**МІНІСТЕРСТВО ОСВІТИ И НАУКИ УКРАЇНИ**

**ОДЕСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ імені І.І. МЕЧНИКОВА**

**Факультет математики, фізики та інформаційних технологій**

**Кафедра математичного забезпечення комп'ютерних систем**

**КУРСОВИЙ ПРОЕКТ**

з дисципліни

**«Організація баз даних»**

на тему:

**«Інформаційна система дитячого садку**»

|  |  |
| --- | --- |
|  | студента ІІI курсу  групи \_\_\_3\_\_\_\_\_  спеціальності \_\_Комп'ютерна інженерія (123)\_  \_Щупака Тимура Олександровича\_\_  (Прізвище, ім’я та по батькові)  Керівник:\_\_\_ст. викл. Розновець О.І.\_\_\_\_\_\_  Захищено «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_202\_\_ р.  з оцінкою \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Комісія:  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (ПIБ) (Підпис)  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (ПIБ) (Підпис)  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (ПIБ) (Підпис) |

**Одеса – 2022**

**АНОТАЦІЯ**

Мета даного курсового проекту – проектування і реалізація інформаційної системи для дитячого саду.

Користувачами даної системи є адміністратори, викладачі та батьки дітей,які навчаються в дитячому садку. Реалізація проекту виконана у вигляді веб додатку з використанням мови програмування JavaScript та СУБД PostgreSQL. Архітектура системи розроблена на основі стилю REST, який в свою чергу є однією з реалізацій триланкової архітектури клієнт-сервер.

У розробленій інформаційній системі передбачено перш за все зберігання та редагування інформації про дітей, їх батьків та викладачів . Основною функцією дитячого саду є електронний щоденник кожної дитини, за допомогою якого їх батьки завжди можуть дізнатися середній бал за весь час, який оновлюється з кожним записом в щоденник або ж продивитися розклад дитини.

Результатом курсового проектування є корпоративний веб додаток дитячого садку.

**ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ І ТЕРМІНІВ**

JSON – JavaScript Objective Notation.

JWT – JSON Web Token.

REST (Representational State Transfer) – передача “самоописуваного” стану.

API (Application Programming Interface) – програмний інтерфейс додатку.

**ЗМІСТ**

[АНОТАЦІЯ 2](#_Toc105051670)

[ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ І ТЕРМІНІВ 3](#_Toc105051671)

[ВСТУП 5](#_Toc105051672)

[1 ПОСТАНОВКА ЗАДАЧІ 6](#_Toc105051673)

[1.1 Типи користувачів системи 6](#_Toc105051674)

[1.2 Список задач користувачів ІС 7](#_Toc105051675)

[2 ПРОЕКТУВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ 12](#_Toc105051676)

[3 ІНФОРМАЦІЙНА МОДЕЛЬ ПРЕДМЕТНОЇ ОБЛАСТІ 15](#_Toc105051677)

[4 МОДЕЛЬ ПРОГРАМНОГО ДОДАТКУ 17](#_Toc105051678)

[5 ВИБІР ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ 18](#_Toc105051679)

[6 СТВОРЕННЯ БАЗИ ДАНИХ 20](#_Toc105051680)

[7 ЗАПИТИ ДЛЯ РОЗВ’ЯЗАННЯ ПОСТАВЛЕНИХ ЗАДАЧ 22](#_Toc105051681)

[8 ПРОГРАМНА РЕАЛІЗАЦІЯ ІНТЕРФЕЙСУ 29](#_Toc105051682)

[9 БЕЗПЕКА КЛІЄНТА ТА ІНФОРМАЦІЙНОЇ СИСТЕМИ 31](#_Toc105051683)

[10 ІНСТРУКЦІЯ КОРИСТУВАЧА 35](#_Toc105051684)

[10.1 Опікун……. 35](#_Toc105051685)

[10.2 Викладач 39](#_Toc105051686)

[10.4 Адміністратор 43](#_Toc105051688)

[ВИСНОВКИ 53](#_Toc105051689)

[СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 54](#_Toc105051690)

