.04.2024), а ціна має бути відображена у гривнях (₴), мови інтерфейсу: українська, англійська);
– інтернаціоналізація (програмний продукт повинен працювати як в Україні, так і в країнах ЄС, США з доступом до англомовної версії користувацького інтерфейсу).

3 АРХІТЕКТУРА ТА ПРОЄКТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 UML проєктування ПЗ

Перш ніж перейти до безпосередньої розробки застосунку, необхідно чітко визначити архітектуру програмного забезпечення. Саме забезпечення розробників інструментами для аналізу, проєктування та впровадження і є метою UML [14].

Таким чином, в рамках проєктування програмної системи було створено низку UML-діаграм: діаграми варіантів використання та діаграми зміни станів для чіткого визначення функціоналу з боку користувача.

Програмна інформаційна система для закладів дошкільної освіти з керуванням режимом дня харчуванням дітей передбачає 4 типи акторів: неавтентифікований користувач, керівник ЗДО, вихователь, батьки.

В залежності від ролі, користувачі отримують доступ до різних функцій, що надаються системою. З урахуванням цих умов, було розроблено діаграми варіантів використання.

UML діаграма варіантів використання – це засіб для визначення вимог систем, тобто того, що системи мають робити, тобто це своєрідна специфікація поведінки [14].

Таким чином, неавтентифікований користувач отримує доступ лише до перегляду переліку закладів дошкільної освіти в цілому та кожного окремо (див. рис. 3.1). Також можлива автентифікація та авторизація.

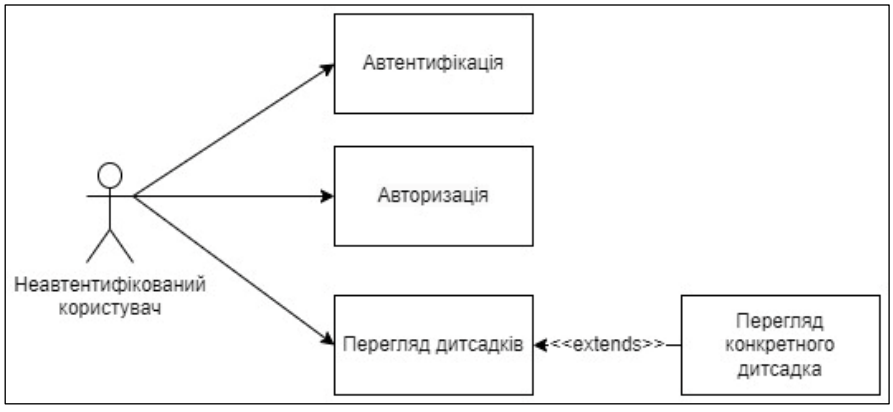


Рисунок 3.1 – Use-case діаграма застосунку «ChildWell» для неавтентифікованого користувача (рисунок виконаний самостійно)

Керівник ЗДО має ширші можливості (див. рис. 3.2). Серед них перегляд заявок, та керування ними, оновлення налаштувань активностей у ЗДО, а також додавання дитини в групу.

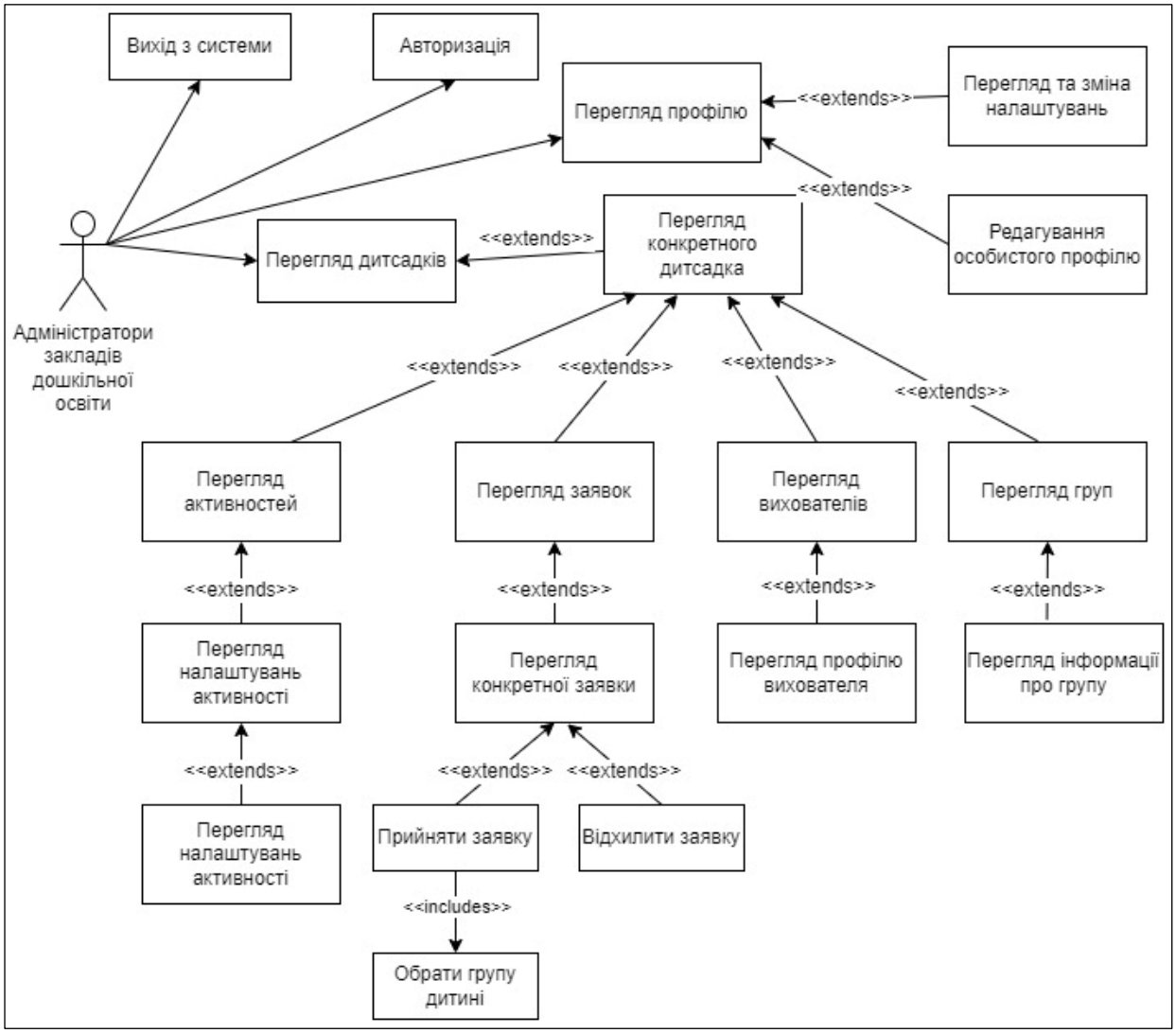


Рисунок 3.2 – Use-case діаграма застосунку «ChildWell» для керівника ЗДО (рисунок виконаний самостійно)

Вихователь має доступ до функціоналу, пов’язаного безпосередньо з групами. Серед них і перегляд групи, і керування розкладом, а також перегляду журналу відвідування за обраний день (див. рис. 3.3).

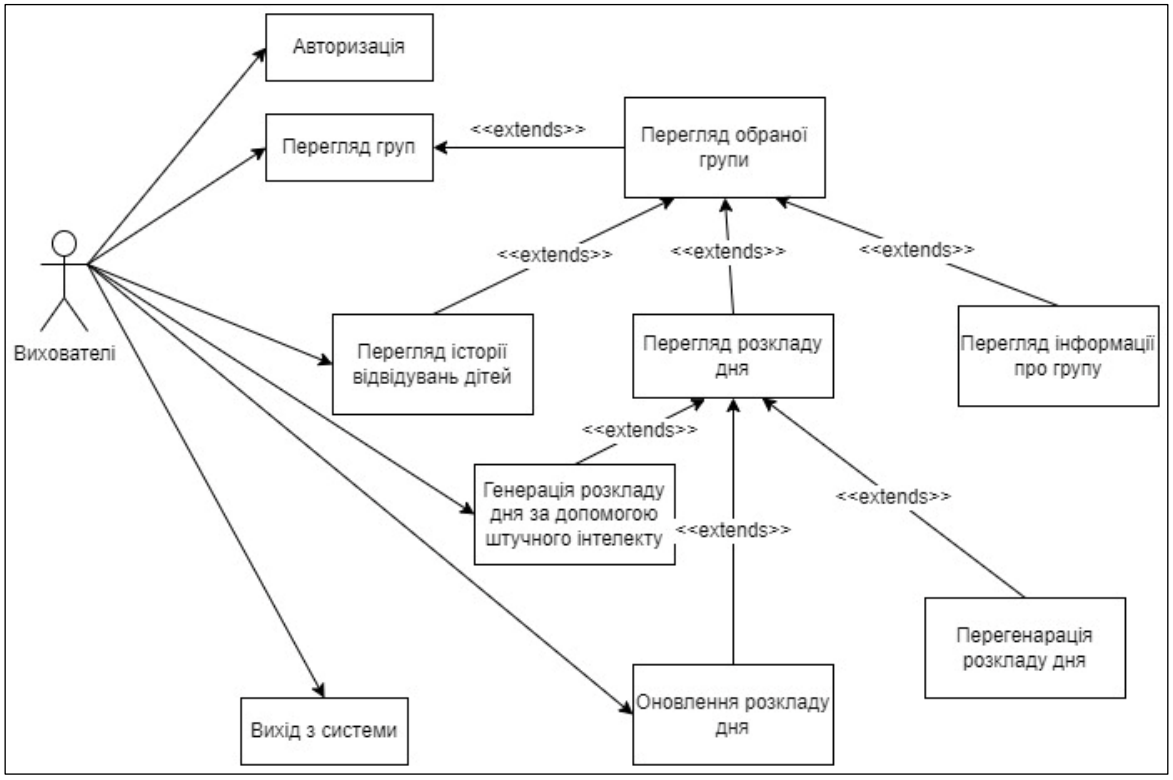


Рисунок 3.3 – Use-case діаграма застосунку «ChildWell» для вихователя (рисунок виконаний самостійно)

Для батьків також надається широкий функціонал (див. рис. 3.4). Він стосується як дітей, так і рахунків та заявок. Зокрема надаються наступні можливості:

- перегляд розкладу дитини;
- перегляд інформації про відвідування дитиною садка;
- перегляд рахунків;
- оплата рахунків;
- перегляд дитячих садків;
- подача заявок в дитячі садки;
- керування заявками;
- перегляд списку дітей;
- створення профілю дитини;
- перегляд інформаційної панелі.

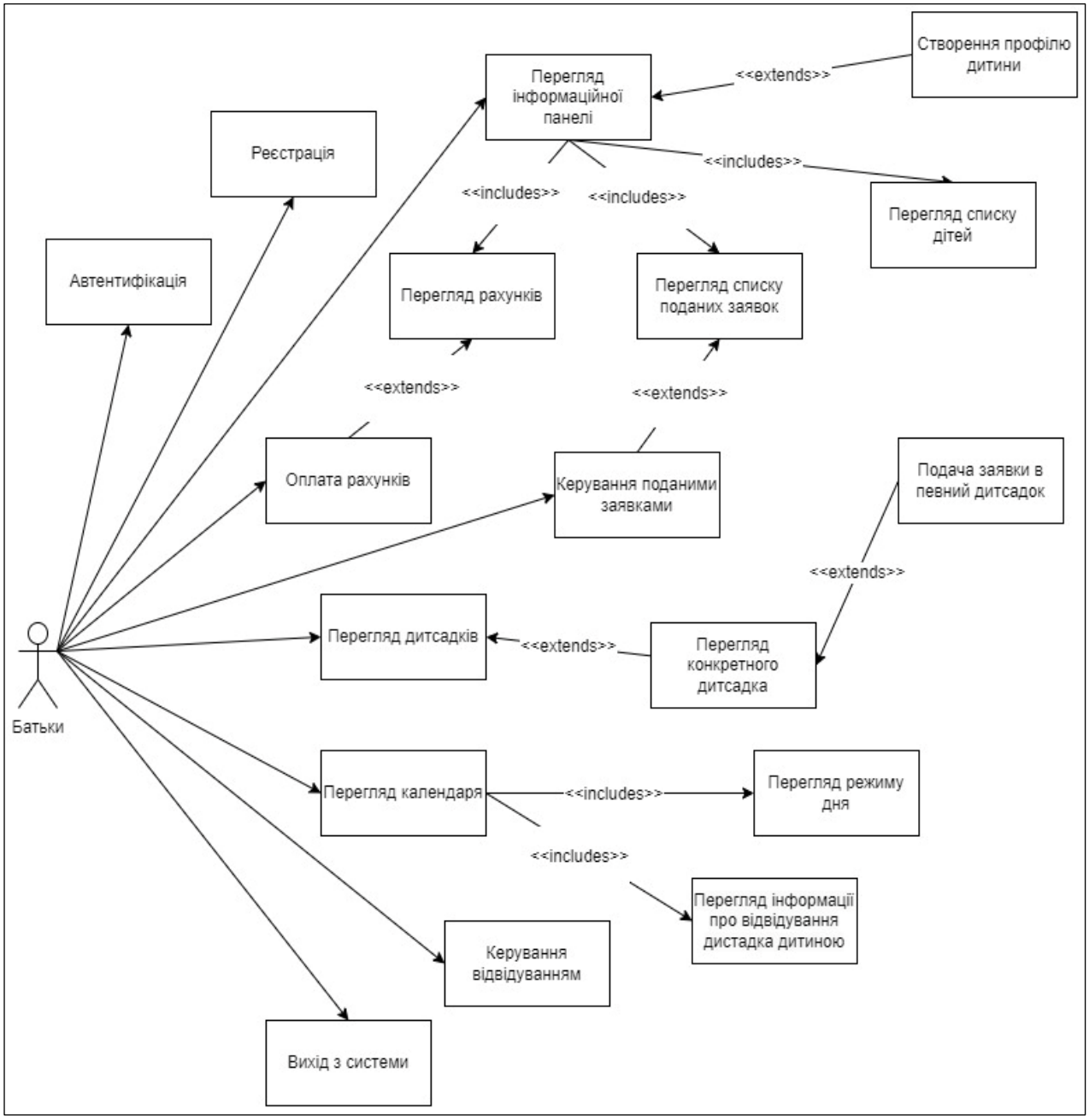


Рисунок 3.4 – Use-case діаграма застосунку «ChildWell» для батьків (рисунок виконаний самостійно)

Діаграми станів і переходів допомагають краще зрозуміти процеси, що відбуваються в системі[14].

Було виділено 2 процеси: генерація розкладу та керування заявками, для яких і було створено UML діаграми станів і переходів.

У розробленому програмному продукті генерація розкладу для групи вихователем відбувається за наступним алгоритмом (див. рис. 3.5):

1. Розкладу не існує;
2. Вихователь генерує розклад;
3. Якщо результат його влаштовує, то розклад сформовано і генерацію розкладу завершено;
4. У протилежному випадку, надається можливість повторно згенерувати розклад;
5. Якщо результат і досі не підходить, можливо відхилити розклад і завершити процес генерації.

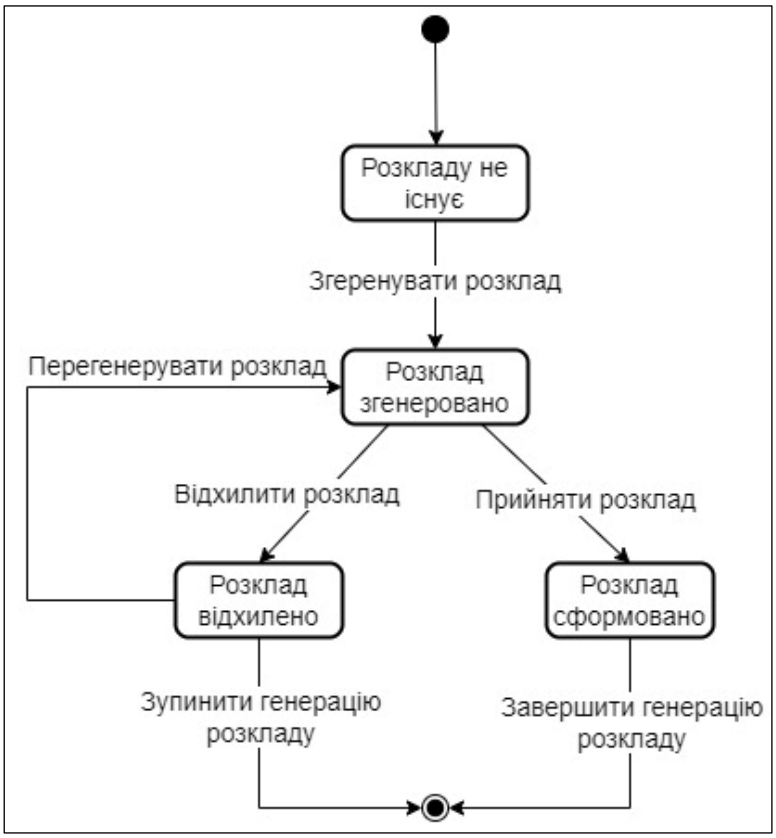


Рисунок 3.5 – Діаграма станів і переходів генерації розкладу (рисунок виконаний самостійно)

Керування заявками є також багатокроковим процесом, як для батьків, так і для керівників ЗДО. Заявки можуть приймати різні стани, які детально відображено на UML діаграмі станів і переходів (див. рис. 3.6).

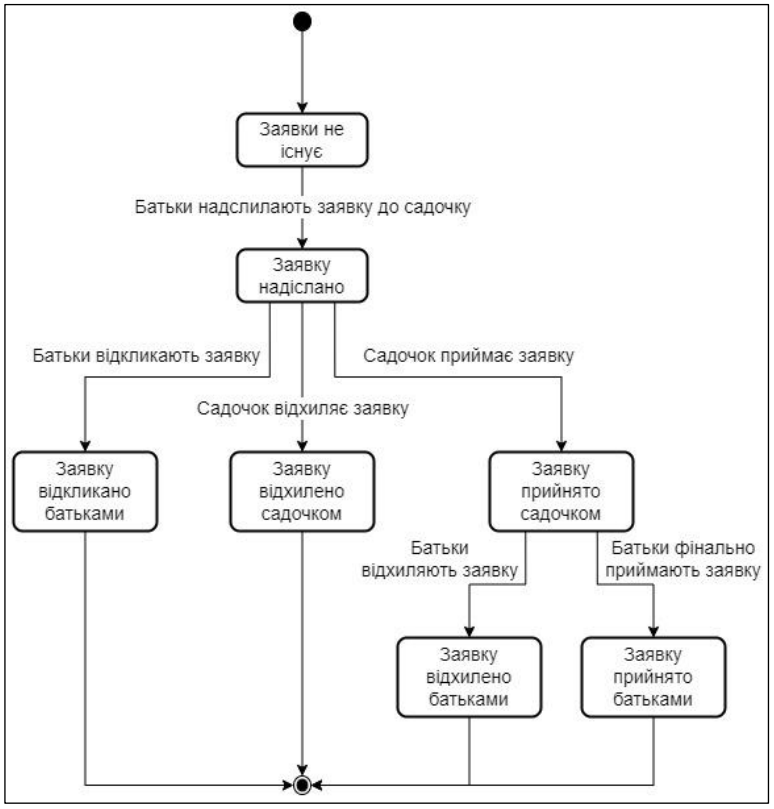


Рисунок 3.6 – Діаграма станів і переходів керування заявок (рисунок виконаний самостійно)

Таким чином, можна визначити наступний алгоритм:

1. Заявки не існує;
2. Батьки надсилають заявку;
3. Батьки можуть відкликати заявку, а керівник ЗДО може або прийняти її, або відхили;
4. Якщо заявку прийнято садочком, батьки або приймають її, або остаточно відхиляють.

Визначення компонентів системи є важливим етапом у процесі розробки. Для програмного продукту «ChildWell» їх налічується 5, а саме 2 кластери Microsoft SQL Server (RESTful API серверу та серверу ідентифікації), сервер ідентифікації «ChildWell Identity», сервер «ChildWell» та веб-клієнт «ChildWell».

Діаграми розгортання фіксують зв’язки між логічними та/або фізичними елементами систем та інформаційними активами, закріпленими за ними [14].

Отже, після створення UML діаграм станів та переходів, було створено діаграму розгортання, що відображає як веб-клієнт взаємодіє з веб-сервером. Її представлено на рис. 3.7.

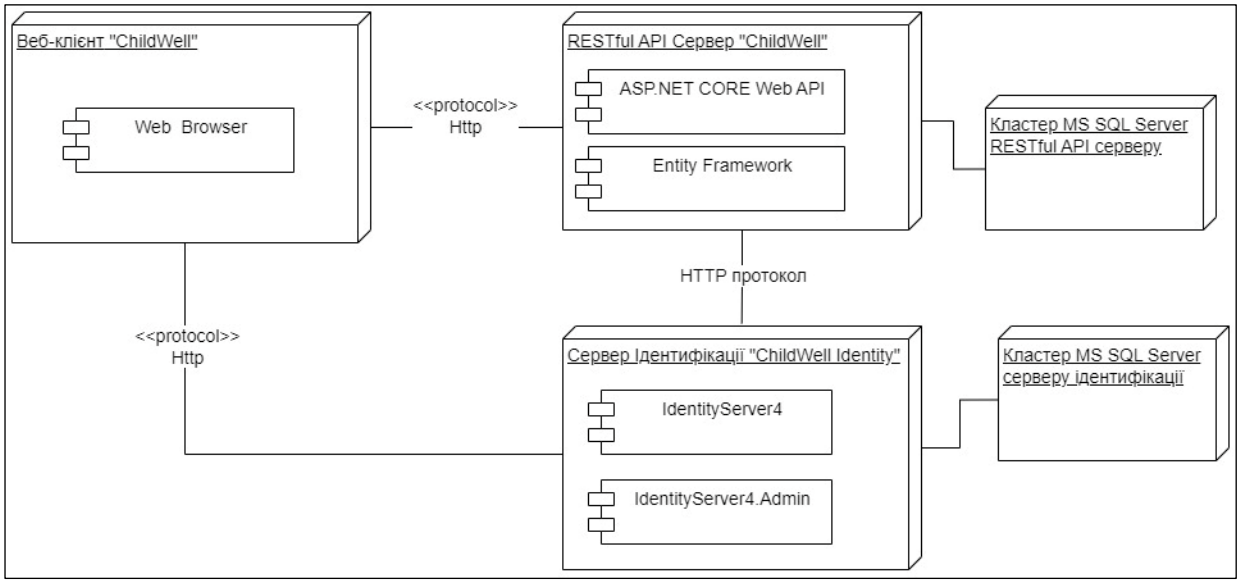


Рисунок 3.7 – Діаграма розгортання програмної системи «ChildWell» (рисунок виконаний самостійно)

Користувачі отримують можливість взаємодії з системою завдяки веб-клієнту, що доступний через браузер. Він взаємодіє з веб-сервером використовуючи протокол HTTP. Так званою «угодою» між клієнтом та сервером виступає ASP.NER Core Web API.

Веб-сервер є центральною частиною системи, оскільки він виконує основну частину бізнес-логіки. З'єднання з базами даних, які зберігають дані системи також встановлюється веб-сервером. Обидві бази розгорнуто в хмарному середовищі Azure.

3.2 Проєктування структури зберігання даних

Проєктування структури зберігання даних стає одним з найвідповідальніших завдань, пов'язаних зі створенням складних інформаційних систем [15].

ER діаграми є зручним інструментом для відображення зв'язків між об'єктами системи. В даній предметній області було виділено такі стрижневі сутності:

- дитячий садок;
- група;
- дитина;
- відвідування;
- рахунок;
- складова рахунку;
- заявка;
- активність;
- категорія активностей;
- налаштування активності;
- запланована активність;
- користувач;
- роль користувача;
- завідувач;
- вихователь.

Зв’язки між ними наведено на створеній ER діаграмі (див. додаток Д). Для ілюстрації обміну «повідомленнями» між програмними частинами системи та базою даних було використано діаграму послідовності.

Для доступу до повного функціоналу системи користувач для початку здійснює автентифікацію завдяки OAuth сервісу. Він після валідації токену надає користувачеві можливість доступу до даних зі сховища, які надаються з допомогою API. UML діаграму послідовності для програмної системи «ChildWell» наведено на рисунку 3.8.

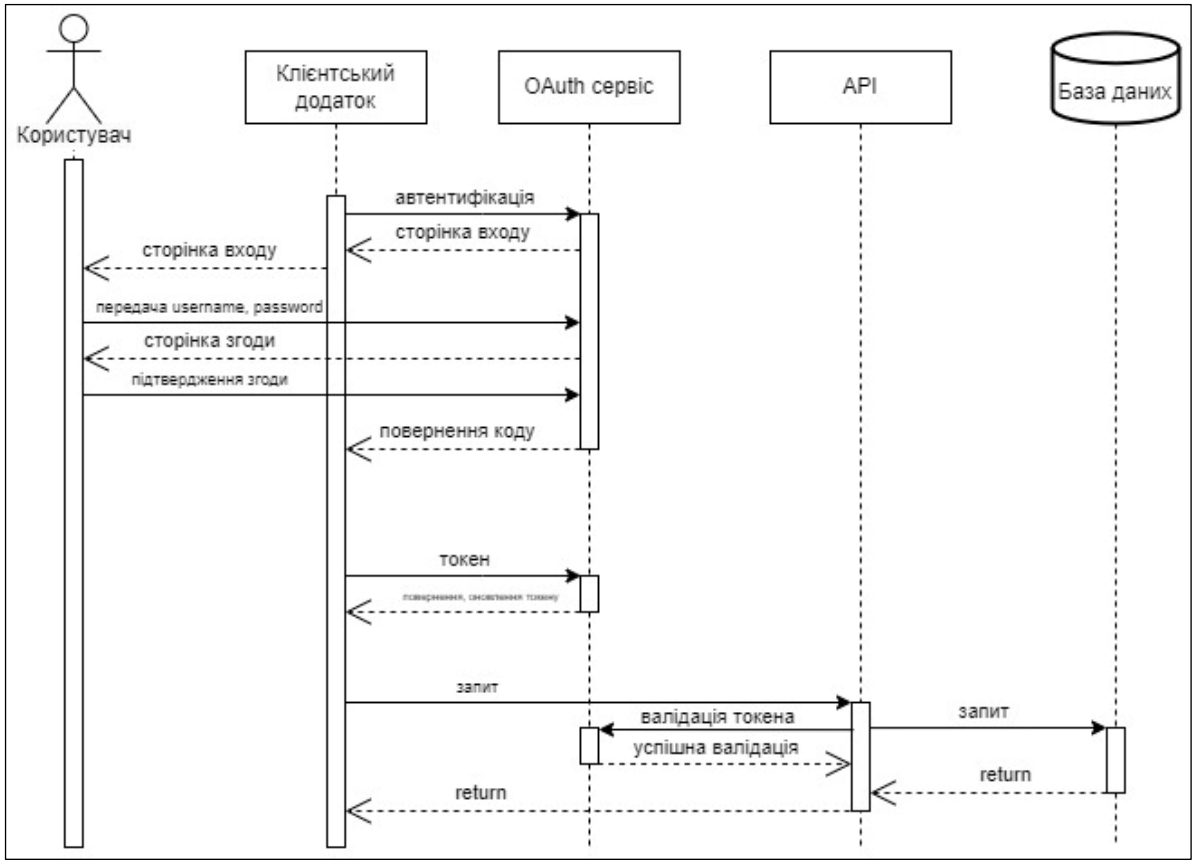


Рисунок 3.8 – Діаграма послідовності програмної системи «ChildWell» (рисунок виконаний самостійно)

Діаграма допомагає побачити загальну картину, яким чином здійснюється інформаційний обмін усередині системи «ChildWell».

3.3 Проєктування архітектури ПЗ

Серед багатьох варіантів серед існуючих архітектур, для реалізації системи обрано архітектурний шаблон, що має назву Клієнт-Сервер.

Основна ідея полягає в поділі функціоналу системи між двома типами додатків: клієнт і сервер. Клієнтом називають додаток, з яким користувач взаємодіє безпосередньо. Сервером є комп’ютер, що розміщує програму, яка містить всю логіку роботи [16].

Особливістю цього підходу є те, що клієнт виступає ініціатором взаємодії, надсилаючи запити до сервера. Останній в свою чергу обробляє надіслані запити та відповідає на них. Кілька клієнтів можуть обслуговуватись одним сервером

одночасно, оскільки запити формують чергу й виконуються по порядку. Таким чином відбувається розділення інтерфейсу та бізнес-логіки.

Правила взаємодії зазвичай описує мережевий протокол. У системі «ChildWell» основним протоколом взаємодії виступає HTTP.

Клієнт-Серверну архітектуру було обрано через низку причин:

– підтримка великої кількості клієнтів;
– централізація контролю безпеки та доступу
– спрощене мережеве адміністрування;
– зниження навантаження на додатки-клієнти;
– відсутня необхідність дублювати код з сервера на клієнті.

Більшість веб-додатків створено на основі Клієнт-Серверної архітектури. Ця архітектура використовується навіть десктопними програмами, що здійснюють взаємодію за допомогою Інтернет [16].

Серверну частину системи «ChildWell» реалізовано за допомогою технології .NET Framework 8.0. Для роботи з базою даних використовується Entity Framework Core. Веб-сервер та клієнт працюють незалежно один від одного.

Клієнтський додаток є додатком типу SPA (Single Page Application). Це означає, що він взаємодіє з користувачем без перезавантаження сторінок і весь контент завантажується разом із запуском додатку й демонструється в рамках однієї сторінки.

Перед створенням клієнтського додатку було проаналізовано різні фреймворки й бібліотеки для розробки користувацького інтерфейсу. У підсумку було вирішено використовувати Java Script бібліотеку React.js, оскільки вона пропонує великий перелік інструментів, компонентів та розширень для швидкої розробки функціональних додатків. Ефективно здійснювати маніпуляції над веб-сторінкою дозволяє віртуальний DOM.

Бібліотека MobX використовується для управління станом додатку React. Завдяки цьому інструменту можливо автоматично відслідковувати зміни стану та оновляти інтерфейс користувача при зміні даних.

У системі «ChildWell» також використовується React Router – інструмент, що призначений для навігації між компонентами та маршрутизації у веб-додатку.

Для дизайну інтерфейсу користувача використовується бібліотека компонентів Material UI з використанням Material Design від Google. За її допомогою було створено тему, яка керує колірною палітрою, шрифтами, виглядом й стилями компонентів. Це дозволяє уніфікувати зовнішній вигляд всіх компонентів інтерфейсу та пришвидшити процес створення UI.

Написання клієнтської частини здійснювалось із використанням Microsoft Visual Studio Code. Система контролю версій Git забезпечила можливість сумісної роботи нас системою.

3.4 Створення UI/UX

Зростання бізнесу та залучення нових клієнтів є головною метою будь-якого бізнесу. Оскільки користувацький інтерфейс є тим, що відвідувач веб-сторінки бачить першим, UI/UX дизайн може відігравати одну з ключових ролей для її досягнення. Наразі користувачам пропонується велика кількість альтернатив серед програмних продуктів, тому інтерфейс користувача може стати інструментом для завоювання довіри клієнтів та переконати їх користуватися саме нашим програмним забезпеченням.

Особливо важливим цей аспект виявляється для стартапів та невеликих компаній, адже UI/UX дизайн може як позитивно, так і негативно вплинути на впізнаваність бренду. Користувацький досвід та інтерфейс користувача впливають на асоціації користувачів з системою та рівень задоволеності від взаємодії з програмним продуктом [16].

Виходячи з цього, значну увагу було приділено розробці дизайну користувацького інтерфейсу. Планування інтерфейсу сторінок почалось з підбору палітри кольорів. Оскільки програмний продукт фокусується на закладах дошкільної освіти, команда ухвалила рішення про використання м’якої та спокійної кольорової гами, яка б асоціювалася з дитинством (див. рис. 3.9).

Рисунок 3.9 – Основна кольорова гамма системи «ChildWell» (рисунок виконаний самостійно)

Оскільки частиною інтерфейсу є календар (див. рис. 3.10), було вирішено, що для простоти сприйняття розкладу, кожна категорія активностей повинна мати власний колір (див. рис. 3.11).

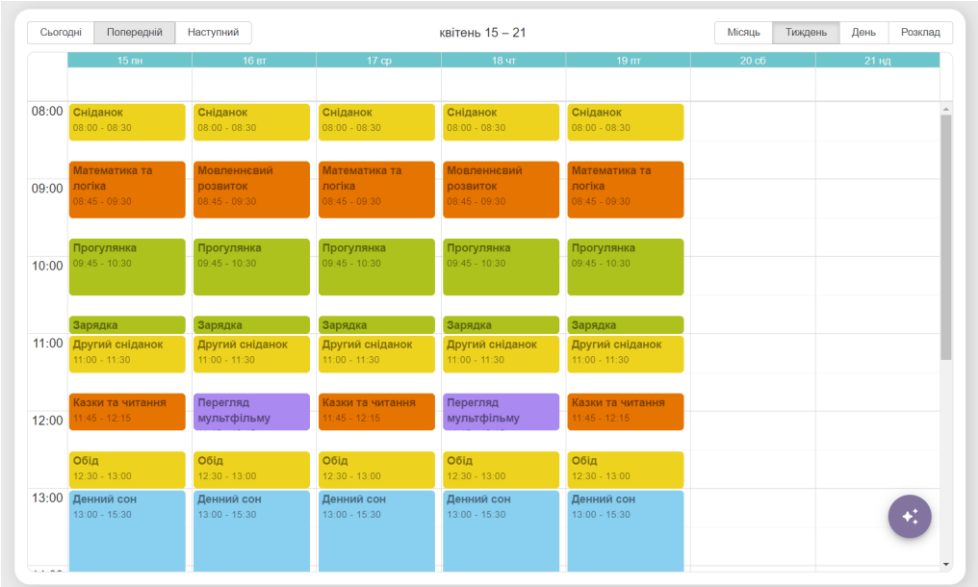


Рисунок 3.10 – Календар у системі «ChildWell» (рисунок виконаний самостійно)

КАТЕГОРІЯ	КОЛІР	HEX КОД
СОН		#89CFF0
ФІЗИЧНА АКТИВНІСТЬ		#AEC21D
ОСВІТА		#E67400
МИСТЕЦТВО		#BF0099
ХАРЧУВАННЯ		#EED31E
ВІДПОЧИНОК		#AA89F0

Рисунок 3.11 – Палітра кольорів для активностей на календарі «ChildWell» (рисунок виконаний самостійно)

Також було створено логотип системи (див. рис. 3.12), який не протирічить кольоровій гаммі системи, а також не вибивається із загальної стилістики сайту.

Рисунок 3.12 – Логотип системи «ChildWell» (рисунок виконаний самостійно)

Завдяки бібліотеці компонентів Material UI, було створено тему, яка дозволяє зберегти стилістичну єдність у зовнішньому вигляді всіх компонентів інтерфейсу (наприклад, кнопок (див. рис. 3.13)).

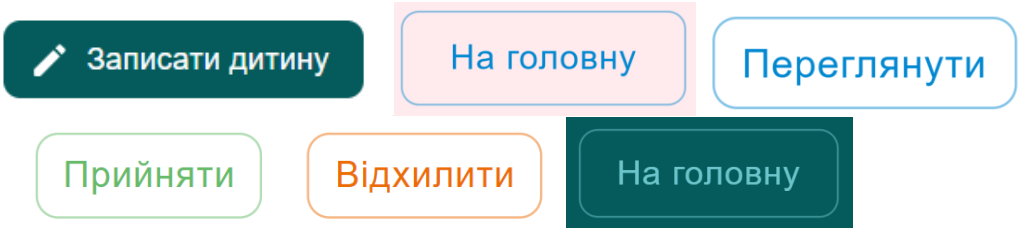


Рисунок 3.13 – Стилістична єдність завдяки використання теми в системи «ChildWell» (рисунок виконаний самостійно)

Для надання особливого користувацького досвіду зареєстрованим користувачам в різних ролях було створено HOC «withAuthentication».

```
export const withAuthentication =
(Component:React.ComponentType<object>) => {

    return withAuthenticationRequired(Component, {
        onBeforeSignin: () => localStorage.setItem("redirectUri",
                                                   window.location.pathname),
        OnRedirecting: () => <LoadingComponent />
    })
}
```

Компонент вищого порядку (Higher-Order Component, HOC) – спосіб для повторного використання логіки. Іншими словами, це «обгортка» навколо компонента, яка додає або модифікує його функціональність [18]. Приклад використання розробленого компонента вищого порядку наведено нижче.

```
export default withAuthentication(
  observer(function ApplicationList() {
}
```

Основною метою HOC «withAuthentication» є перевірка наявності в користувача доступу до тої чи іншої частини додатку. Завдяки цьому забезпечується захист конфіденційних даних, доступних тільки автентифікованим користувачам.

4 ОПИС ПРИЙНЯТИХ ПРОГРАМНИХ РІШЕНЬ

4.1 Вибір технологій для front-end

Стек технологій, використаних у процесі розробки front-end частини програмної системи для закладів дошкільної освіти з керуванням режимом дня та харчуванням дітей налічує мову програмування TypeScript, бібліотеку React.js, бібліотеки React Router, Material UI.

TypeScript – строго типізована мова програмування, яка ґрунтується на JavaScript і надає інструменти будь-якого масштабу [19].

До основних переваг цього рішення можна віднести наступні аспекти:

- ремонтопридатність коду;
- підвищена продуктивність розробки (розширена підтримка IDE, автодоповнення та Intellisense);
- масштабованість;
- раннє виявлення можливих помилок (за рахунок строгої типізації);
- сумісність з бібліотеками та фреймворками JavaScript;
- велика та активна спільнота (внаслідок цього широкий спектр бібліотек, фреймворків та інструментів, які підтримують TypeScript).

Проте навчання може бути трудомістким та довготривалим процесом, оскільки TypeScript оперує новими концептами, порівняно з JavaScript, наприклад типи, інтерфейси. Початково розробники можуть відстежувати дуже повільний прогрес, особливо у великих проєктах. TypeScript також може потребувати більшого обсягу написаного коду.

React.js – бібліотека, що дозволяє створювати інтерфейси користувача з окремих частин, які називаються компонентами [20]. Бібліотеку було обрано відповідно до потреб нашого застосунку та із урахуванням плюсів React.js, зокрема:

- компонентна архітектура;
- віртуальний DOM;
- синтаксис JSX;
- сильна спільнота та екосистема;

– кросплатформена розробка.

На традиційних веб-сайтах браузер запитує документ із веб-сервера, завантажує ресурси CSS і JavaScript і відображає HTML, надісланий із сервера. Коли користувач натискає посилання, процес починається заново для нової сторінки. React Router забезпечує «маршрутизацію на стороні клієнта», що дозволяє програмі оновлювати URL-адресу після натискання посилання без повторного запиту на інший документ із сервера. Замість цього програма може негайно відтворити новий інтерфейс і оновити сторінку новою інформацією. Це забезпечує швидшу взаємодію з користувачем, оскільки веб-браузеру не потрібно запитувати повністю новий документ або повторно завантажувати ресурси CSS і JavaScript для наступної сторінки [21].

Корисними для нашого проєкту аспектами є:

– різноманітні види маршрутизації (декларативна, динамічна, вкладена);

– взаємодія з параметрами url-адреси;

– інтеграція з функціями React;

– статична генерація сайту.

MobX – бібліотека на основі сигналів, що дозволяє керувати станом програми незалежно від структури інтерфейсу користувача [22].

Для нас важливі такі особливості бібліотеки:

– простота і легкість використання;

– реактивне програмування;

– масштабованість;

– легке управління станом додатку.

Material-UI (MUI) – фреймворк, що пропонує повний набір безкоштовних інструментів для розробки інтерфейсу користувача. MUI обрано для створення стильного та сучасного користувацького інтерфейсу, який відповідає принципам Material Design (від компанії Google) [23] .

Важливим для нашого програмного продукту є те, що фреймворк надає можливості перевикористати наявні компоненти або створити власні на основі розробленого дизайну [24]. Вагомими чинниками є :

– багатий набір компонентів;
– налаштування та оформлення тем;
– адаптивний дизайн;
– послідовний UI/UX;
– надійна документація та підтримка спільноти.

Бібліотека i18next використовується для додавання підтримки багатомовності в додатку. Завдяки їй здійснюється переклад текстових елементів інтерфейсу на дві мови: українська та англійська, динамічно змінюється мова додатку.

Спираючись на результати аналізу переваг та недоліків обраних інструментів, було проведене проєктування архітектури системи.

4.2 Архітектурні рішення

Важливим кроком у процесі розробки програмного забезпечення є проєктування архітектури. Таким чином, було ухвалено низку архітектурних рішень для front-end частини програмної системи для закладів дошкільної освіти з керуванням режимом дня та харчуванням дітей.

По-перше, було обрано Single Page Application (SPA) архітектуру. Цією концепцією передбачено, що програма працює як одна веб-сторінка. Доречним є те, що шар представлення (presentation layer) відокремлений від сервера та керується з браузера [25].

До переваг односторінкових додатків над традиційними веб-додатками також можна віднести наступні:

– відтворення, подібне до відтворення настільної програми, але робота в браузері;
– менші накладні витрати;
– менший час очікування користувача;
– спрощена підтримка коду [25].

Отже, враховуючи описані переваги, клієнтський застосунок «ChildWell» було створено як SPA.

Підхід до розробки ПЗ, при якому додаток складається з окремих, незалежних частин, які можна використовувати повторно, називається компонентною архітектурою. Цей вибір є популярним для розробки веб-додатків з використанням фреймворків React.js, Vue.js, Angular. Це спричинено рядом переваг.

По-перше, можливе перевикористання компонентів. Таким чином, компонентна архітектура забезпечує дотримання принципу DRY (Don’t Repeat Yourself [26]).

По-друге, функціонал розподілено між різними компонентами, що працюють незалежно і взаємодіють через чітко визначені інтерфейси. Більш того, спрощується підтримка додатку, оскільки він розбивається на невеликі відокремлені керовані частини.

Для управління станом у веб-додатках використовуються різні бібліотеки. Для розробки запропонованого програмного продукту «ChildWell» було обрано бібліотеку MobX. Завдяки простому API автоматично відстежуються зміни й оновлюється інтерфейс відповідно до них, це і робить MobX потужним інструментом для розробки нашого додатку.

Контейнером для стану бізнес-логіки додатку виступає MobX store. У MobX store зберігаються дані та декларуються методи взаємодії з ними. Код такого store, розробленого для застосунку «ChildWell» наведено у додатку Е.

4.3 Інтеграція з Back-end

Важливим етапом є визначення інструментів для взаємодії front-end і back-end частин програмної системи.

У межах нашого проєкту протоколом для передачі даних було обрано безстановий (stateless) протокол прикладного рівня HTTP (HyperText Transfer Protocol), оскільки він є загальноприйнятим стандартом [27]. Завдяки йому в нашій системі забезпечено надійну та безпечну взаємодію між клієнтом та сервером.

HTTP також є основою для створення RESTful API для здійснення CRUD-операцій. За допомогою запитів GET, POST, PUT, DELETE здійснюється

маніпуляція ресурсами на сервері. Дані між клієнтським та серверним додатком передаються у JSON форматі, який є компактним і простим для парсингу.

Для здійснення HTTP-запитів з клієнтського застосунку було використано бібліотеку Axios завдяки підтримці асинхронних запитів та широкому фунціоналу. Зокрема Axios автоматично перетворює дані у JSON форматі в зручну для роботи структуру, таку як об'єкт TypeScript, що дозволяє легко отримувати доступ до різних властивостей цих даних. Коли викликається метод Axios, у відповідь повертається promise [28], що дозволяє керувати асинхронними запитами та їх результатами.

4.4 Інтерфейс користувача

Цілісний дизайн допомагає покращити досвід користувача, оскільки він прямо впливає на простоту використання та привабливий зовнішній вигляд веб-додатків. Виходячи з цього, при створенні клієнтського застосунку «ChildWell» значну увагу було приділено розробці дизайну інтерфейсу.