[ДОДАТОК А Опис сутностей та їх властивостей 55](#_Toc105051691)

[ДОДАТОК Б Схема бази даних 60](#_Toc105051692)

[ДОДАТОК В Запити для створення бази даних 61](#_Toc105051694)

[ДОДАТОК Г Запити на створення представлень і тригерів 67](#_Toc105051695)

**ВСТУП**

З розвитком науки і техніки та прискоренням темпу життя людини з’являється потреба у автоматизації всіх функцій. Дитячий садок не є винятком, часто досить важко вести журнали і зберігати сотні записів з тією чи іншою інформацією. Україна одна з найрозвинутіших країн Європи в сфері технологій. У нас найпершими прийняли електронні документи на рівні державних. Тому лікарні, університети, школи, дитячі садки все частіше переходять до електронної форми запису та зберігання інформації.

Проектований додаток призначений насамперед для батьків та викладачів які працюють в дитячому садку і дозволяє їм ефективніше продивлятися або записувати інформацію. Це все зроблено за рахунок:

* систематизації та швидкого пошуку потрібної їм інформації;
* надання звітів про успіхи дитини в навчанні;
* конфіденційність інформації для інших користувачів.

Мета даного курсового проекту – проектування і реалізація інформаційної системи для дитячого садку.

Для досягнення зазначеної мети необхідно вирішити наступні задачі:

* виконати аналіз предметної області «Дитячий садок»;
* визначити категорії користувачів і сформулювати їхні вимоги до створюваної інформаційної системи;
* обрати архітектуру і шаблон проектування створюваної інформаційної системи;
* спроектувати базу даних;
* обрати технології та засоби реалізації інформаційної системи;
* з урахуванням вибраних засобів створити БД, а також клієнтську програму, яка дасть можливість різним категоріям користувачів ефективно маніпулювати даними предметної області відповідно до їхніх повноважень;
* забезпечити цілісність і безпеку даних як на рівні БД, так і на рівні застосунку;
* забезпечити захист системи від несанкціонованого доступу і розмежування повноважень з боку різних категорій користувачів.

**1 ПОСТАНОВКА ЗАДАЧІ**

До основних задач інформаційної системи можна віднести наступні:

1. Реєстрація працівників, батьків та опікунів, дітей та створення сімейних зв’язків;
2. Надання інформації про працівників, дітей, батьків та опікунів;
3. Надання звітів про активність дітей, їх харчування та загальний настрій протягом дня, та запис цієї інформації у базі даних для подальшого занесення в атестат при випуску дитини з дитячого садку.
   1. **Типи користувачів системи**

Назви типів користувачів та їх кількість залежить від специфіки предметної області. Наприклад, у даній області «Дитячий садок» виділяються такі категорії користувачів:

1)Адміністратор – займається реєстрацією інших користувачів веб додатку. Створює розклад, додає нові предмети та назначає викладачів для них. Створює нові групи, добавляє в них дітей та назначає їх керівників. Може редагувати будь-яку й будь-чию інформацію в базі даних;

2)Викладач – займається викладанням предметів та виставленням оцінок в добовий звіт. Має повне право на перегляд будь-якої інформації. Але не може нічого редагувати окрім щоденника дитини.

3) Батьки, або опікуни дитини – має право на перегляд інформації про свою дитину або дітей, якщо їх декілька. Може продивлятися їх щоденник за розклад групи дитини або дітей.