Ми керувалися принципами Material Design від Google, оскільки ця система вказівок, компонентів і інструментів надала найкращі методи розробки інтерфейсу користувача [23].

Material Design бере за основу фізичний світ та його об’єкти, включно з папером та чорнилом. Запропоновані вище принципи також ґрунтуються на типографіці. Зокрема сітки, масштаб, колір та контраст використовуються для створення ієрархії та акценту. Перевагою такого підходу є те, що застосування знайомих атрибутів допомагає користувачам швидко зрозуміти функціональні можливості нашого програмного продукту.

Фреймворк MUI містить компоненти, що створені за принципами Material Design та які можна перевикористати у власному клієнтському додатку. Важливим для нашого проєкту є наявність компонентів, які задовольняють низку потреб при розробці інтерфейсу, наприклад:

- впорядкування вмісту (компоненти Card, Box, Stack, List, Grid);
- навігація додатком (компоненти Link, Stepper, Menu);

- керування (компоненти Button, Checkbox, Form);
- надання інформації (компоненти TextField)
- відображення інформації (компоненти Badge, CircularProgress, Table, Typography, Avatar).

Враховуючи те, що MUI радить передавати значення за допомогою кольору, він став важливим елементом дизайну в ході створення користувацького інтерфейсу. Колір допоміг акцентувати увагу, виділити важливу інформацію, відобразити стан елементів.

Визначальним для створення front-end частини є наявність теми, як унікального концепту у фреймворці MUI. Тему використано для налаштування параметрів дизайну. В ході роботи над UI додатку «ChildWell» було розроблено MUI тему (див. додаток Ж), що задає основні та вторинні кольори, кольори станів помилки й успіху, стилі для кнопок, поділок та комірок таблиць, а також шрифти. Таким чином було використано можливість досягти гармонійного та естетичного вигляду додатку, зменшити кількість повторюваного коду, пришвидшити внесення змін у зовнішній вигляд компонентів.

Під час створення front-end частини програмної системи було вирішено зробити акцент на діях користувача, щоб основні функції були очевидними та користувачеві надавалися орієнтири. Це допомогло підвищити ефективність використання за рахунок зменшення кількості помилок та часу для виконання задач. Значну увагу приділено тому, щоб інтерфейс був інтуїтивно зрозумілим для всіх груп користувачів.

Одним з прикладів реалізації цього є навігаційна панель з чіткими назвами розділів (див. рис. 4.1), такими як «Заявки», «Групи», «Персонал» та релевантними іконками (у варіанті для керівника або адміністратора ЗДО).

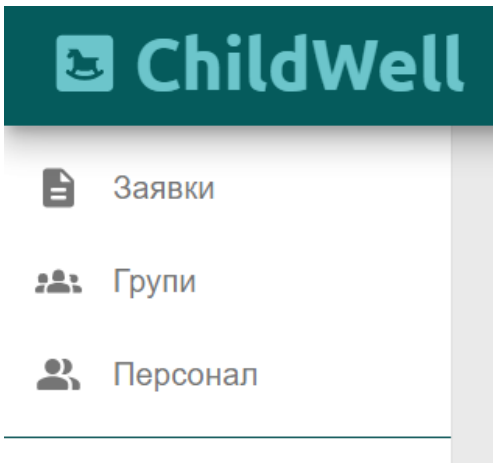


Рисунок 4.1 – Навігаційна панель системи «ChildWell» у варіанті для керівника або адміністратора ЗДО (рисунок виконаний самостійно)

Завдяки кільком ключовим аспектам інтуїтивно зрозумілими також є форми в нашому додатку. Наприклад розглянемо форму прийняття садочком заяви на вступ (див. рис. 4.2). Чіткий заголовок вказує на мету форми, поле для вибору групи забезпечує простий та зрозумілий механізм вибору відповідної групи зі списку. Поле для коментаря має назву та дає змогу ввести додаткову інформацію. Кнопки однозначно визначають можливі дії користувача. Всі елементи форми розташовані логічно і зручно для швидкого заповнення. Таким чином користувачі знають, які дії виконувати та в якому порядку.

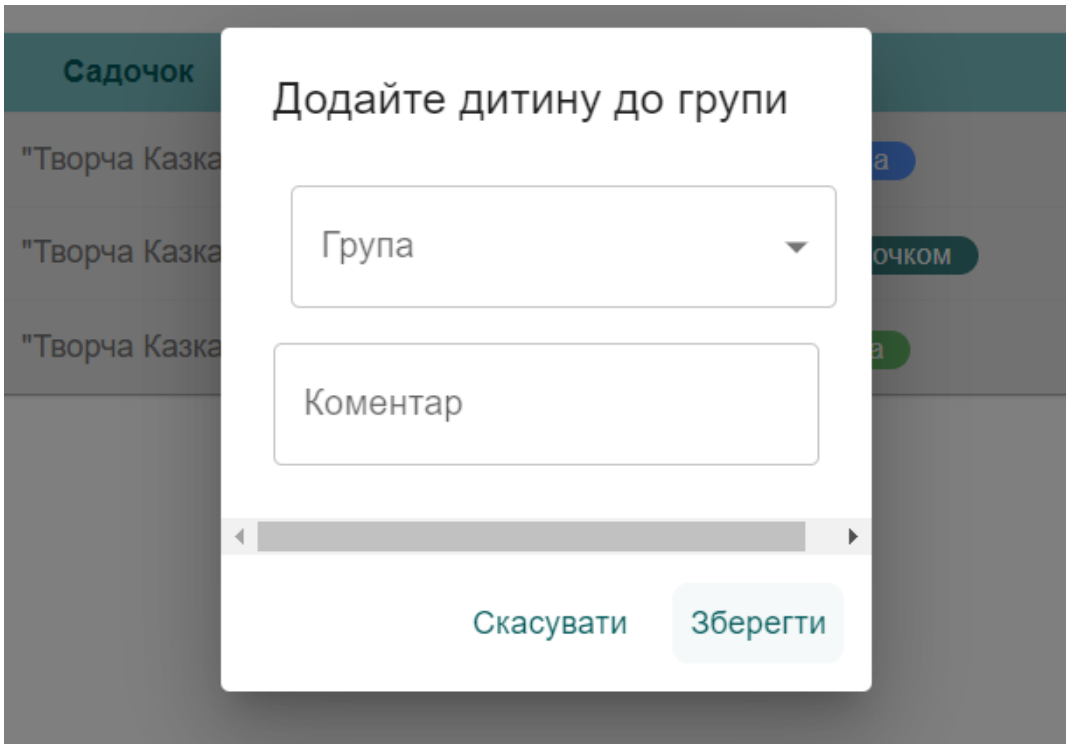


Рисунок 4.2 – Форма прийняття садочком заяви на вступ (рисунок виконаний самостійно)

У ході роботи увагу було зосереджено на тому, щоб календар (див. рис. 4.3), який є одним з центральних елементів інтерфейсу, був якомога простіший у використанні, але водночас гнучким та інформативним. Назви кнопок допомагають зрозуміти їх призначення, поточний день виділено кольором, а також присутня поділка, що відображає поточний час й активність. Загальний зовнішній вигляд календаря є стандартним, а значить знайомим для користувачів.

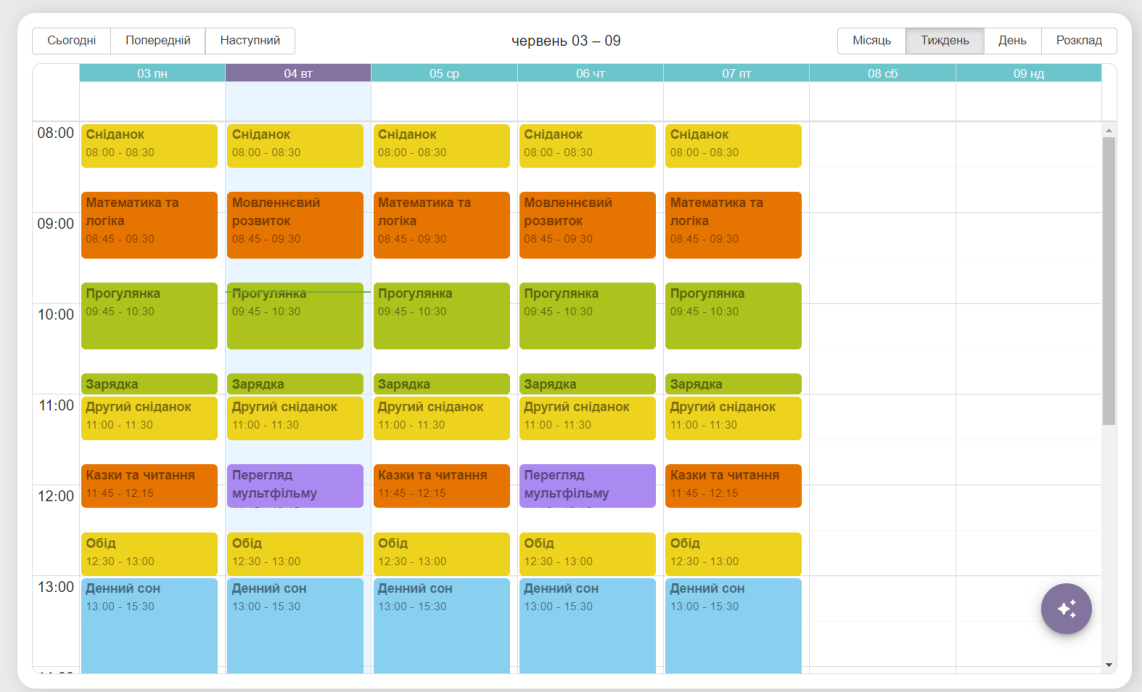


Рисунок 4.3 – Календар у «ChildWell» (рисунок виконаний самостійно)

Для нашого програмного продукту важливо, щоб журнал відвідування (див. рис. 4.4) допомагав вихователям безпомилково визначати присутність чи відсутність дитини в групі, а також час прибуття і час, коли дитина залишає заклад освіти. Отже, було використано текстові примітки, а також колір, які задовольняють описані вище потреби.

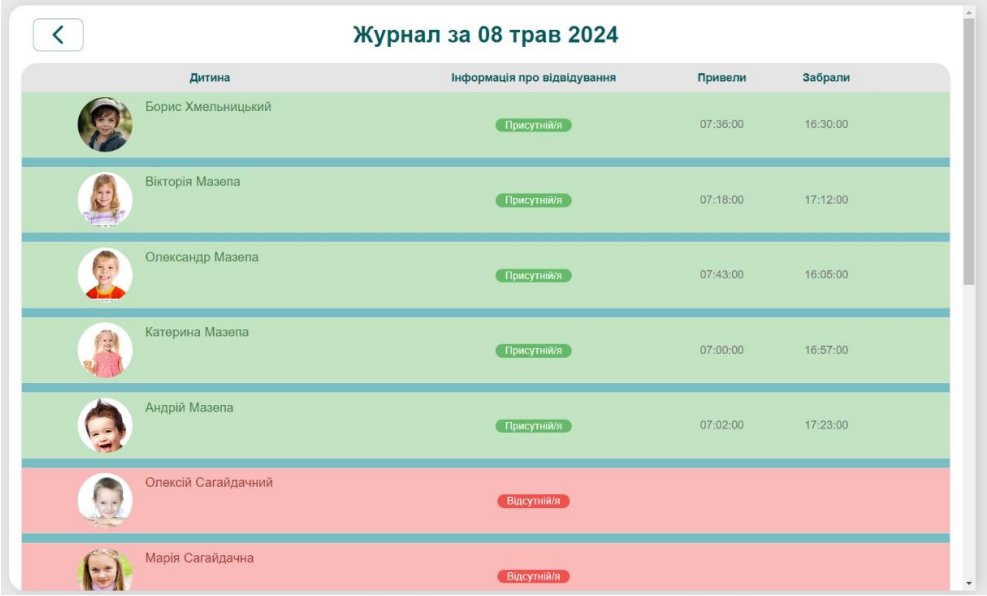


Рисунок 4.4 – Журнал відвідування в «ChildWell» (рисунок виконаний самостійно)

Беручи до уваги відомий закон Хіка: «Чим більше варіантів вибору надається, тим більше часу потрібно на вибір» [29], користувачу було надано можливість вибору (лише там, де це можливо та необхідно). Однак, їх не перевантажено цією можливістю. Наприклад групу можна обрати зі списку, а не записувати вручну, дату можна вибрати безпосередньо на календарі в рамках місяця або тижня. Зокрема часові рамки, доступні для вибору (наприклад при налаштуванні активностей ЗДО див. рис. 4.5) обмежені робочими годинами ЗДО.

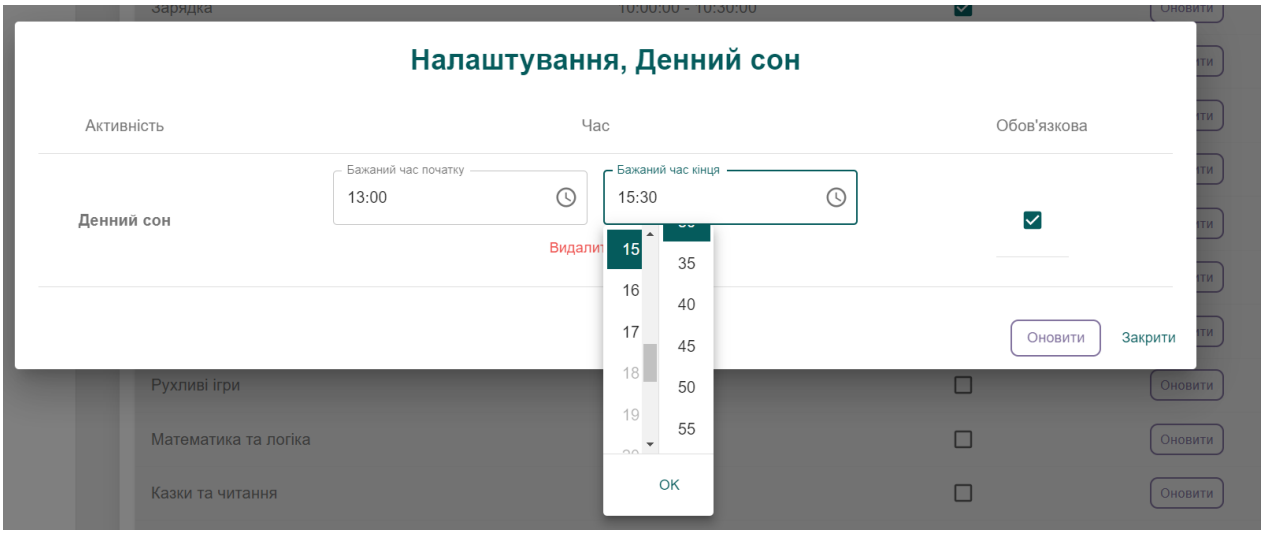


Рисунок 4.5 – Налаштування активностей ЗДО у «ChildWell» (рисунок виконаний самостійно)

Враховуючи наявність різних ролей у застосунку, кожну групу користувачів було забезпечено унікальним інтерфейсом, відповідним до їх запитів та потреб. Розглянемо це на прикладі головної сторінки для кожної з ролей.

Головна сторінка (або ж дашборд) дозволяє батькам користуватися найважливішими функціями одразу без додаткових переходів на інші сторінки (див. рис. 4.6). Запропоновані функції охоплюють як дітей, так і рахунки та заявки, наприклад:

– перегляд інформації про дітей;
– перегляд та оплата рахунків;
– керування заявками;
– створення профілю дитини.

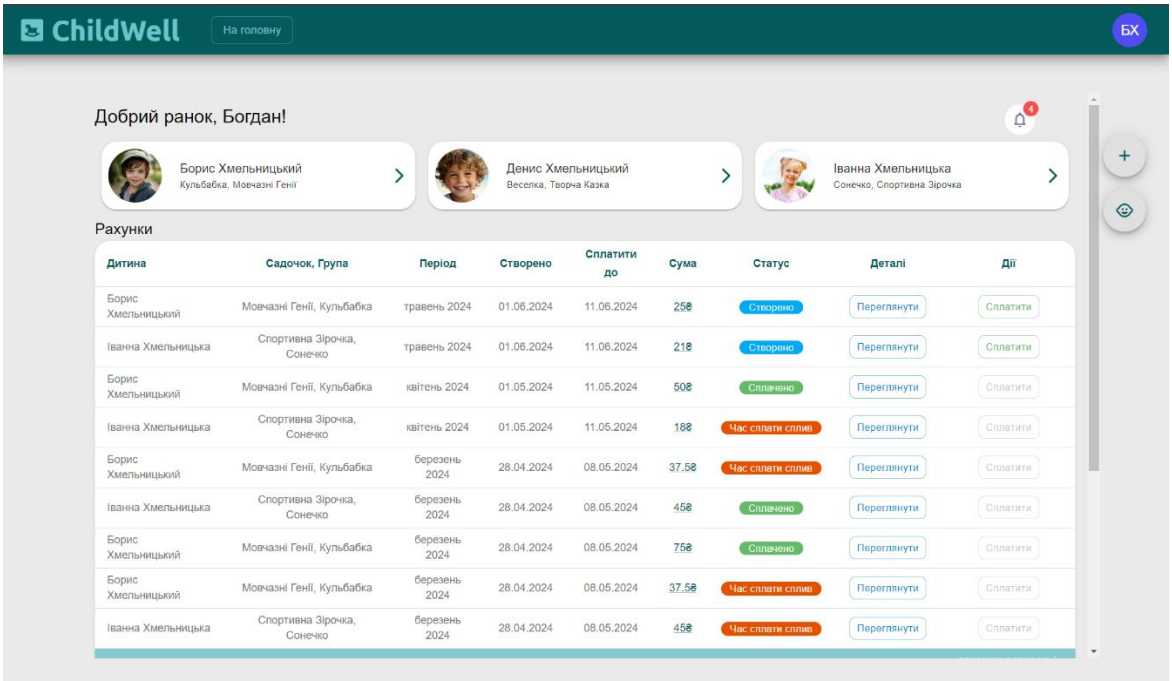


Рисунок 4.6 – Головна сторінка батьків у «ChildWell» (рисунок виконаний самостійно)

Ця ж сторінка для керівника ЗДО має інший вигляд (див. рис. 4.7). Вона надає функції, які стосуються закладів освіти, зокрема перегляд інформації про дитячі садки та керування ними.

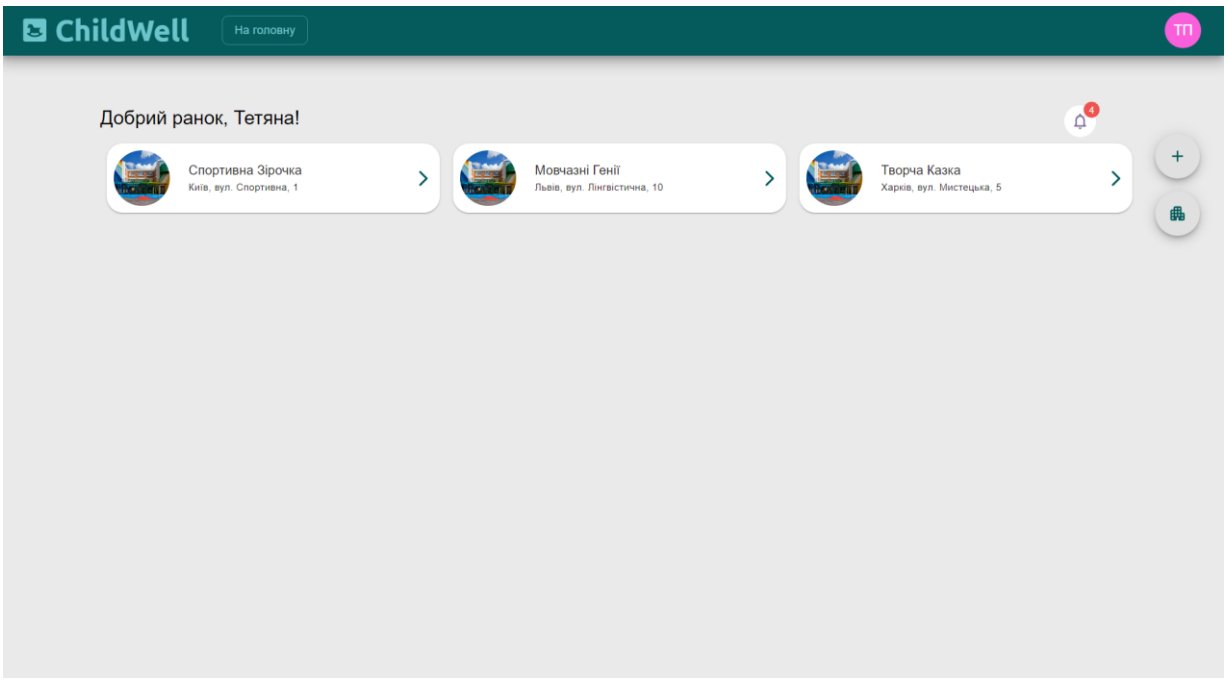


Рисунок 4.7 – Головна сторінка керівників ЗДО у «ChildWell» (рисунок виконаний самостійно)

Для вихователя на головній сторінці доступна інформація групи, в якій він працює (див. рис. 4.8).

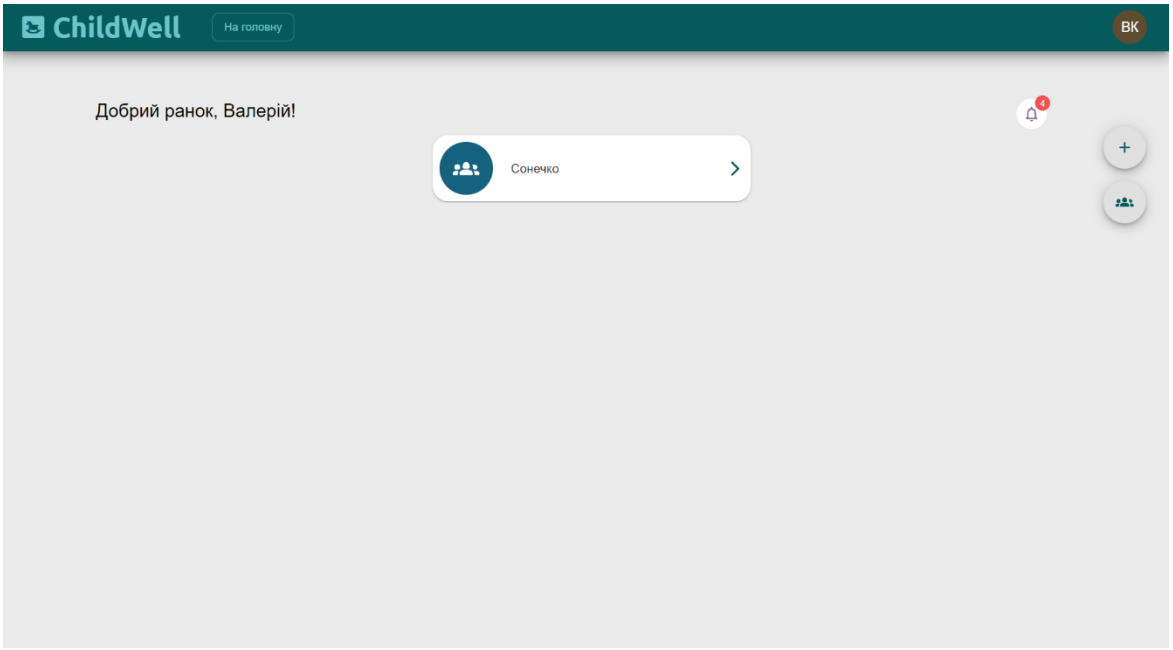


Рисунок 4.8 – Головна сторінка вихователя у «ChildWell» (рисунок виконаний самостійно)

Персоналізація інтерфейсу корегує наповнення сторінки користувача, візуалізує інформацію відповідно до його особистих запитів.

4.5 Безпека

Критичним аспектом розробки нашого програмного забезпечення є безпека. Тому було вжито низку заходів для забезпечення конфіденційності, цілісності й доступності даних. Процедури автентифікації й авторизації грають ключову роль у контексті обробки особистої інформації користувачів освітніх інформаційних систем.

Автентифікацією називається процес підтвердження ідентичності користувача, що намагається отримати доступ до системи. Авторизація є процедурою визначення відповідних прав та обмежень доступу користувачів до системи [30].

Для забезпечення авторизації у програмній системі для закладів дошкільної освіти з керуванням режимом дня та харчуванням дітей «ChildWell» використовується протокол Open Authorization (OAuth). Цей протокол є стандартним для авторизації та забезпечує доступ до додатку без необхідності передавати пароль. Тобто виділяється спеціальний токен, завдяки якому здійснюється доступ до ресурсів додатку без передачі конфіденційної інформації (пароля) третім сторонам [31].

Розширення OAuth OpenID Connect (OIDC) забезпечує автентифікацію (підтвердження, що користувач той, за кого себе видає) у нашому додатку. OIDC використовує ідентифікаційні маркери для передачі результатів автентифікації та будь-якої відповідної інформації перевіряючій стороні. Надсилатися можуть дані, що включають ідентифікатор, адресу електронної пошти та ім’я. [32].

Фрагмент коду, за допомогою якого здійснюється конфігурація для автентифікації за допомогою OIDC наведено нижче.

```
export const oidcConfig: AuthProviderProps = {
    onSigninCallback: () => {
        window.history.replaceState({}, document.title,
        window.location.pathname)
    },
```

```
    authority: `${appConfig.identityServerUrl}`,
    client_id: `${appConfig.reactAppClientId}`,
    redirect_uri: `${appConfig.reactPublicUrl}/signin-oidc`,
    silent_redirect_uri: `${appConfig.reactPublicUrl}/silent-renew`,
    post_logout_redirect_uri: `${appConfig.reactPublicUrl}`,
    response_type: `${appConfig.appResponseType}`,
    automaticSilentRenew: false,
    loadUserInfo: false,
    scope: `${appConfig.appClientScope}`,
    userStore: new WebStorageStateStore({ store:
     appConfig.useLocalStorageToPersistUser ?
     localStorage : sessionStorage })
}
```

У наведеному коді експортується об’єкт oidcConfig, що містить налаштування параметрів для OpenID Connect. Серед них адреса сервера авторизації, адреса перенаправлення в разі успішної авторизації, ідентифікатор додатку-клієнта, тип відповіді, очікуваної від сервера, адреса перенаправлення в разі виходу з системи.

5 ТЕСТУВАННЯ

Надійність, функціональність і безпеку програмного забезпечення, що є критичним для коректної роботи програми та позитивного враження від користування системою, перевіряється завдяки тестуванню.

У рамках тестування було випробувано доступ до функцій залежно від ролі користувача згідно з наведеними вимогами.

Під час тестування можливості доступу батьками до відповідного функціоналу було визначено, що батьки можуть переглядати розклад занять своєї дитини через інтерфейс додатку. У разі відсутності розкладу для групи виводилось повідомлення про це (див. рис. 5.1). Результати тестування показали, що батькам доступна інформація про відвідування їхньою дитиною ЗДО, включно з часовими межами знаходження дитини в ЗДО. Важливим аспектом перевірки було те, що батьки можуть переглядати рахунки за послуги дитячого садка та оплачувати їх через систему. Тестування також включало перевірку коректності проведення оплат та точність відображення історії платежів. Можливість батьків подавати заявки в дитячі садки, а також керувати статусом цих заявок також було протестовано успішно.

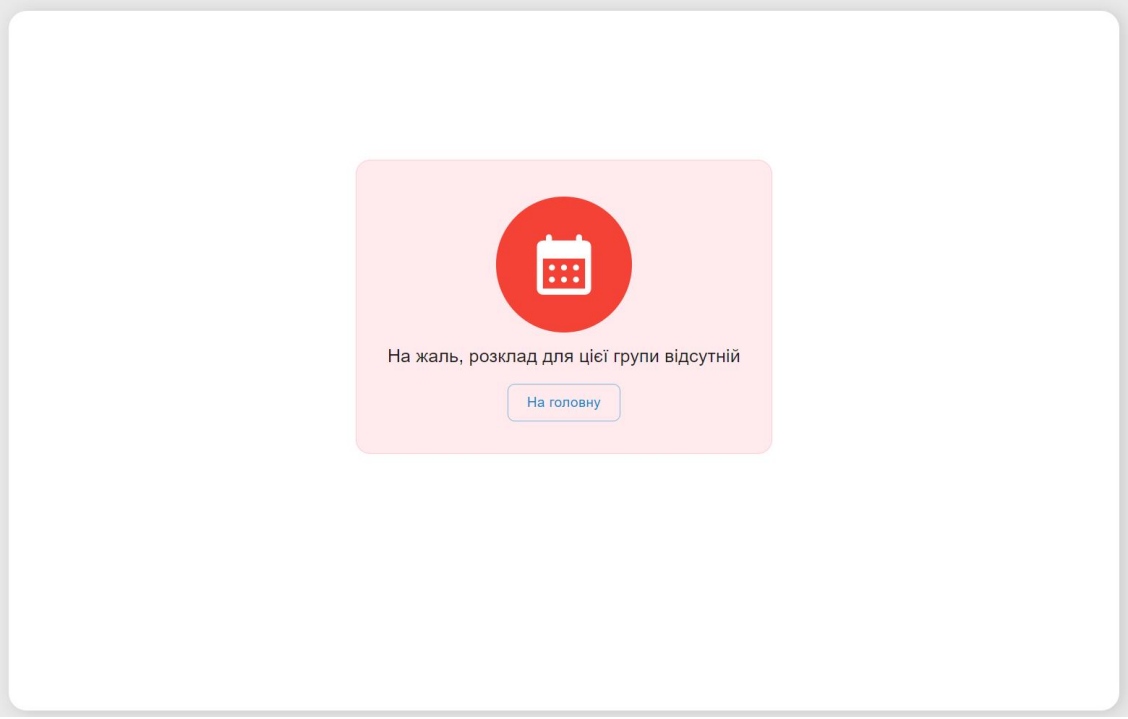


Рисунок 5.1 – Екран батьків при відсутності розкладу для групи (рисунок виконаний самостійно)

Для ролі вихователя перевірялося, чи є для нього доступним перегляд групи та основних даних. Тестування також включало перевірку точності та актуальності відображеної інформації. Після цього було випробувано можливості вихователів переглядати, генерувати (лише при відсутності див. рис. 5.2) й змінювати розклад занять для своєї групи. Перевірка доступу вихователів до журналу відвідування включала випробування можливості перегляду списку дітей, їх статусу відвідування за обраний період часу.

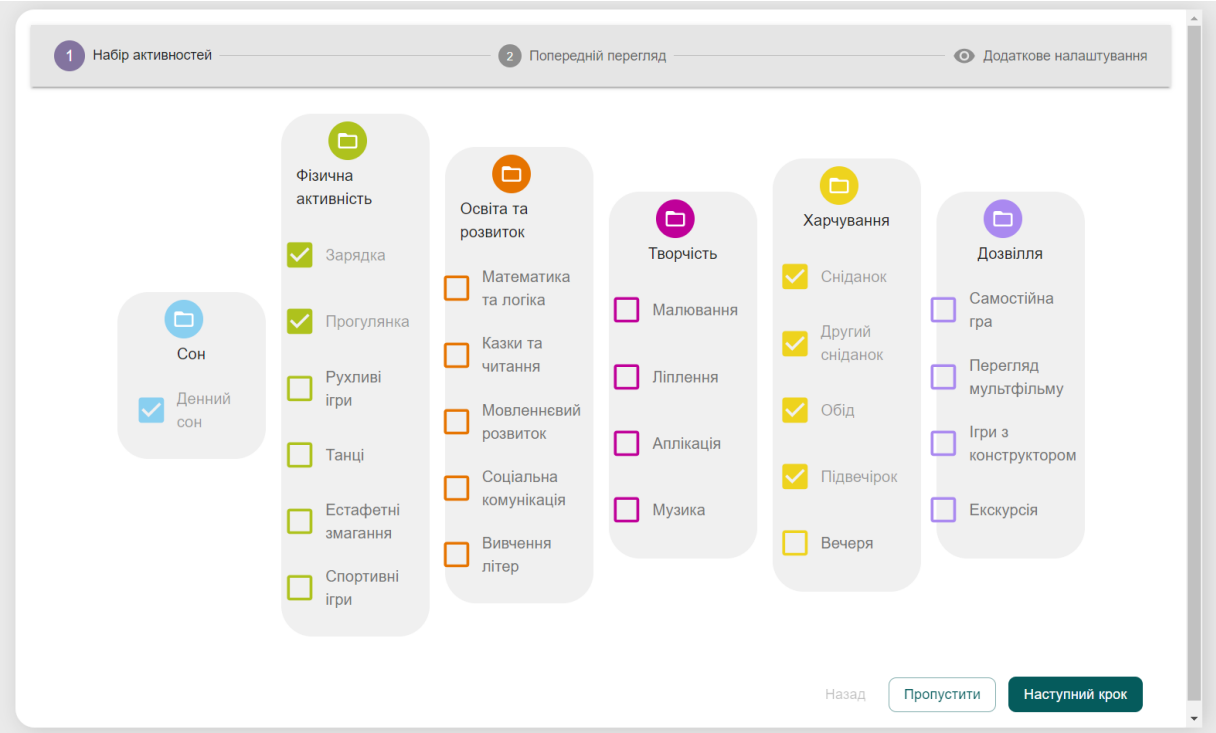


Рисунок 5.2 – Екран вихователів при відсутності розкладу для групи (рисунок виконаний самостійно)

Під час тестування інтрефейсу для керівників ЗДО акцент був зроблений на перевірці доступу до низки ключових можливостей. Наприклад тестування керування заявками на вступ охопило як перегляд, так і обробку заявок з урахуванням їхнього статусу. Також було вивчено можливості оновлення налаштувань активностей у закладі. Значну увагу було зосереджено на перевірці функції додавання нових дітей до груп, яка забезпечує ефективне управління складом груп.

Також було успішно проведене тестування переключення мови інтерфейсу між українською та англійською. У ході випробування було підтверджено, що користувач може легко переключати мову інтерфейсу за допомогою доступних інструментів. Здійснено перевірку правильності відображення тексту, як для української, так і для англійської мови, а також коректність відображення всіх елементів інтерфейсу після зміни мови.

Результати проведеного тестування підтвердили коректність та функціональність доступу кожної з ролей користувачів відповідно до вимог та очікувань щодо їхніх функціональних можливостей у системі.

Для оцінки складності виконання конкретних завдань користувачами було проведено квантифікаційний аналіз (KLM) за моделлю GOMS [34].

У розрахунках використано наступні константи:

- K= 0.2 сек – Keying (час для виконання одного натискання клавіші або клацання миші);
- P = 1.1 сек – Pointing (час для позиціонування курсору миші);
- H = 0.4 сек – Homing (час для користувача пересувати руки від клавіатури до миші);
- M = 1.35 сек – Mental (час для користувача підготуватися до наступного кроку);
- R = ? – Responding (час для комп'ютера на відповідь на виконані користувачем дії).

Фрагмент розрахунків для генерації розкладу вихователем одразу після входу в акаунт наведено нижче. Список необхідних жестів:

- Перемістити мишку на картку з групою – P;
- Натиснути на картку – K;
- Обрати активності для генерації – M;
- Перемістити мишку на кнопку «Наступний крок» – P;
- Натиснути на кнопку «Наступний крок» – K;
- Генерація розкладу та його завантаження – R;
- Перемістити мишку на поле для промпту – P;

– Натиснути на поле для промпту – K;

– Введення промпту – M;

– Перемістити мишку на кнопку «Наступний крок» – P;

– Натиснути на кнопку «Наступний крок» – K;

– Перемістити мишку на кнопку «Завершити» – P;

– Натиснути на кнопку «Завершити» – K;

Отже тепер можна розрахувати приблизний час для генерації розкладу вихователем:

$$T_{EXECUTE} = 1.1 + 0.2 + 1.35 + 1.1 + 0.2 + 1.1 + 0.2 + 1.35 + 1.1 + 0.2 + 1.1 + 0.2 = 9.2$$

Таким чином, загальний час виконання послідовності дій, які призведуть до генерації розкладу, дорівнює 9,2 секунди. Він не перевищує прийнятні межі, тож цей сценарій не потребує удосконалення.

Отже, за результатами тестування, інтерфейс відображає лише ті функції, доступ до яких має конкретна роль. Усі можливості, відповідні до рол, працюють безпомилково та задовольняють розроблені вимоги.

6 ВПРОВАДЖЕННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Використання програмного продукту «ChildWell» у діяльності широкого кола закладів дошкільної освіти України засвідчить успішне впровадження цього ПЗ. Це допоможе перевірити на практиці ефективність використання програмної системи для генерації розкладу, моніторингу відвідування дітьми груп, подачі заявок та інших аспектів функціонування дитячих садочків.

На даному етапі програмна система «ChildWell» має теоретичну апробацію (див. рис. 6.1) та результативні консультації з фахівцями. Матеріали нашого дослідження були представлені в рамках 28-го Міжнародного молодіжного форуму «РАДІОЕЛЕКТРОНІКА І МОЛОДЬ В XXI СТОЛІТТІ». Авторський колектив здобув ІІ місце на секції «Програмна Інженерія. Інформаційні технології в освіті» конференції «Інформаційні інтелектуальні системи» з доповіддю «Аналіз використання штучного інтелекту для генерації розкладу занять і харчування дітей в закладах дошкільної освіти».


ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ

ДИПЛОМ

НАГОРОДЖУЄТЬСЯ

авторський колектив у складі

Бондаренко А.А., Нестеренко В.В., Смєйко Б.М.

(групи ПЗПІ-20-1, ПЗПІ-20-3, Харківський національний університет радіоелектроніки),

який здобув **ІІ** місце

на секції

«Програмна Інженерія. Інформаційні технології в освіті»

конференції

«Інформаційні інтелектуальні системи»

в рамках **28-го** Міжнародного молодіжного форуму

«РАДІОЕЛЕКТРОНІКА І МОЛОДЬ В XXI СТОЛІТТІ»

з доповіддю

«Аналіз використання штучного інтелекту для генерації розкладу занять і харчування дітей в закладах дошкільної освіти»

Голова конференції «Інформаційні інтелектуальні системи», д.т.н., проф. Андрій ЄРОХІН

м. Харків, 2024

Рисунок 6.1 – Отриманий диплом (рисунок виконаний самостійно)

Також у процесі розробки «ChildWell» значну увагу було приділено взаємодії з працівниками закладів дошкільної освіти, зокрема керівником ЗДО. Результати взаємодії та детального обговорення практичного застосування додатку підтвердили релевантість та користь ПЗ саме в освітній системі України. Зазначимо, що використання front-end частини можливе лише за умови впровадження всіх частин додатку «ChildWell», а саме серверного застосунку, веб та мобільного клієнта.

За нашим переконанням, цілісна система зможе ефективно вирішувати поставлені задачі, серед яких реєстрація та авторизація, генерація розкладу для дітей, з урахуванням фізіологічних особливостей, зміна вихователями розкладів. До них також належать контроль батьків над процедурою передачі дітей довіреним особам, ведення електронного журналу відвідування групи з часовими рамками. Батьки також зможуть переглядати докладну інформацію про режим дня та активності дитини, сплачувати рахунки й подавати заявки до закладів освіти. Керівники ЗДО отримають можливість приймати заявки до закладу освіти та налаштовувати активності.

ВИСНОВКИ

Результатом роботи стала розробка програмного продукту «ChildWell». Процес створення програмного забезпечення складався з аналізу предметної області, виявлення в ній проблем та методів їх вирішення, специфікації вимог до ПЗ, а також проєктування архітектури, тестування й умов впровадження.

Було проаналізовано аналоги програмної системи «ChildWell», зокрема додатки «Preschool2me», «Procare», «KidCheck», «Teach 'n Go» та «LineLeader».

Для реалізації «ChildWell» було використано такий стек технологій:

- мову програмування C# на платформі .NET Framework 8.0, фреймворк ASP.NET;
- засоби мови TypeScript та бібліотеки React.js, за допомогою яких реалізована клієнтська частина;
- React Router, MobX, Material UI для створення працездатного та приємного в користуванні інтерфейсу.

Програмна система «ChildWell» задовольняє визначені функціональні та нефункціональні вимоги, зокрема відображає дані всіх об’єктів, підтримує керування об’єктами, здійснює генерацію розкладу та автоматичне обчислення даних. Вона також підтримує дві мови інтерфейсу та відповідає вимогам до локалізації й інтернаціоналізації, портативності, масштабованості, безпеки й ремонтопридатності.

До переваг створеної системи можна віднести наявність інструментів для генерації розкладу засобами ШІ з урахуванням вікових особливостей, можливість відстеження режиму дня та моніторингу відвідування. Корисними є також функція оплати рахунків, цифровізація процесу подачі заявок до дитячих садків. Для батьків найважливішою є можливість контролю процесу передачі дітей довіреним особам. Наявність україномовного інтерфейсу є особливою перевагою.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Освіта.ua. Державний стандарт дошкільної освіти: основні положення. [Електронний ресурс] – URL: https://osvita.ua/school/reform/80932/ (дата звернення: 23.04.2024).

2. Богданова Н. Г. Роль дошкільної освіти у вибудовуванні стратегії життєтворчості особистості. Гілея: науковий вісник. 2019. № 149 (10). С. 20–23. URL: http://surl.li/tqjwo (дата звернення: 25.04.2024).

3. Комплексне дослідження стану системи дошкільної освіти України / А. Горський та ін. Київ, 2013. 115 с. URL: https://www.irf.ua/files/ukr/programs/edu/edu_preschool.pdf (дата звернення: 25.04.2024).

4. Preschool2me. [Електронний ресурс] – URL: https://www.preschool2me.com/ (дата звернення: 27.04.2024).

5. Procare | The #1 Child Care Management Software. [Електронний ресурс] – URL: https://www.procaresoftware.com/ (дата звернення: 27.04.2024).

6. KidCheck Children's Check-In. [Електронний ресурс] – URL: https://www.kidcheck.com/ (дата звернення: 27.04.2024).

7. Teach 'n Go Ireland. [Електронний ресурс] – Modern School Management Software. URL: https://www.teachngo.com/ (дата звернення: 27.04.2024).

8. Childcare Software | LineLeader Unified Platform. [Електронний ресурс] – URL: https://lineleader.com/ (дата звернення: 27.04.2024).

9. Войчур Ю., Медзатий Д. Метод аналізу вимог до програмного забезпечення на предмет пошуку значень атрибутів якості. Міжнародний науково-технічний журнал «Вимірювальна та обчислювальна техніка в технологічних процесах». 2024. № 1. С. 146–151. URL: https://doi.org/10.31891/2219-9365-2024-77-18 (дата звернення: 29.04.2024).

10. ISO 25023:2016. Systems and software engineering – Systems and software Quality Requirements and Evaluation (SQuaRE) – Measurement of system and software product quality. Official edition. URL: https://www.iso.org/standard/35747.html (дата звернення: 29.04.2024).

11. Grytsiuk Y. I., Leshkevych I. F. The problems of definition and analysis of software requirements. Scientific Bulletin of UNFU. 2017. T. 27, № 4. C. 148–158. URL: https://doi.org/10.15421/40270433 (дата звернення: 30.04.2024).

12. Glib T., Finzi S. Principles of Software Engineering Management. 1988. URL: https://www.researchgate.net/publication/225070548_Principles_of_Software_Engineering_Management/references (дата звернення: 30.04.2024).

13. McCall J., Richards P., Walters G. Factors in software quality: concept and definitions of software quality. Rome : Rome Air Development Center, 1977. 168 c.

14. About the Unified Modeling Language Specification Version 2.5.1 [Електронний ресурс] – URL: https://www.omg.org/spec/UML/ (дата звернення: 30.04.2023).

15. Вакуленко Я. О., Мазурова О. О. Застосування методів аналізу текстів для підтримки концептуального моделювання баз даних. Системи обробки інформації. 2017. № 1(147). С. 127–130. URL: https://doi.org/10.30748/soi.2017.147.23 (дата звернення: 01.05.2024).