* 1. **Список задач користувачів ІС**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Номер | | Задача | Вхідні данні | | Вихідні данні | |
|  | | **Адміністратор** |  | |  | |
| А1 | | Подивитися список працівників | Відсутні | | ФІО працівника та його статус | |
| А2 | | Подивитися інформацію  про працівника | Індекс працівника | | ФІП працівника, його посада, дата народження, дата прийому на роботу, дата закінчення контракту, адреса проживання, посада, номер телефону, електронна пошта | |
| А3 | | Подивитися списки груп дітей | Назва групи | | Назва групи и ФІП дітей | |
| А4 | | Подивитися інформацію про дитину | Індекс дитини | | ФІП дитини, дата народження, дата зарахування, дата випуску, адреса проживання, медичне заключення(інформація про алергії та протипоказання, якщо такі присутні, інакше там буде написано що «Протипоказань немає») гендер дитини, тип групи, назва групи | |
| А5 | | Перевести дитину в іншу групу | Індекс дитини, назва групи | | Змінені дані про дитину | |
| А6 | | Створити профіль працівника | ФІП працівника, його посада, дата народження, дата прийому на роботу, дата закінчення контракту, адреса проживання, посада, номер телефону, електронна пошта, пароль | | Профіль працівника | |
| А7 | | Створити нову групу дітей | Назва групи, її тип, керівник групи | | Нова група | |
| А8 | | Створити профіль дитини | ФІП дитини, дата народження, дата зарахування, дата випуску, адреса проживання, медичне заключення(інформація про алергії та протипоказання, якщо такі присутні, інакше там буде написано що «Протипоказань немає») гендер дитини, тип групи, назва групи | | Профіль дитини | |
| А9 | | Видалити групу | Порожня група | | Видалена група | |
| А10 | | Редагування групи | Нова назва, новий тип, новий керівник | | Відредагована група | |
| А11 | | Звільнити працівника | Новий статус | | Змінений статус | |
| А12 | | Додати предмет | Назва предмету, індекс викладача | | Новий предмет | |
| А13 | | Створення розкладу | День, час, предмет, група | | Новий предмет в розкладі | |
| А14 | | Реєстрація батьків або опікунів | ФІП опікуна, його тип, дата народження , адреса проживання, адреса місця роботи, посада, номер телефону, електронна пошта,  пароль | | Профіль опікуна | |
| А15 | | Подивитися профіль опікуна | Індекс працівника | | ФІП опікуна, його тип, дата народження , адреса проживання, адреса місця роботи, посада, номер телефону, електронна пошта | |
| А16 | Видалити предмет | | | Індекс предмету | | Предмет видалено | |
| А17 | Видалити опікуна | | | Індекс опікуна | | Опікуна видалено | |
| А18 | Випустити дитину | | | Індекс дитини | | Дитина випустилась | |
| A19 | Додати новину | | | Тема, текст | | Новина з датою, темою, та текстом | |
|  | | **Працівник** |  | |  | |
| П1 | | Подивитися список дітей за групами | Назва групи | | Назва групи і список дітей. | |
| П2 | | Подивитися інформацію про дитину | Індекс дитини | | ФІП дитини, дата народження, дата зарахування, дата випуску, адреса проживання, медичне заключення(інформація про алергії та протипоказання, якщо такі присутні, інакше там буде написано що «Протипоказань немає») гендер дитини, тип групи, назва групи | |
| П3 | | Зробити добовий звіт дитини | Індекс дитини, оцінка активності, оцінка харчування, оцінка загального настрою | | Добовий звіт | |
| П4 | | Подивитися розклад | Назва групи | | День неділі, час початку предмету, викладач предмету | |
| П5 | | Подивитися профіль опікуна | Індекс працівника | | ФІП опікуна, його тип, дата народження , адреса проживання, адреса місця роботи, посада, номер телефону, електронна пошта | |
|  | | **Опікуни** |  | |  | |
| О1 | | Подивитись розклад | Назва групи | | День неділі, час початку предмету, викладач предмету | |
| О2 | | Подивитися добовий звіт | Знаходиться в профілі дитини | | Добовий звіт | |
| О3 | | Подивитись головну сторінку дитячого саду. Його опис. Новини, плани. | Новинна панель на головній сторінці сайту | | Загальна інформація та новинна панель | |
| О4 | | Подивитись середні оцінки дитини за весь час | Знаходиться в профілі дитини | | Середні бали за весь час | |

**2 ПРОЕКТУВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ**

При проектуванні архітектури інформаційної системи обрано триланкову архітектуру клієнт-сервер та архітектурний стиль REST.