16. Семков Д. Розуміння Клієнт-Серверної Архітектури на прикладах. [Електронний ресурс] – URL: https://dou.ua/forums/topic/44636/ (дата звернення: 02.05.2024).

17. The Importance of UI/UX Design. [Електронний ресурс] – URL: https://www.igexsolutions.com/blog/the-importance-of-ui-ux-design/ (дата звернення: 04.05.2024).

18. Higher-Order Components. [Електронний ресурс] – URL: https://legacy.reactjs.org/docs/higher-order-components.html (дата звернення: 05.05.2024).

19. TypeScript: JavaScript With Syntax For Types. [Електронний ресурс] – URL: https://www.typescriptlang.org (дата звернення: 20.05.2024).

20. React. [Електронний ресурс] – URL: https://react.dev/ (дата звернення: 20.05.2024).

21. React Router. [Електронний ресурс] – URL: https://reactrouter.com/en/main (дата звернення: 20.05.2024).

22. MobX. [Електронний ресурс] – URL: https://mobx.js.org/README.html (дата звернення: 20.05.2024).

23. Material Design. [Електронний ресурс] – URL: https://m2.material.io/design/introduction#principles (дата звернення: 20.05.2024).

24. MUI: The React component library you always wanted. [Електронний ресурс] – URL: https://mui.com/ (дата звернення: 20.05.2024).

25. Scott E. SPA Design and Architecture: Understanding Single Page Web Applications. Manning Publications, 2015. 275 с.

26. Hunt A., Thomas D. Pragmatic Programmer: Your Journey to Mastery, 20th Anniversary Edition. Pearson Education, Limited, 2021. 352 с.

27. IETF HTTP Working Group Specifications. [Електронний ресурс] – URL: https://httpwg.org/specs/rfc9110.html (дата звернення: 22.05.2024).

28. Axios. [Електронний ресурс] – URL: https://axios-http.com/docs/intro (дата звернення: 22.05.2024).

29. Shapoval Agency. ТОП-7 законів у UX-дизайні. CASES. [Електронний ресурс] – URL: https://cases.media/en/article/top-7-zakoniv-u-ux-dizaini (дата звернення: 25.05.2024).

30. Комплекс навчально-методичного забезпечення навчальної дисципліни "Скриптові мови програмування" підготовки бакалавра спеціальність 121-Інженерія програмного забезпечення [Електронний ресурс] : освітня програма "Програмна інженерія" / ХНУРЕ ; розроб. І. П. Сокорчук. – Харків, 2017. – 274 с. (дата звернення: 25.05.2024).

31. OAuth Community Site. [Електронний ресурс] – URL: https://oauth.net/2/ (дата звернення: 25.05.2024).

32. What Is OpenID Connect (OIDC)? | Microsoft Security. [Електронний ресурс] – URL: https://www.microsoft.com/en-us/security/business/security-101/what-is-openid-connect-oidc (дата звернення: 25.05.2024).

33. Комплекс навчально-методичного забезпечення навчальної дисципліни "Людино-машинна взаємодія" підготовки бакалавра спеціальності 121 - Інженерія програмного забезпечення [Електронний ресурс] : освітня програма "Програмна

інженерія" / ХНУРЕ ; розроб. Р. В. Мельнікова. – Харків, 2017. – 181 с. (дата звернення: 31.05.2024).

34. John B. E., Kieras D. E. Using GOMS for user interface design and evaluation. ACM Transactions on Computer-Human Interaction. 1996. Т. 3, № 4. С. 287–319. [Електронний ресурс] – URL: https://doi.org/10.1145/235833.236050 (дата звернення: 01.06.2024).

Додаток А

Звіт результатів перевірки на унікальність тексту в базі ХНУРЕ

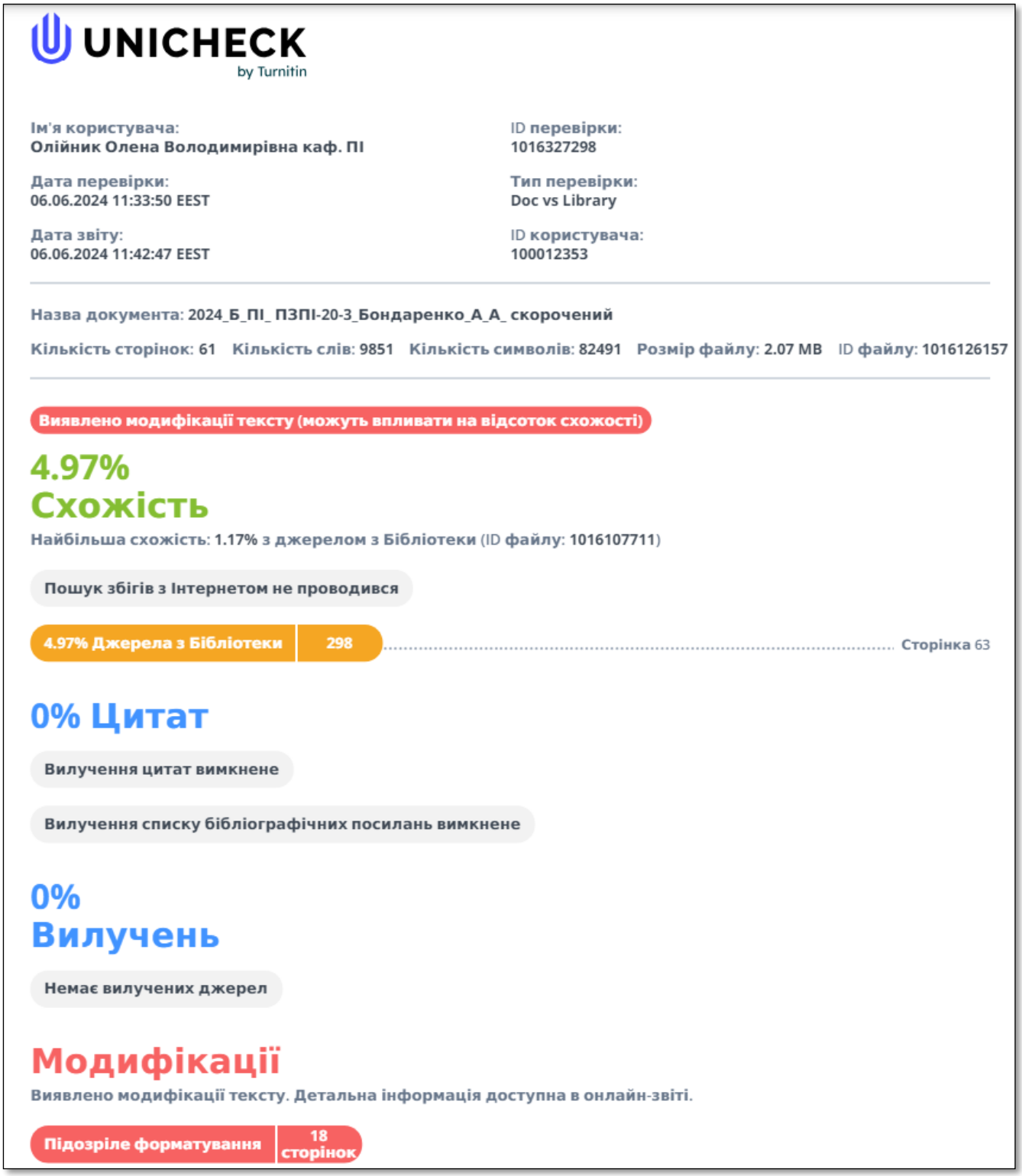


Рисунок А.1 – Результат перевірки на унікальність тексту в базі ХНУРЕ (рисунок виконаний самостійно)

Додаток Б

Слайди презентації

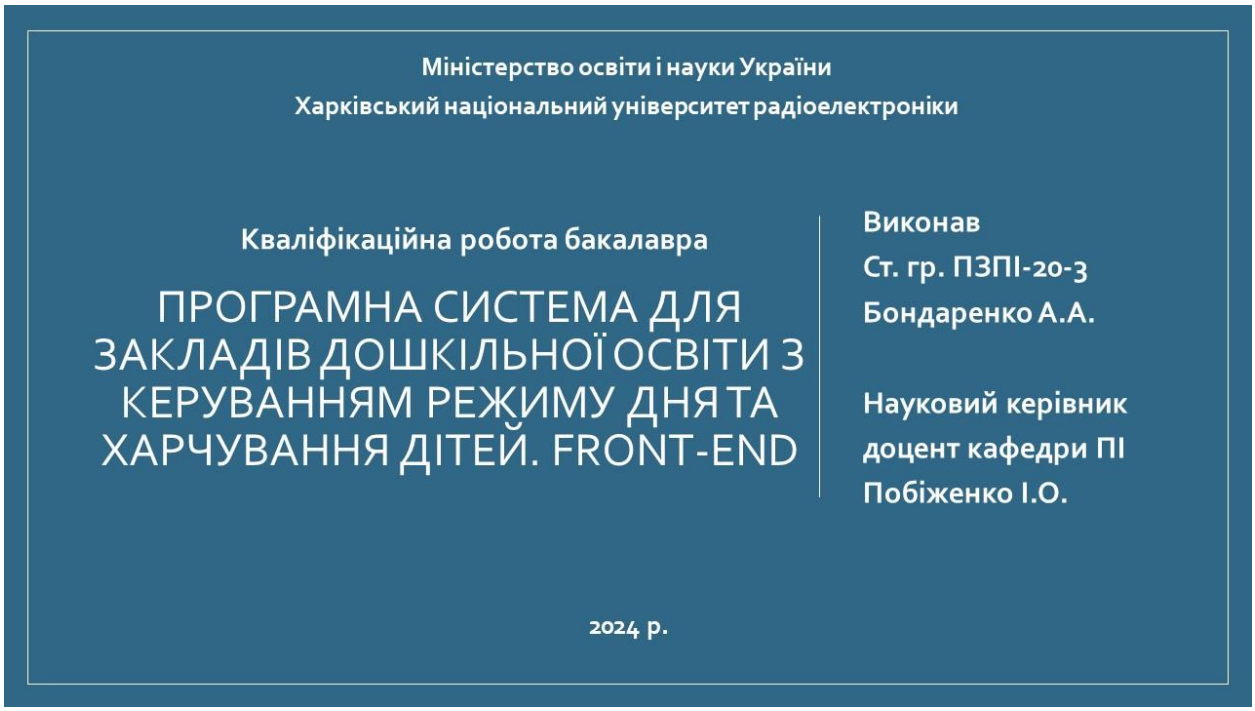


Рисунок Б.1 – Слайд 1 (рисунок виконаний самостійно)

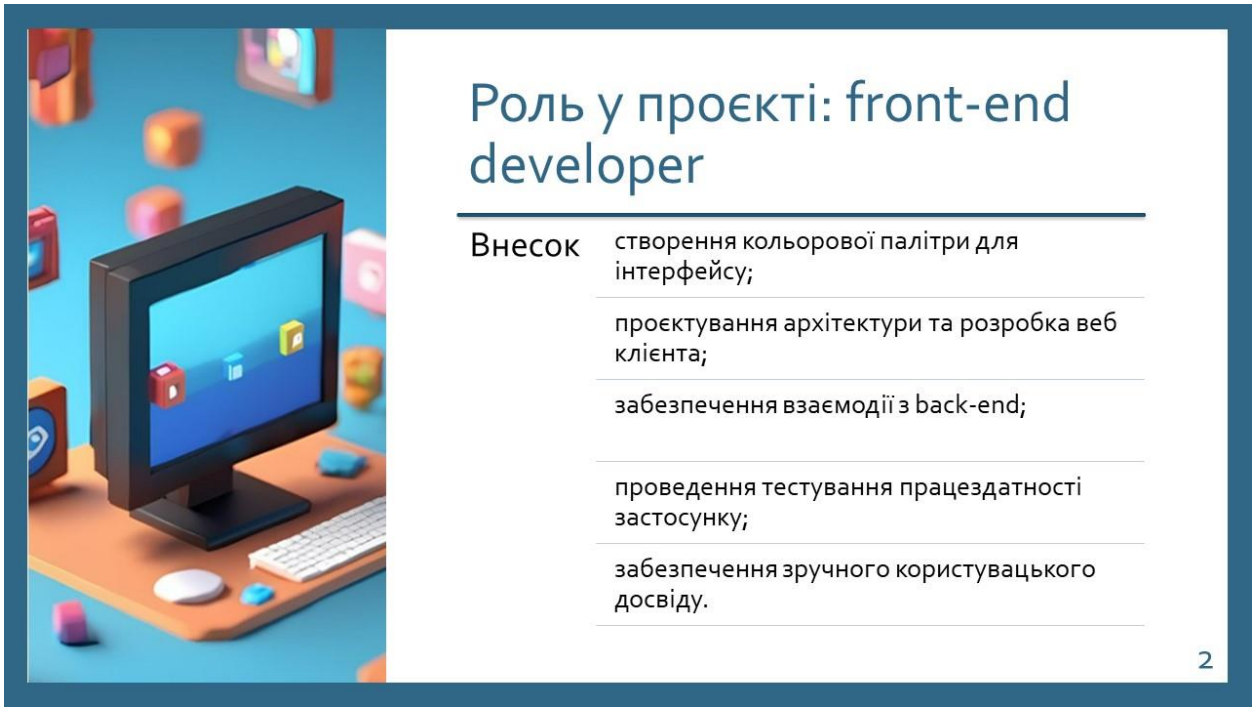


Рисунок Б.2 – Слайд 2 (рисунок виконаний самостійно)

Рисунок Б.3 – Слайд 3 (рисунок виконаний самостійно)

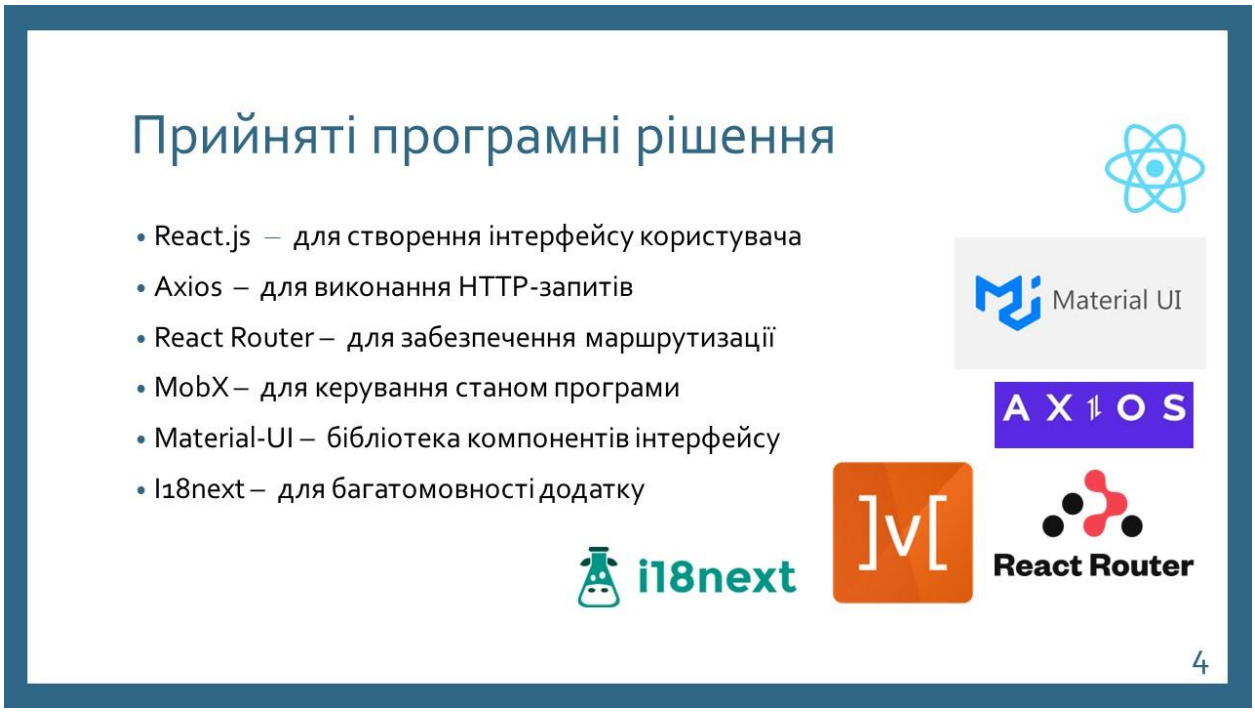


Рисунок Б.4 – Слайд 4 (рисунок виконаний самостійно)

Рисунок Б.5 – Слайд 5 (рисунок виконаний самостійно)

Рисунок Б.6 – Слайд 6 (рисунок виконаний самостійно)

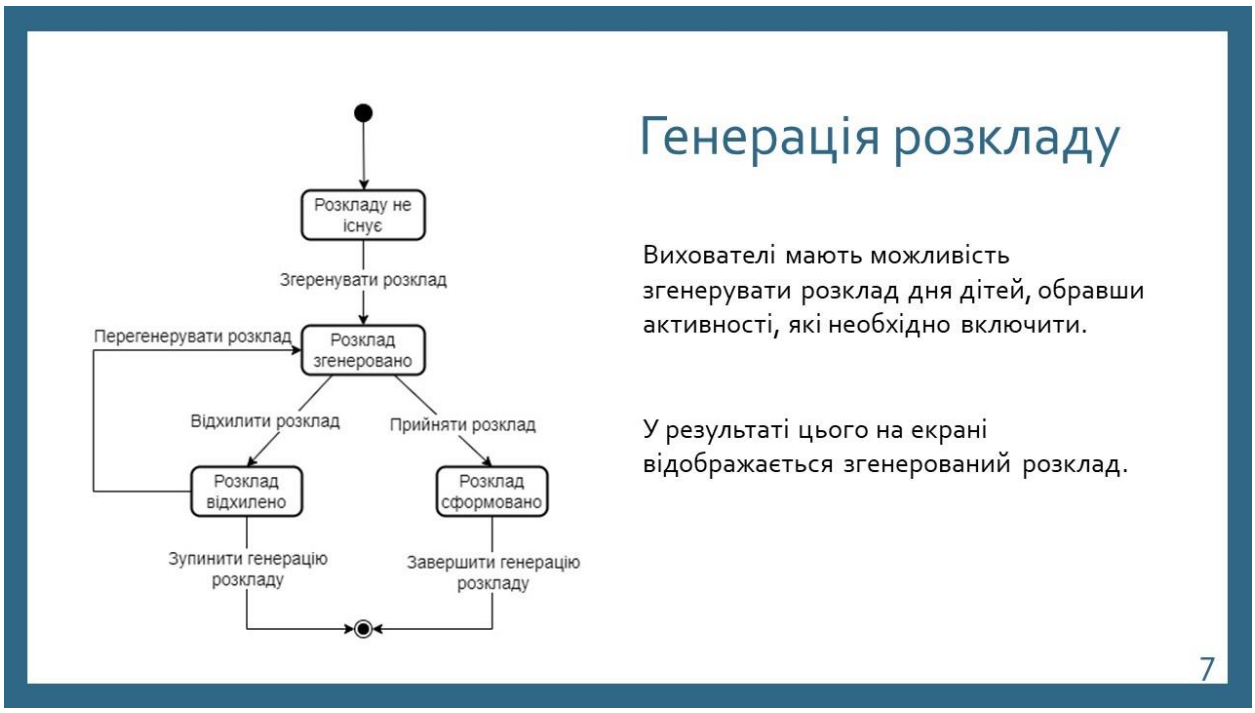


Рисунок Б.7 – Слайд 7 (рисунок виконаний самостійно)

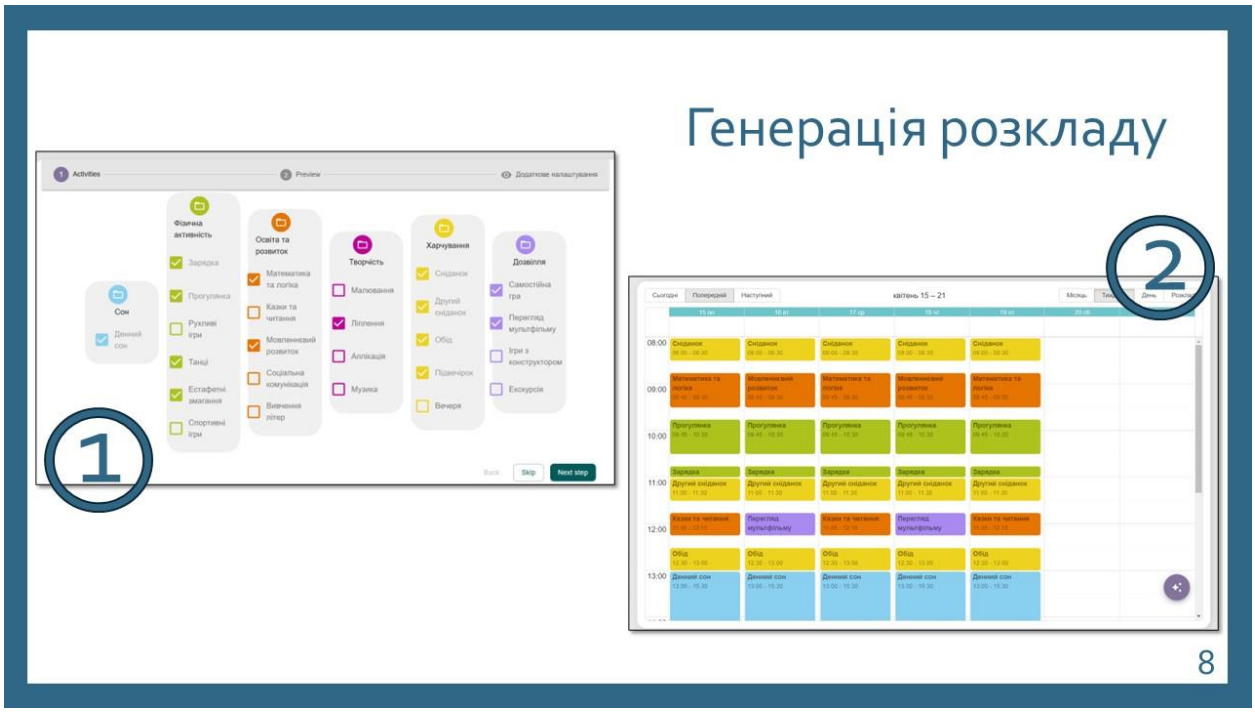


Рисунок Б.8 – Слайд 8 (рисунок виконаний самостійно)

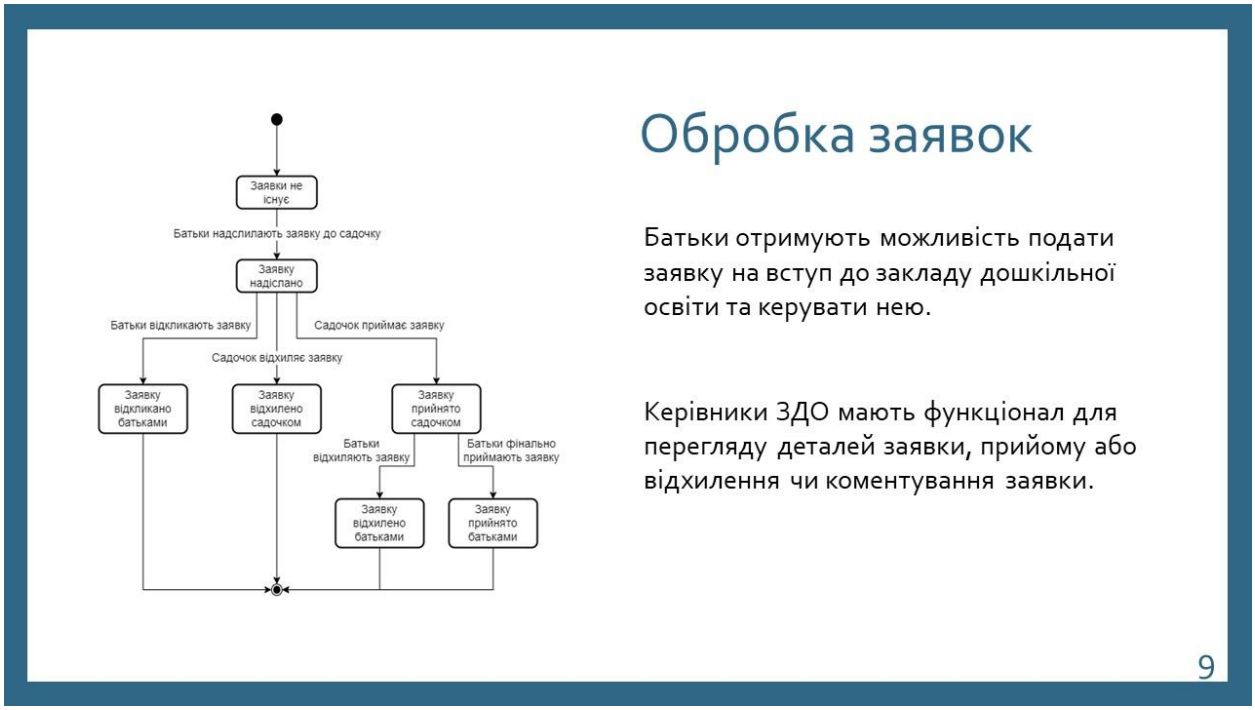
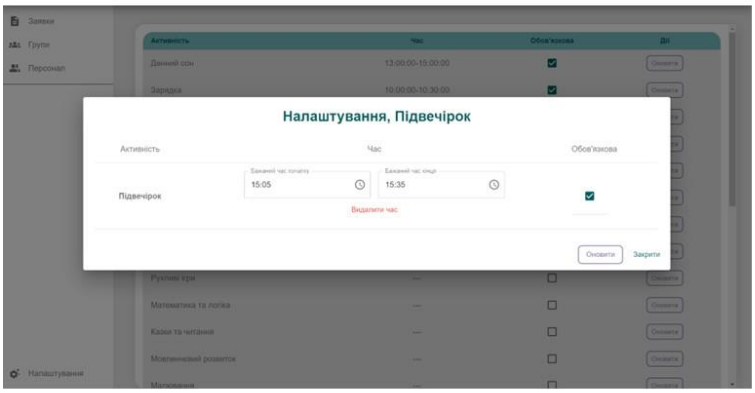


Рисунок Б.9 – Слайд 9 (рисунок виконаний самостійно)

Рисунок Б.10 – Слайд 10 (рисунок виконаний самостійно)

Оновлення налаштувань



Фунціонал доступний для керівника ЗДО для керування активностями садочку та бажаними часовими межами

11

Рисунок Б.11 – Слайд 11 (рисунок виконаний самостійно)

Висновки

- Розроблений додаток забезпечує взаємодію користувача з наданими функціями.
- Створена система генерації розкладу дітей дозволяє врахувати фізіологічні потреби різних вікових груп та прискорити рутинні процеси.
- Цифровізація процесу подачі заявок економить час і зусилля батьків дошкільнят, а для керівників ЗДО знижує навантаження та зменшення помилок.

12

Рисунок Б.12 – Слайд 12 (рисунок виконаний самостійно)

Додаток В

Специфікація програмного забезпечення

ChildWell

Software Requirements Specification

4.0

15.02.2024

Богдан Смєйко

Віталій Нестеренко

Анна Бондаренко

Історія версій

Дата	Опис	Автор	Коментарі
25.01.2024	Версія 1.0		Перша редакція
05.02.2024	Версія 2.0	Анна Бондаренко	Перелік функцій продукту розміщено в порядку пріоритетності Проєкту надано назву
10.02.2024	Версія 3.0	Богдан Смєйко	Оновлено класи
15.02.2024	Версія 4.0	Віталій Нестеренко	Оновлено діаграми

Затвердження документів

Наступну Специфікацію вимог до програмного забезпечення було прийнято та схвалено:

Підпис	Ім’я	Роль	Дата
	Богдан Смєйко	Back-end частина	15.02.2024
	Віталій Нестеренко	Mobile частина	15.02.2024
	Анна Бондаренко	Front-end частина	15.02.2024

Зміст

1. Вступ

У сучасному світі розвиток дитини визначається не лише академічними досягненнями, проте й фізичним та ментальним здоров’ям, емоційним станом, обсягом уваги. Тож виникає виклик у створенні та впровадженні ефективного розпорядку дня в закладах дошкільної освіти, де закладається фундамент для розвитку подальших здібностей дитини та формування їхніх звичок, що безумовно має вплив на все подальше життя вихованців. Важко сумістити бажання вихователів забезпечити оптимальний розвиток дітей, уникнути виснаження та гарантування засвоєння необхідних навичок і знань та потребу батьків у своєчасному доступі до інформацію про активності дітей в дитячих садках.

Виходячи з цього, постає актуальна потреба в розробці системи, яка не лише допоможе в розвитку навичок, а дозволить планувати розпорядок дня так, щоб діти не виснажувалися й забезпечить батькам надійний та миттєвий доступ до інформації щодо режиму дня та харчування їхніх дітей. Наша програма має на меті оптимізацію описаних вище процесів, що відповідає як вимогам педагогів, так і батьків. Детальніше про проєкт розповідається в даному документі.

1.1 Мета

Мета документу — надати детальний опис розроблюваної програмної системи для закладів дошкільної освіти з керуванням режимом дня та харчуванням дітей. Специфікація призначена для опису функціональних та нефункціональних вимог першої версії програмного продукту. SRS використовуватиметься членами команди, які будуть реалізовувати та забезпечувати коректну роботу системи та іншими стейкхолдерами.

1.2 Межі

Система, яку ми розробляємо отримала назву «ChildWell». Вона надаватиме можливість для викладачів створювати ефективний розклад для вихованців з

урахуванням їх фізіологічних особливостей, який гарантує здобуття потрібних навичок. Для батьків — можливість отримувати інформацію про активності дитини в закладі дошкільної освіти та її харчування в режимі реального часу.

Корисним функціоналом для вихователів стане планувальних розпорядку дня дітей, щоб вони не виснажувались та здобували потрібні навички. Батьки ж в свою чергу зможуть швидко отримувати достовірну інформацію щодо режиму дня дітей та їхнього харчування. Більш того корисним функціоналом стане можливість відмічання присутності дітей за допомогою QR-кодів, що буде доступним як для батьків так і вихователів. Крім цього батьки зможуть управляти заявками на подання дітей до закладів дошкільної освіти, а після матимуть змогу зручно сплачувати рахунки за послуги закладу.

Таким чином, система сприятиме ефективній організації розвитку дітей, забезпечуючи комфорт та контроль як для вихователів й адміністраторів закладу, так і для батьків.

1.3 Означення та абревіатури

Front-end — частина програмного забезпечення, яка відповідає за взаємодію користувача з системою або веб-додатком (дизайн, інтерфейс користувача та усі елементи, які користувач може бачити та з якими він може взаємодіяти).

int — дані цілочисельного типу.

text — дані текстового типу.

bool — це тип даних, який має одне з двох можливих значень (істине і хибне), які призначені для представлення двох істинних значень логіки та булевої алгебри.

date — тип даних для зберігання інформації про конкретний день у форматі рік-місяць-день.

datetime — тип даних, що включає в себе інформацію про конкретний момент у часі, включаючи як дату, так і час.

Git — це розподілена система контролю версій, яка відстежує зміни в будь-якому наборі комп'ютерних файлів.

ISO/IEC 27001 — це міжнародний стандарт для управління інформаційною безпекою.

WCAG (Інструкції щодо доступності веб-вмісту) — частина серії рекомендацій щодо доступності веб-сайтів, опублікованих Ініціативою веб-доступності (WAI) Консорціуму Всесвітньої павутини (W3C), головною міжнародною організацією стандартів для Інтернету. Вони являють собою набір рекомендацій щодо того, як зробити веб-вміст більш доступним.

GDPR (Загальний регламент захисту даних) — це нормативний акт Європейського Союзу щодо конфіденційності інформації в Європейському Союзі та Європейській економічній зоні.

1.4 Посилання

- ГОСТ 34.602-89: стандарти для автоматизованих систем, які регулюють вимоги до проектування та розробки.
- Методичні рекомендації МОН України: вказівки Міністерства освіти і науки України щодо організації дошкільної освіти дітей.
- ISO/IEC 27001: стандарт, що встановлює вимоги до систем управління інформаційною безпекою, який допоможе забезпечити безпеку даних користувачів.
- WCAG (Web Content Accessibility Guidelines): рекомендації щодо доступності веб-контенту, що допоможуть зробити додаток доступним для всіх користувачів, включаючи людей з обмеженими можливостями.
- GDPR (General Data Protection Regulation): регламент ЄС щодо захисту персональних даних.

1.5 Огляд

Цей документ містить повну інформацію про проєкт: його перспективи, функції, інтерфейс, обмеження, відомості про варіанти використання та діаграми для візуалізації різних аспектів системи. Кожна секція призначена для різних стейкхолдерів, від чого залежить зміст секцій.

Розділ «Загальний опис» націлений на опис загальних факторів, що впливають на вимоги до продукту і продукт в цілому. Цільовою аудиторією цього розділу перш за все є власник проєкту.

Розділ «Конкретні вимоги», на відміну від попереднього, містить подробиці щодо вимог до проєкту, його інтерфейси та функціональність, у ньому деталі реалізації описані технічною мовою для сприйняття розробниками.

У розділі «Моделі аналізу» розміщено діаграми та моделі, що використовуватимуться в процесі розробки відповідно до вимог, визначених в цьому документі SRS. Вони допоможуть у розумінні системи, шляхом візуалізації її роботи та взаємодії, що відбувається всередині системи.

Останній розділ, «Процес управління змінами», передбачує та описує яким чином відбуватиметься оновлення SRS, якщо виникне така необхідність при внесенні змін до вимог чи обсягів проєкту. Також він має вказувати на те, ким вноситимуться та ухвалюватимуться зміни.

2. Загальний опис

2.1 Перспективи продукту

Система планується як самостійний продукт, що не залежить від схожих застосунків. Для повноцінного використання користувачі повинні зареєструватися та використовувати браузерну або мобільну версію додатку.

2.2 Функції продукту

Основною метою продукту є оптимізація планування розпорядку дня дітей у закладах дошкільної освіти з управлінням харчуванням. Викладачам треба мати змогу керувати режимом дня дітей та отримувати інформацію про те, чи є він відповідним до вікових особливостей вихованців. Батьки ж у свою чергу отримають інструмент для відстеження активностей дитини та комунікації з працівниками закладу дошкільної освіти. До основних функцій продукту належать наступні:

- MF-1: Реєстрація та авторизація користувачів;
- MF-2: Створення оптимального розкладу для дітей, з урахуванням фізіологічних особливостей;
- MF-3: Зміна вихователями розкладу;
- MF-4: Контроль батьків над процедурою передачі дітей в довірені руки;
- MF-5: Автоматичне сповіщення вихователів про відсутність дитини;
- MF-6: Електронний журнал відвідуваності з часовими рамками;
- MF-7: Перегляд докладної інформації про режим дня та активності дитини;
- MF-8: Оплата рахунків (батьками);
- MF-9: Подача заявок до закладу освіти;
- MF-10: Керування заявками до закладу освіти;
- MF-11: Налаштування завідувачем (адміністратором) певних аспектів закладу освіти.

  Також система повинна надавати інформацію про розробників та контакти.

2.3 Характеристики користувачів

Передбачено 3 види користувачів, що матимуть доступ до системи:

- батьки дітей дошкільного віку (подання та управління заявками до дитячих садків, перегляд режиму дня, сплата рахунків);

- вихователі закладів дошкільної освіти (створення режиму дня, відмічання присутності дітей);
- адміністративний персонал закладу (управління заявками від батьків, управління та налаштування дитячого садка, створення занять, реєстрація вихователів).

2.4 Загальні обмеження

Система повинна працювати на пристроях з наступними характеристиками:

- мінімум 4 ГБ оперативної пам'яті;
- процесор не нижче як чотирьохядерний, з тактовою частотою не менше 2.0 ГГц.

Система має забезпечувати сумісність з операційними системами:

- Android 6.0 або новіше;
- iOS 13.4 або новіше.

Сучасні браузери підтримують виконання JavaScript, тому front-end частина буде виконана з використанням фреймворку React.js. Система повинна бути готовою до впровадження не пізніше 8 травня 2024 року. Система повинна відповідати стандартам безпеки даних, включаючи шифрування особистої інформації та заходи забезпечення конфіденційності.

2.5 Припущення й залежності

- Припущення: Користувачі мають доступ до Інтернету і він достатньо якісний;
    - Залежність: За відсутності доступу до Інтернету, користувачі не можуть використовувати застосунок у повній мірі;

- Припущення: Користувачі зареєстровані в системі;
    - Залежність: Незареєстровані користувачі не можуть користуватися системою в повному обсязі;
- Припущення: Всі користувачі мають апаратні та програмні засоби, необхідні для коректної роботи системи, зокрема, підтримують необхідні версії браузерів та операційних систем;
    - Залежність: Використання браузерів або пристроїв, що не підтримуються системою, може призвести до нестабільної роботи або відмови в доступі до певних функцій;
- Припущення: Всі користувачі володіють основами мови, в якій реалізовано інтерфейси користувача;
    - Залежність: Коректне користування системою передбачає знання користувачами мові, в якій вона реалізована.

3. Конкретні вимоги

3.1 Вимоги до зовнішніх інтерфейсів

3.1.1 Інтерфейс користувача

Зареєстровані користувачі поділятимуться на вихователів та батьків й матимуть трохи різні інтерфейси.

Ці обидва типи користувачів матимуть доступ до режимів дня та планів харчування дітей, але лише вихователі зможуть їх редагувати. Також батьки матимуть окрему сторінку з дошкою оголошень, на якій бачитимуть повідомлення від дитсадків та вихователів.

3.1.2 Апаратний інтерфейс

Для відображення інтерфейсу користувача до пристрою має бути підключений дисплей.

3.1.3 Програмний інтерфейс

Для доступу до веб-системи користувач повинен використовувати веб-браузер, який повинен підтримувати Javascript, HTML5, CSS3. Для використання мобільного додатку, користувачеві необхідно мати пристрій з ОС Android 6.0+ або iOS 13.4+.

3.1.4 Комунікаційні інтерфейси

Щоб користувач мав доступ до веб-системи, необхідна наявність Інтернету.

3.2 Функціональні вимоги

3.2.1 Реєстрація та авторизація користувачів

3.2.1.1 Опис

Нові користувачі можуть створювати акаунти в системі реєструватися за допомогою Google акаунта та використовувати їх для входу.

3.2.1.2 Вхідні умови

Заповнення усіх обов’язкових полів для реєстрації або входу в систему.

3.2.1.3 Виконання

Сервер створює нового користувача при реєстрації. При вході в систему сервер перевіряє чи існує такий користувач, якщо ні, то виводить відповідне повідомлення.

3.2.1.4 Вихідні умови

Створений новий користувач.

3.2.1.5 Обробка помилок

Якщо введено неправильний пароль або логін, буде показане відповідне повідомлення.

3.2.2 Генерація розкладу для дітей, з урахуванням фізіологічних особливостей

3.2.2.1 Опис

Система в автоматичному режимі може створити розклад для дітей, враховуючи розподіл часу між іграми, навчанням, прогулянками та прийомами їжі.

3.2.2.2 Вхідні умови

Задано початкові значення та пріоритети (обрано активності з наданих категорій).

3.2.2.3 Виконання

Сервер виконує обробку, розташовуючи активності в потрібному та правильному порядку.

3.2.2.4 Вихідні умови

Створений розклад дня для дітей.

3.2.2.5 Обробка помилок

В разі помилки система запропонує змінити пріоритети в розкладі, так як з наявними неможливо створити оптимальний розклад.

3.2.3 Зміна викладачами режимів дня та активностей

3.2.3.1 Опис

Викладачі можуть корегувати розклад дня дітей, що був згенерований системою.

3.2.3.2 Вхідні умови

Користувач повинен мати роль викладача.

3.2.3.3 Виконання

Сервер оновлює існуючий розклад.

3.2.3.4 Вихідні умови

Змінений розклад дня.

3.2.3.5 Обробка помилок

Якщо користувач не викладач – заборонити зміну розкладу.

3.2.4 Перегляд журналу відвідування групи

3.2.4.1 Опис

Вихователі повинні мати можливість переглядати журнал відвідуваності групи дітьми за обраний день.

3.2.4.2 Вхідні умови

Обрано групу та день, за який присутня інформація про відвідуваність групи дітьми.

3.2.4.3 Виконання

Сервер обробляє запит на надає відповідні дані у відповідь.

3.2.4.4 Вихідні умови

Журнал відвідуваності групи за обраний день успішно відображено на екрані вихователя.

3.2.4.5 Обробка помилок

В разі помилки система виводить повідомлення про відсутність запитуваних даних.

3.2.5 Контроль батьків над процедурою передачі дітей в довірені руки

3.2.5.1 Опис

Можливість батьків контролювати процедури передачі своєї дитини в довірені руки.

3.2.5.2 Вхідні умови

Батьки обирають дитину, для якої треба згенерувати check-in/check-out QR-код.

3.2.5.3 Виконання

Сервер генерує QR-код, що містить всю необхідну інформацію для передачі дитини в довірені руки.

3.2.5.4 Вихідні умови

Надано дані, які система буде використовувати при здійсненні цієї процедури.

3.2.5.5 Обробка помилок

Система надає повідомлення про будь-які помилки.

3.2.6 Автоматичне сповіщення вихователів про відсутність дитини

3.2.6.1 Опис

Система автоматично повідомляє вихователів про відсутність дитини, відображаючи відповідну інформацію в журналі присутності.

3.2.6.2 Вхідні умови

Система фіксує інформацію про відсутність дитини, якщо її присутність не зареєстровано за допомогою QR-код.

3.2.6.3 Виконання

Сервер автоматично збирає необхідну інформацію та надає її вихователям.

3.2.6.4 Вихідні умови

Вихователі отримують інформацію про відсутність дитини в закладі дошкільної освіти.

3.2.6.5 Обробка помилок

Система виявляє можливі конфлікти чи невідповідності в інформації і повідомляє вихователів для вирішення ситуації.

3.2.7 Додавання дитини в групу

3.2.7.1 Опис

Керівники ЗДО матимуть можливість додати дитину в групу, обравши одну з-поміж переліку груп садочка, відповідних до віку дитини.

3.2.7.2 Вхідні умови

Для додавання дитини в групу користувач повинен мати роль керівника ЗДО. Дитина як і група повинні бути обрані для коректного здійснення операції.

3.2.7.3 Виконання

Сервер додає нову дитину до групи та зберігає зміни.

3.2.7.4 Вихідні умови

Батьки та керівник бачать оновлену інформацію .

3.2.7.5 Обробка помилок

При відсутності необхідної ролі – повідомити користувача про це.

3.2.8 Перегляд інформації про режим дня та активності дитини

3.2.8.1 Опис

Батьки та вихователі зможуть переглядати детальну інформацію про режим дня та активності дітей.

3.2.8.2 Вхідні умови

На відповідній сторінці календар дозволить переглядати розклад і одному з режимів (місяць, тиждень, день, табличний вигляд).

3.2.8.3 Виконання

Сервер збирає та надсилає інформацію про дитину.

3.2.8.4 Вихідні умови

Батьки або вихователі бачать інформацію про режим дня та активності дитини.

3.2.8.5 Обробка помилок

При помилці сервер спробує зібрати інформацію кілька разів, після чого відобразить повідомлення про неуспішність операції.

3.2.9 Оплата рахунків для батьків

3.2.9.1 Опис

Функція дозволяє батькам здійснювати оплату рахунків за утримання та надані послуги в дошкільному закладі.

3.2.9.2 Вхідні умови

Батькам доступно отримання деталізації рахунків.

3.2.9.3 Виконання

Система формує рахунки для батьків, відображаючи вартість та послуги, надані дитині.

3.2.9.4 Вихідні умови

Батьки мають можливість вчасно та зручно оплачувати рахунки.

3.2.9.5 Обробка помилок

Система надає повідомлення про будь-які помилки та рекомендації для виправлення ситуації.

3.2.10 Подача заявки до закладу освіти

3.2.10.1 Опис

Батьки матимуть можливість подати заявку на вступ до закладу дошкільної освіти.

3.2.10.2 Вхідні умови

Обрано дитину, на ім’я якої й буде створено заявку, набір у освітній заклад відкрито.

3.2.10.3 Виконання

Сервер обробляє подану заявку, зберігає її та надає інформацію про неї керівнику ЗДО.

3.2.10.4 Вихідні умови

Можливість переглянути заявки на відповідній сторінці.

3.2.10.5 Обробка помилок

При помилці сервер спробує відправити заявку кілька разів, після чого відобразить повідомлення про неуспішність відправлення.

3.2.11 Керування заявкою до закладу освіти

3.2.11.1 Опис

Батьки матимуть можливість скасувати, прийняти або відхилити заявку. Керівники ЗДО матимуть можливість для перегляду деталей заявки, прийому або відхилення чи коментування заявки.

3.2.11.2 Вхідні умови

Обрано заявку, керування якою треба здійснити.

3.2.11.3 Виконання

Сервер обробляє оновлену заявку, зберігає її.

3.2.11.4 Вихідні умови

Можливість переглянути заявки на відповідній сторінці та оновити за допомогою відповідних кнопок.

3.2.11.5 Обробка помилок

При помилці сервер спробує оновити заявку кілька разів, після чого відобразить повідомлення про неуспішність відправлення.

3.2.12 Оновлення налаштувань активностей закладу освіти

3.2.12.1 Опис

Керівники ЗДО можливість оновити налаштування активностей ЗДО.

3.2.12.2 Вхідні умови

Обрано ЗДО та активності.

3.2.12.3 Виконання

Сервер обробляє оновлені налаштування, зберігає їх.

3.2.12.4 Вихідні умови

Можливість переглянути налаштування на відповідній сторінці та оновити за допомогою відповідних кнопок.

3.2.12.5 Обробка помилок

При помилці сервер спробує оновити налаштування кілька разів, після чого відобразить повідомлення про неуспішність відправлення.

3.3 Варіанти використання (Use Cases)

3.3.1 Неавтентифікований варіант використання

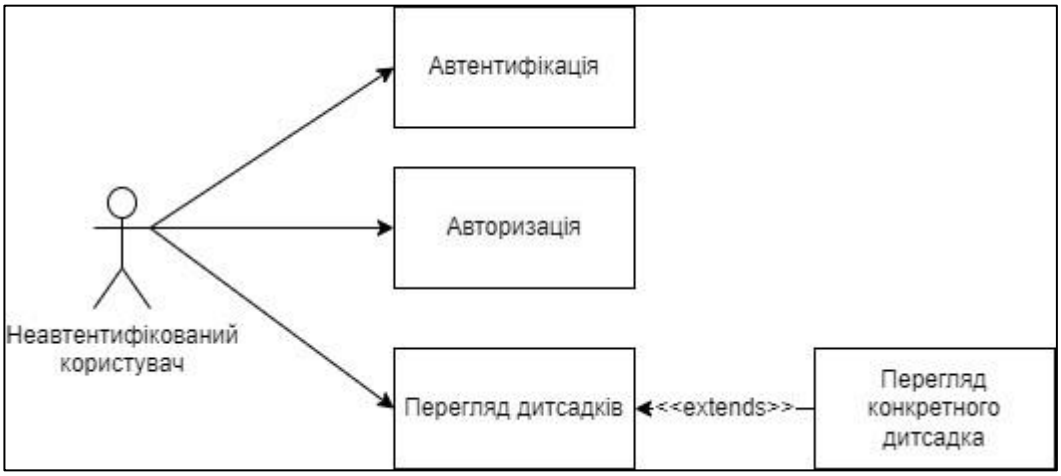


Рисунок 3.1 – Неавтентифікований варіант використання

3.3.2 Варіант використання батьків

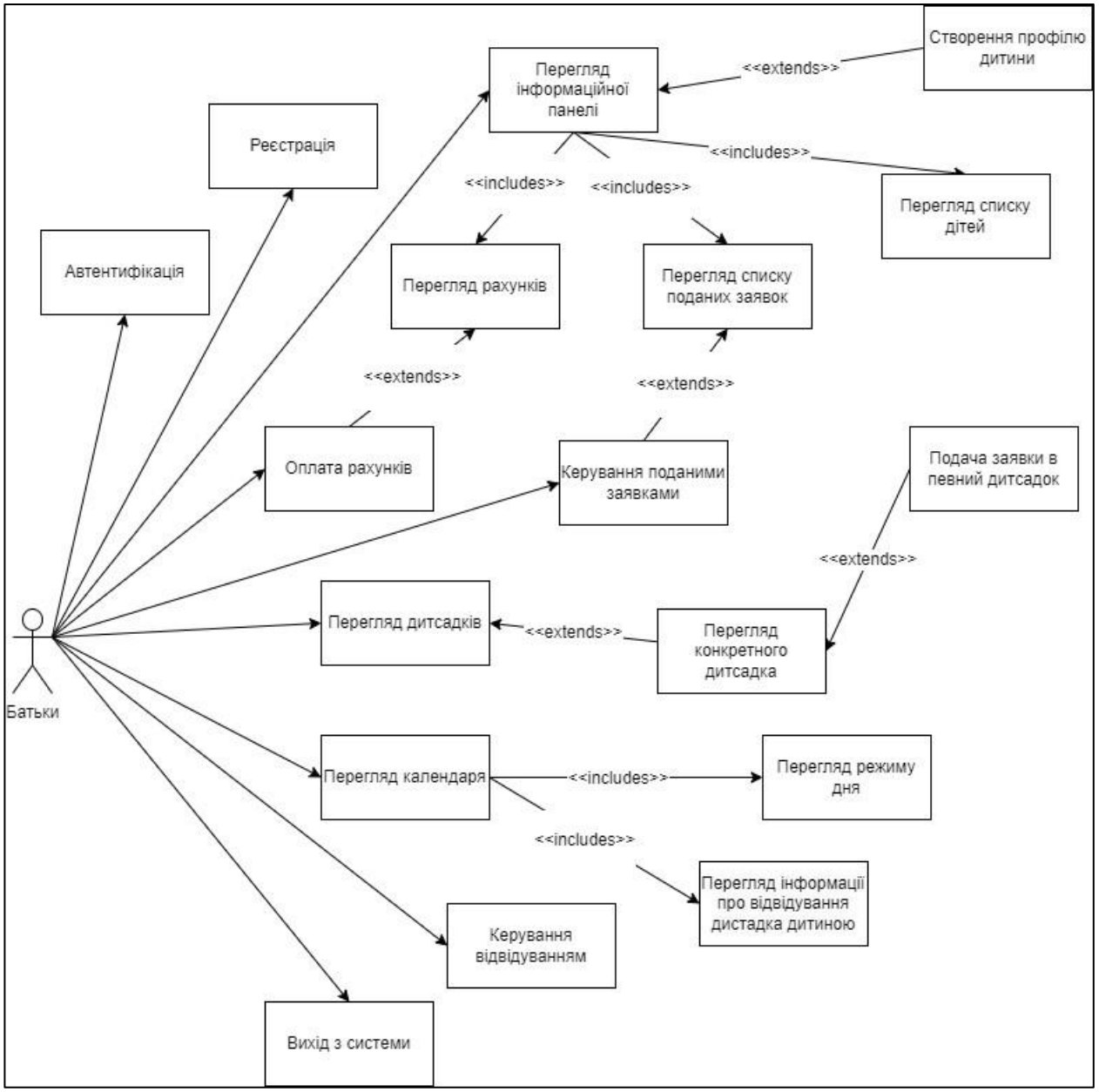


Рисунок 3.2 – Варіант використання батьків

3.3.2 Варіант використання вихователів

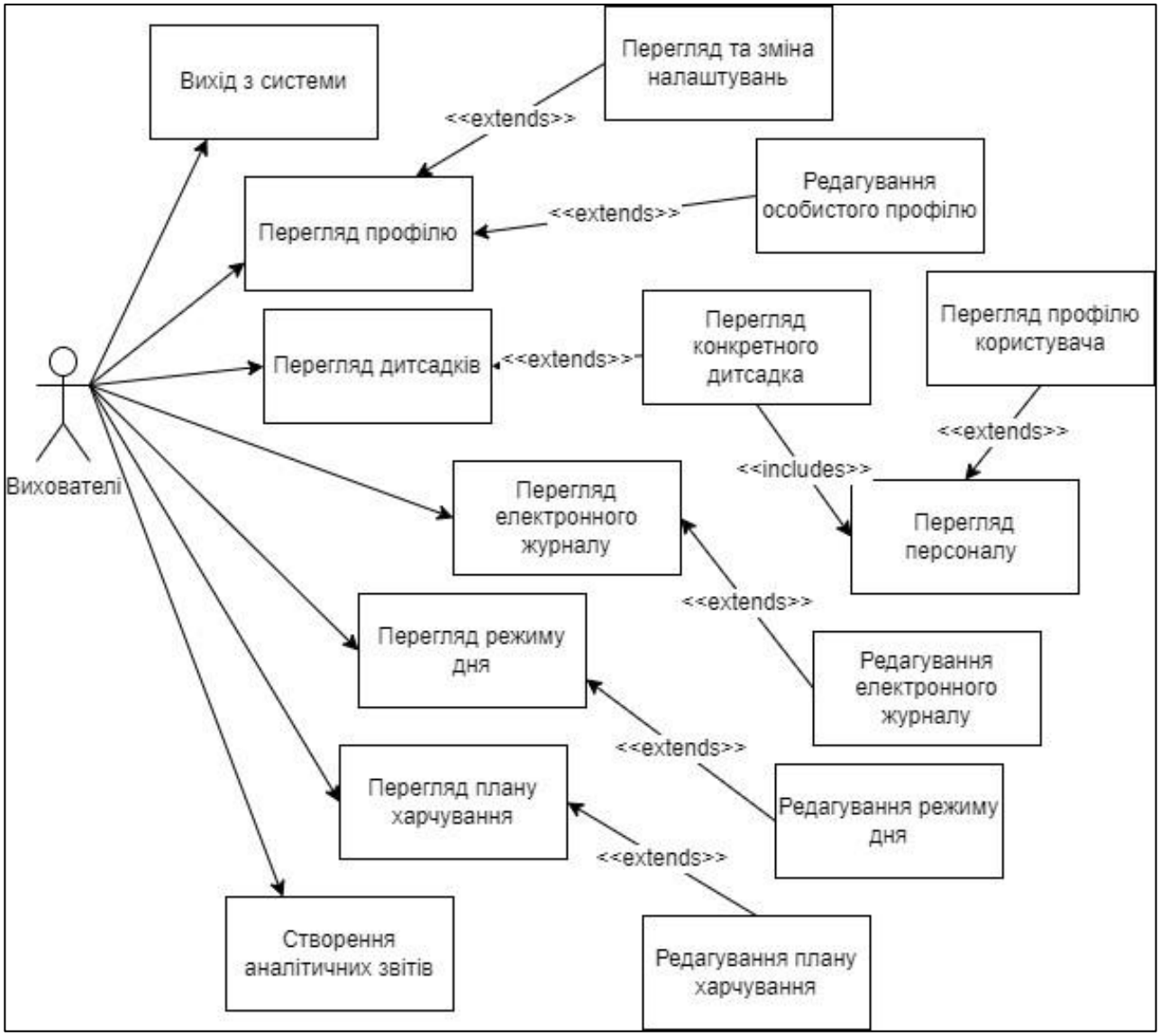


Рисунок 3.3 – Варіант використання вихователів

3.3.3 Варіант використання адміністраторів садочків

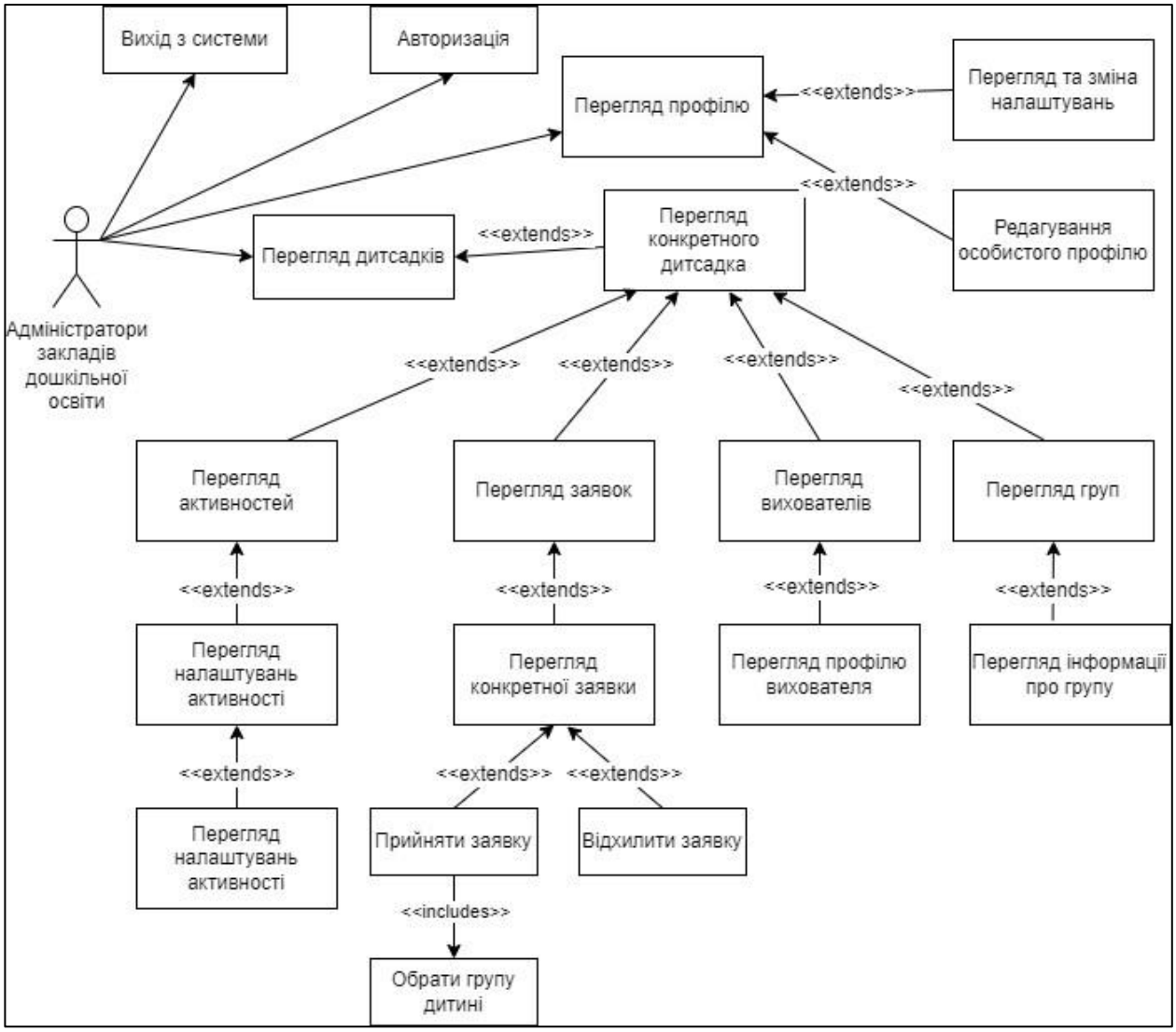


Рисунок 3.4 – Варіант використання адміністраторів садочків

3.4 Класи / Об’єкти

3.4.1 Group

3.4.1.1 Атрибути

Id – text

PreschoolId – text

Name – text

Description – text

MinAge – int

MaxAge – int

StartDateTime – datetime

EndDateTime – datetime

3.4.2 Child

3.4.2.1 Атрибути

Id – text

ParentId – text

GroupId – text

FirstName – text

LastName – text

BirthDate - date

Gender – text

Weight – float

3.4.3 Preschool

3.4.3.1 Атрибути

Id – text

Name – text

Description – text

ContactNumber – text

Email – text

City – text

Country – text

Address – text

Website – text

Facilities – text

OpeningHours – datetime

ClosingHours – datetime

OpenRegistration – bool

PricePerDayInUah – decimal

3.4.4 Attendance

3.4.4.1 Атрибути

Id – text

ChildId – text

Date – date

DropOffTime – datetime

PickUpTime – datetime

3.4.5 Application

3.4.5.1 Атрибути

Id – text

ChildId – text

PreschoolId – text

GroupId – text

Status – int

Comment – text

3.4.6 Invoice

3.4.6.1 Атрибути

Id – text

ChildId – text

ExternalId – text

Status – int

DateCreated – datetime

DueDate – datetime

Period – date

PayUrl – text

3.4.7 ScheduleActivity

3.4.7.1 Атрибути

Id – text

ActivityId – text

GroupId – text

StartDateTime – datetime

End DateTime – datetime

3.5 Нефункціональні вимоги

3.5.1 Вимоги до продуктивності

1. Веб-додаток і мобільний клієнт мають відображати вміст у відповідний спосіб, забезпечуючи оптимальну взаємодію з користувачем на різних пристроях і розмірах екрана.

2. Мобільні сторінки мають завантажуватися протягом 5 секунд, зберігаючи стабільну продуктивність веб-версії.

3.5.2 Вимоги до надійності

1. Впроваджувати надійні механізми обробки помилок, щоб ефективно обробляти несподівані помилки та запобігати збоям системи.

2. Вести докладні журнали помилок, включаючи відповідну інформацію для налагодження та аналізу.

3. Забезпечити цілісність транзакцій, особливо в критичних операціях, шляхом впровадження таких механізмів, як транзакції бази даних.

4. Спроектувати систему для плавного відновлення після збою, мінімізуючи вплив на користувачів.

3.5.3 Вимоги до доступності

1. Система має прагнути до мінімум 99,9% часу безвідмовної роботи, враховуючи вікна планового технічного обслуговування.

2. Програма має бути розроблена з механізмами відмовостійкості для обробки несподіваних збоїв у апаратному забезпеченні, програмному забезпеченні або мережевих компонентах.

3. Регулярно виконувати автоматичне резервне копіювання даних програми, включаючи бази даних і конфігураційні файли.

3.5.4 Вимоги до безпеки

1. Система повинна реалізовувати безпечні механізми автентифікації як для веб-програми, так і для мобільного клієнта.

2. Облікові дані користувача мають надійно зберігатися за допомогою галузевих стандартних методів шифрування.

3. Необхідно запровадити контроль доступу на основі ролей, щоб гарантувати, що користувачі мають доступ лише до ресурсів і функцій, які відповідають їхнім ролям.

4. Усі конфіденційні дані, включно з обліковими даними користувача, мають передаватися захищеними каналами з використанням протоколів шифрування, таких як HTTPS.

5. База даних програми повинна реалізовувати шифрування для конфіденційної інформації в стані спокою.

3.5.5 Вимоги до ремонтопридатності

1. Розробити систему з модульною архітектурою, що дозволяє незалежну розробку та підтримку компонентів.

2. Чітко визначені інтерфейси та залежності між модулями для полегшення майбутніх оновлень.

3. Використовуйте систему контролю версій (наприклад, Git) для відстеження змін у вихідному коді, забезпечуючи співпрацю між командами розробників і забезпечуючи історію змін коду.

3.5.6 Вимоги до портативності

1. Веб-програма сумісна з основними веб-браузерами, включаючи, але не обмежуючись ними, Google Chrome, Mozilla Firefox, Microsoft Edge і Safari.

2. Програма працює узгоджено в різних браузерах і версіях.

3. Мобільний клієнт сумісний з основними мобільними операційними системами, такими як iOS і Android.

3.6 Обмеження та винятки (Inverse Requirements)

1. Веб-програма може не працювати оптимально в застарілих або непідтримуваних веб-браузерах.

2. Сумісність не гарантується для операційних систем, відмінних від Windows, macOS і Linux.

3. Мобільний клієнт може не підтримуватися на пристроях із застарілою операційною системою або обмеженими апаратними можливостями.

4. Сумісність обмежена платформами iOS і Android, інші мобільні операційні системи не підтримуються.

5. Хоча мобільний клієнт підтримує основні функції в автономному режимі, певні функції можуть бути обмежені або взагалі не працювати без підключення до Інтернету.

3.7 Архітектурні обмеження

Система має відповідати відповідним галузевим стандартам і вказівкам, таким як стандарти безпеки (наприклад, ISO/IEC 27001) і стандарти веб-доступності (наприклад, WCAG), щоб забезпечити відповідність юридичним і нормативним вимогам.

Система має відповідати положенням про захист даних, таким як Загальний регламент захисту даних (GDPR), забезпечуючи безпечну обробку та обробку даних користувачів.

3.8 Вимоги до збереження даних (Logical Database Requirements)

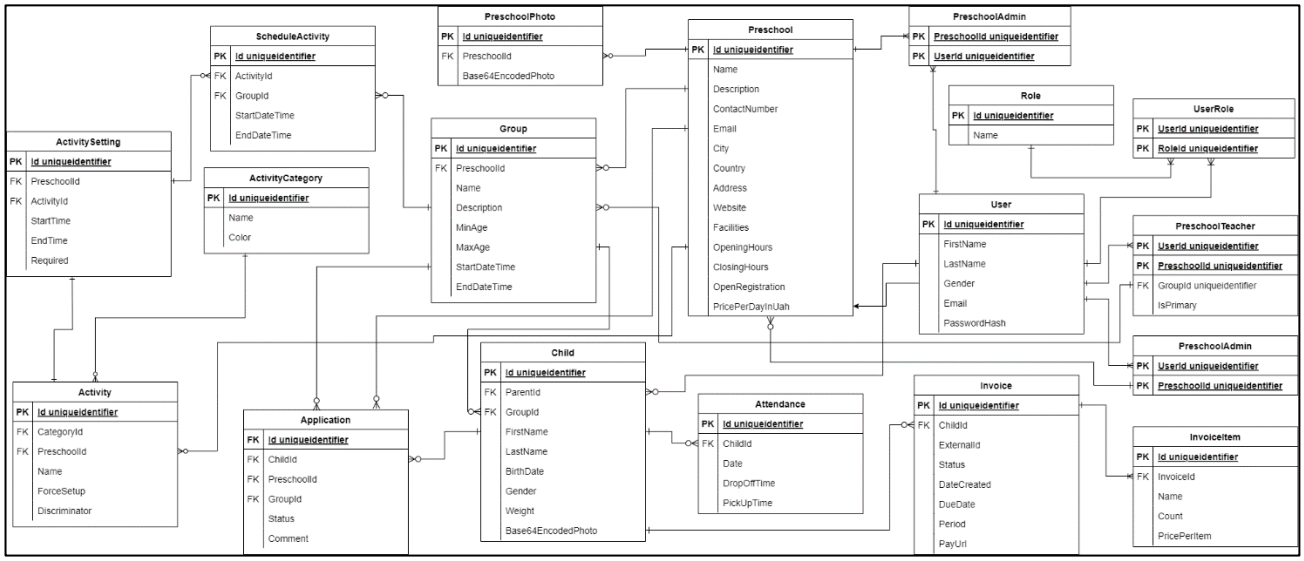


Рисунок 3.5 – ER-діаграма

3.9 Інші вимоги

Інших вимог до продукту немає.

4. Моделі аналізу

4.1 Діаграма послідовності

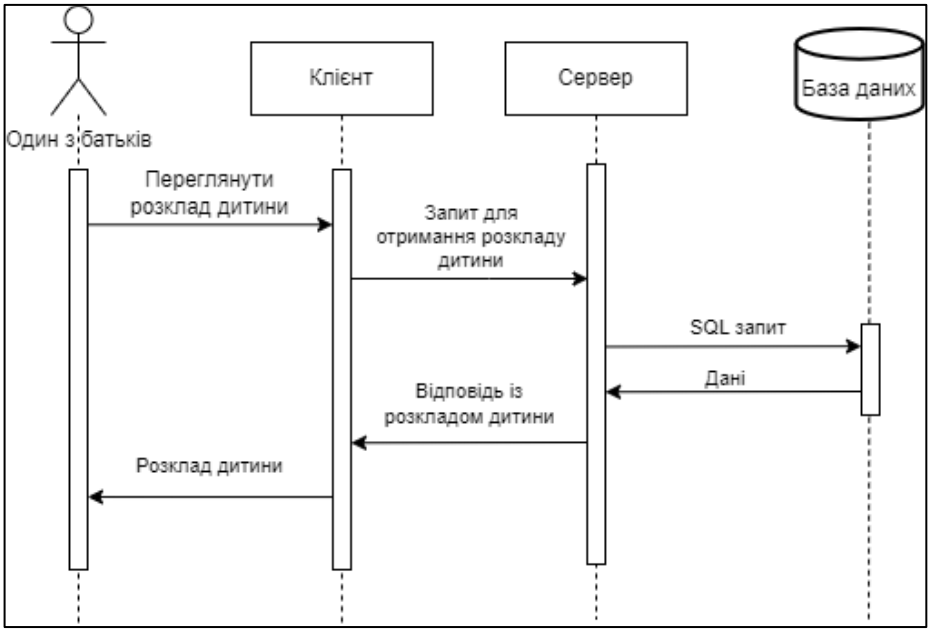


Рисунок 4.1 – Діаграма послідовності перегляду розкладу дитини

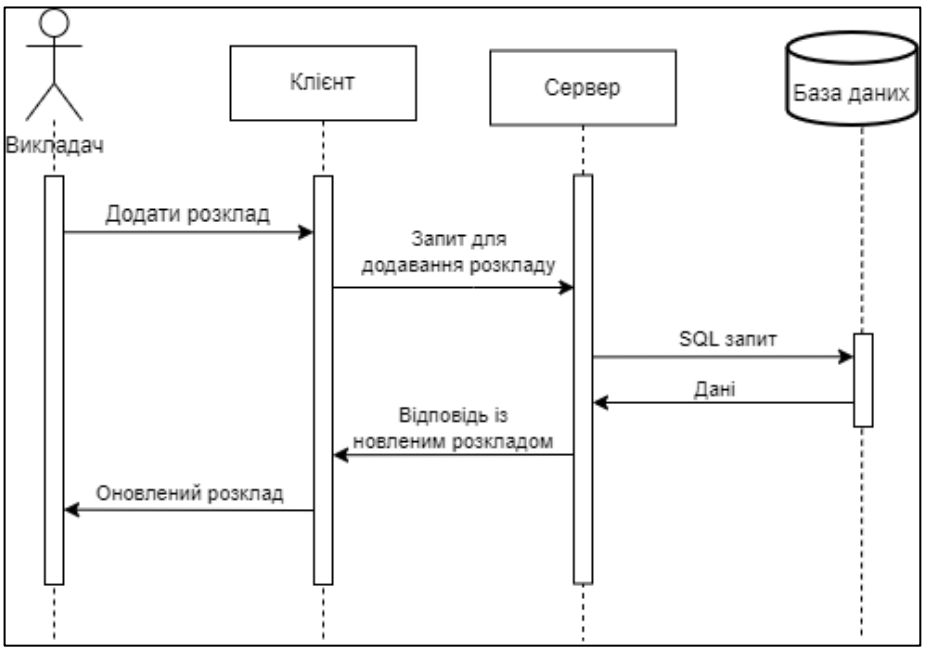


Рисунок 4.2 – Діаграма послідовності додавання розкладу викладачем

4.2 Діаграма переходу станів (STD)

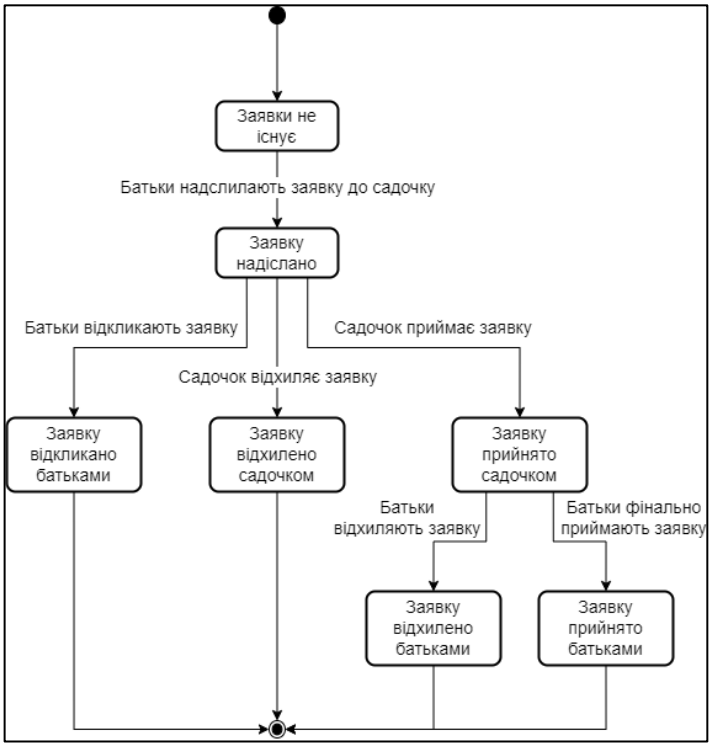


Рисунок 4.3 – Діаграма переходу станів заявки до садочку

Рисунок 4.4 – Діаграма переходу станів відвідуваності дитини

Рисунок 4.5 – Діаграма переходу станів генерації розкладу

4.3 Діаграма потоку даних (DFD)

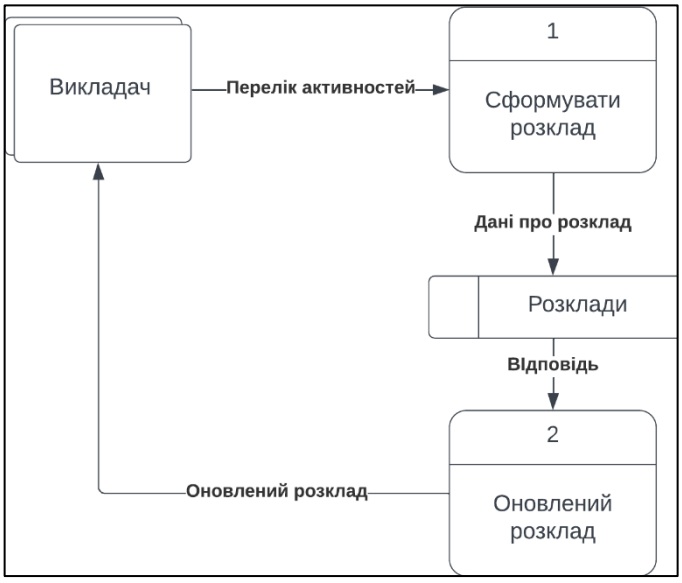


Рисунок 4.6 – Діаграма потоку даних формування розкладу викладачем

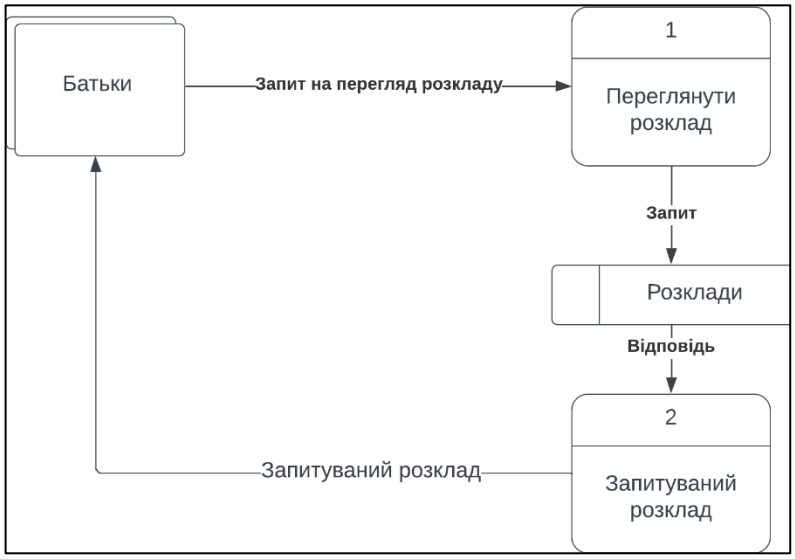


Рисунок 4.7 – Діаграма потоку даних перегляду розкладу батьками

5. Процес управління змінами

Зміни до SRS можуть бути запропоновані будь-яким членом команди, це може бути викликане змінами вимог до системи або обсягу робіт. Наступним кроком повинно бути проведено обговорення щодо змін, після чого визначений член команди вносить схвалені зміни в документ. Оновлена версія має бути внесена в таблицю та підписана кожним членом команди.

Додаток Г

Порівняльна таблиця аналогів

Порівняння відбувалося за такими критеріями:

– наявність інструментів для генерації розкладу з урахуванням особливостей дітей та їхніх потреб (А);

– можливість відстеження інформації про режим дня або розклад дітей та надання цієї інформації батькам (Б);

– функції подачі заявки до садка через додаток (В);

– можливість оплатити рахунки (Г);

– наявність інструментів для вчителів та адміністраторів для зручного управління закладом (Д);

– можливість встановлення та контролю процесу видачі дітей особам, визначеним батьками чи опікунами (Е);

– наявність україномовного інтерфейсу (Ж);

Таблиця Г.1. Порівняння програмних продуктів для планування розкладу та забезпечення інформації про харчування дітей в закладах дошкільної освіти

Програмний продукт	А	Б	В	Г	Д	Е	Ж
«Teach 'n Go»	-	+	-	+	+	-	-
«LineLeader»	-	-	+	-	+	-	-
«Preschool2me»	-	+	-	-	+	-	-
«Procare»	-	+	+	+	+	+	-
«KidCheck»	-	-	-	-	+	+	-
Програмний продукт, що розробляється	+	+	+	+	+	+	+

Додаток Д

ER діаграма основної бази даних системи «ChildWell»

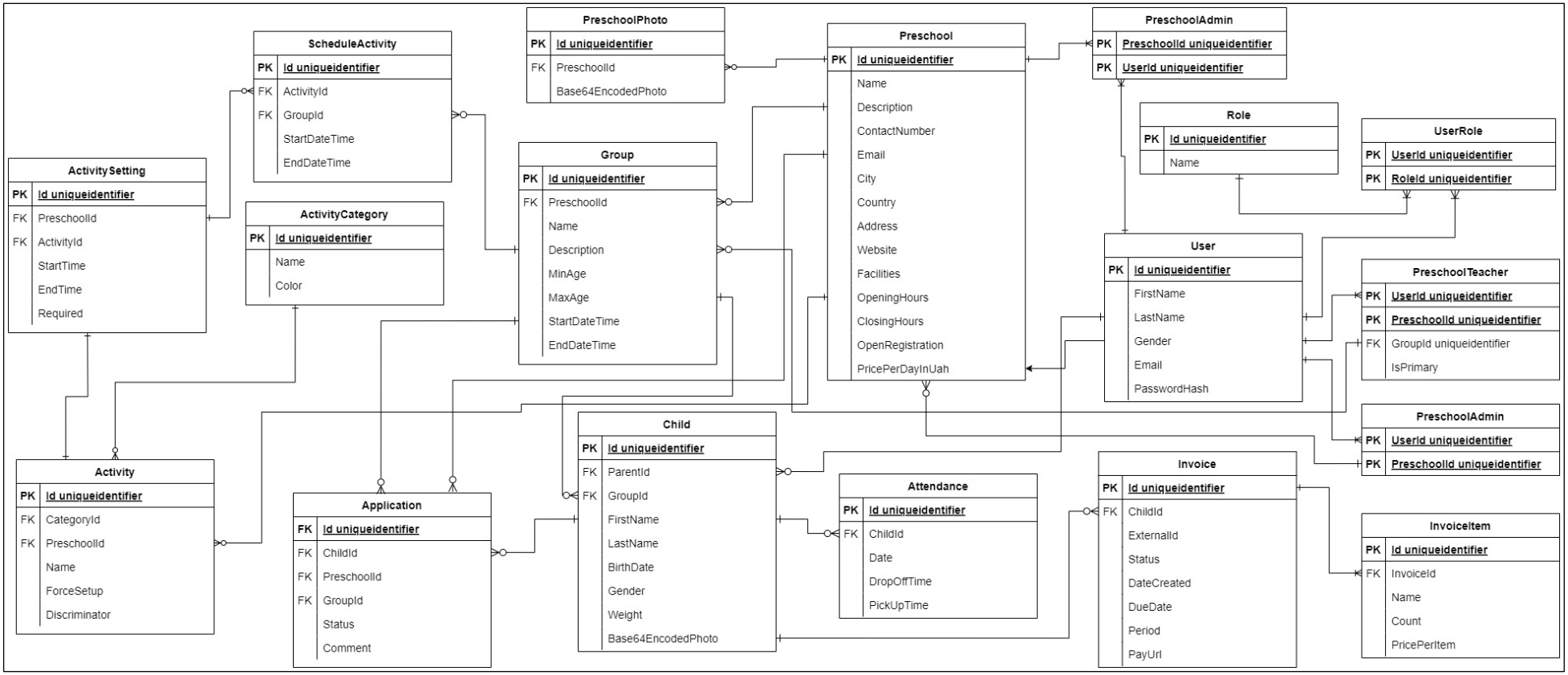


Рисунок Е.1 – ER діаграма програмної системи «ChildWell» (рисунок виконаний самостійно)

Додаток Е

MobX store, розроблений для дитячого садка

```
import { makeAutoObservable, runInAction } from "mobx";
import agent from "../api/agent";
import { Preschool } from "../models/preschool";
import { Activity } from "../models/activity";
import { PreschoolSetting } from "../models/preschoolSetting";
import { format } from "date-fns";

export default class PreschoolStore {
  preschoolRegistry = new Map<string, Preschool>();
  preschoolAdminRegistry = new Map<string, Preschool>();
  loadingInitial = false;
  loading = false;
  selectedPreschool: Preschool | undefined = undefined;
  activityRegistry = new Map<string, Activity>();
  settingsRegistry = new Map<string, PreschoolSetting[]>();

  constructor() {
    makeAutoObservable(this);
  }

  loadPreschools = async () => {
    this.setLoadingInitial(true);
    try {
      const preschools = await agent.Preschools.list();
      runInAction(() => {
        preschools.forEach((preschool) => {
          this.setPreschool(preschool);
        });
        this.setLoadingInitial(false);
      });
    } catch (error) {
      console.log(error);
      this.setLoadingInitial(false);
    }
  };

  loadPreschoolsForAdmin = async () => {
    this.setLoadingInitial(true);
    try {
      const preschools = await agent.Preschools.adminList();
      runInAction(() => {
        preschools.forEach((preschool) => {
          console.log(preschool.name);
          this.preschoolAdminRegistry.set(preschool.id, preschool);
        });
        this.setLoadingInitial(false);
      });
    } catch (error) {
      console.log(error);
      this.setLoadingInitial(false);
    }
  };
```

```
    get adminPreschoolsList() {
      this.setLoadingInitial(true);
      console.log("Preschools list");

console.log(Array.from(this.preschoolAdminRegistry.values())[0]);
      this.setLoadingInitial(false);
      return Array.from(this.preschoolAdminRegistry.values());
    }

    private getPreschool = (id: string) => {
      return this.preschoolAdminRegistry.get(id);
    };

    loadPreschool = async (id: string) => {
      let preschool = this.getPreschool(id);
      if (preschool) {
        this.selectedPreschool = preschool;
        console.log(this.selectedPreschool.name);
        return preschool;
      } else {
        this.setLoadingInitial(true);
        try {

          preschool = await agent.Preschools.details(id);
          this.setPreschool(preschool);
          runInAction(() => (this.selectedPreschool = preschool));
          this.setLoadingInitial(false);
          return preschool;
        } catch (error) {
          console.log(error);
          this.setLoadingInitial(false);
        }
      }
    };

    loadPreschoolSettings = async (preschoolId: string) => {
      this.setLoading(true);
      console.log("Loading (true):", this.loading);
      try {
        const settings = await
agent.Preschools.settingsList(preschoolId);
        runInAction(() => {
          this.settingsRegistry.set(preschoolId, settings);
          console.log(this.settingsRegistry.get(preschoolId));
        });
        this.setLoading(false);
        console.log("Loading (false):", this.loading);
      } catch (error) {
        console.log(error);
        this.setLoading(false);
      }
    };

    getSettingsForPreschool = (preschoolId: string):
                                  PreschoolSetting[] => {
      const settings = this.settingsRegistry.get(preschoolId);
```

```
   106      if (!settings) return [];
   107
   108      return Array.from(settings);
   109    };
   110
   111    updateSettingsOfPreschool = async (id: string, setting:
   112                                       PreschoolSetting) => {
   113      console.log("Preschool id: ", id);
   114      console.log("Setting to update: ", setting);
   115      try {
   116        runInAction(() => {
   117          if (this.settingsRegistry.has(id)) {
   118            const settings = this.settingsRegistry.get(id);
   119            const updatedSettings = settings?.map((set) =>
   120              set.activityId === setting.activityId ? { ...set,
   121                                                        ...setting
} : set
   122            );
   123            if (updatedSettings) {
   124              this.settingsRegistry.set(id, updatedSettings);
   125            }
   126          }
   127        });
   128
   129        await agent.Preschools.updateSettings(
   130          id,
   131          this.getSettingsForPreschool(id)
   132        );
   133      } catch (error) {
   134        console.log(error);
   135      }
   136    };
   137
   138    get preschoolsList() {
   139      return Array.from(this.preschoolRegistry.values());
   140    }
   141
   142    setLoadingInitial = (state: boolean) => {
   143      this.loadingInitial = state;
   144    };
   145
   146    setLoading = (state: boolean) => {
   147      this.loading = state;
   148    };
   149
   150    private setPreschool = (preschool: Preschool) => {
   151      this.preschoolRegistry.set(preschool.id, preschool);
   152    };
   153
   154    loadActivities = async (id: string) => {
   155      try {
   156        const activities = await agent.Preschools.activities(id);
   157        runInAction(() => {
   158          activities.forEach((globalActivity) => {
   159            this.activityRegistry.set(globalActivity.id,
globalActivity);
```

```
        });
      });
    } catch (error) {
      console.log(error);
    }

  };

  get activitiesList() {
    return Array.from(this.activityRegistry.values());
  }

  get groupedActivities() {
    return Object.entries(
      this.activitiesList.reduce(
        (activities: Record<string, Activity[]>, activity) => {
          const category = activity.category.id;
          activities[category] = activities[category]
            ? [...activities[category], activity]
            : [activity];
          return activities;
        },
        {} as Record<string, Activity[]>
      )
    );
  }

  setSelectedPreschool(id: string) {
    this.selectedPreschool = this.getPreschool(id);
    console.log(this.selectedPreschool?.id);
  }
}
function setForceRender(arg0: (prev: any) => boolean) {
  throw new Error("Function not implemented.");
}
```

Додаток Ж

Розроблена MUI тема

```
import { createTheme } from "@mui/material";

let theme = createTheme({
  palette: {
    primary: {
      main: "#055B5C",
    },
    secondary: {
      main: "#83749F",
    },
    error: {
      main: "#EF5350",
    },
    success: {
      main: "#66BB6A",
    },
    customGreen: {
      main: "#0E97A1",
    },
    lightPink: {
      main: "#CBACD8",
    },
    lightMint: {
      main: "#6CC5CB",
    },
    lightText: "#eceff1",
    darkText: "#757575",
  },
  components: {
    MuiButton: {
      defaultProps: {
        disableRipple: false,
        disableElevation: true,
      },
      styleOverrides: {
        root: {
          textTransform: "none",

          borderRadius: "8px",
          margin: "0.3rem",
        },
        outlinedSecondary: {
          border: "2px solid",
          "&:hover": {
            border: "2px solid",
            borderColor: "#CBACD8",
            backgroundColor: "#CBACD8",
            color: "#eceff1",
          },
        },
      },
    },
```

```
53      MuiDivider: {
54        styleOverrides: {
55          root: {
56            backgroundColor: "#055B5C",
57            height: "1px",
58          },
59        },
60      },
61      MuiTableCell: {
62        styleOverrides: {
63          root: {
64            color: "#757575",
65          },
66        },
67      },
68    },
69    typography: {
70      header: {
71        fontSize: "1.8rem",
72        fontWeight: 600,
73      }    74      logo: {
75        fontSize: "2.3rem",
76        fontWeight: 700,
77        fontFamily: "Ubuntu",
78      },
79    },
80  });
81
82  export default theme;
```