REST – це стиль архітектури програмного забезпечення для

розподілених систем, таких як World Wide Web, який зазвичай використовується для побудови веб-служб. Термін REST був запроваджений у 2000 році Роєм Філдінгом, одним із авторів HTTP-протоколу. Системи, що підтримують REST, називаються RESTful-системами. (Малюнок 2.1) [1].

У випадку REST є дуже простим інтерфейсом управління без використання будь-яких додаткових внутрішніх шарів. Кожна одиниця інформації однозначно визначається світовим ідентифікатором, наприклад URL. Кожна URL, своєю чергою, має строго заданий формат. В ідеальному випадку одна програма-сервер може виконувати запити від багатьох програм-клієнтів, тому її слід розміщувати на спеціально виділеній обчислювальній машині, налаштованій особливим чином, як правило, спільно з іншими програмами-серверами, тому продуктивність цієї машини максимізується.

Через особливу роль такої машини в мережі, специфіки її обладнання та програмного забезпечення її також називають сервером, а машини, що виконують клієнтські програми, відповідно клієнтами.

Клієнт (рівень клієнта) – це комп'ютер або програма, яка звертається до сервера і запитує необхідні для роботи користувача дані, зазвичай має графічний інтерфейс, що допомагає користувачу зручно та легко працювати з отриманою інформацією. Крім цього клієнт не тільки отримує інформацію, а й обробляє її – наприклад, здійснює обчислення та перетворення даних.

Цей рівень не повинен мати прямі зв'язки з базою даних (за вимогами безпеки та масштабованості) та зберігати стан програми (за вимогами надійності).

Сервер (рівень даних) надає функцію або послугу одному або декільком клієнтам, які ініціюють запити. Сервери класифікуються за наданими послугами. Наприклад, веб-сервер обслуговує веб-сторінки, файловий сервер обслуговує комп'ютерні файли, сервер бази даних надає доступом до баз даних.

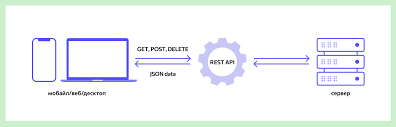
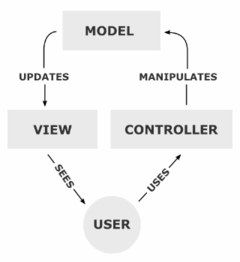


Рисунок 2.1 – Схема архітектури REST

Ґрунтуючись на архітектурі, необхідно також обрати шаблон проектування додатку. Відповідним можна назвати шаблон MVC (рисунок 2.2).

Model-View-Controller – схема поділу даних програми, призначених для користувача інтерфейсу та керуючої логіки на три окремі компоненти: модель, уявлення та контролер – таким чином, що модифікація шкірного компонента може здійснюватися незалежно [2].



Малюнок 2.2 – Схема шаблону MVC

Model надає дані та способи роботи з ними: запити до бази даних, перевірка на коректність. Модель не залежить від подання та контролера, просто надаючи доступ до даних та управління ними. За рахунок незалежності від візуального подання може мати кілька різних уявлень для однієї моделі.

View відповідає за відображення даних моделі користувача, реагуючи на зміни моделі. Отримання необхідних даних з моделі та надсилання їх користувачеві. Подання не робить обчислень з даними користувача.

Controller інтерпретує дії користувача, сповіщаючи модель необхідність змін. Цей елемент забезпечує зв'язок між користувачем та системою. Контроль та спрямування даних від користувача в систему і навпаки. Використовує модель та подання для реалізації необхідної дії.

Покроковий принцип роботи програми під час використання шаблону MVC:

1) Контролер отримує наступний запит від користувача;

2) Далі, залежно від внутрішньої логіки:

a) Формується подання будь-якої сторінки;

б) Або викликані методи моделі;

3) Модель повідомляє про зміни;

4) Подання оновлюється (якщо в ланцюжку була задіяна модель) та відображається користувачеві.

**3 ІНФОРМАЦІЙНА МОДЕЛЬ ПРЕДМЕТНОЇ ОБЛАСТІ**

В інформаційній системі дитячого саду слід виділити наступні сутності:

-працівник – містить повну інформацію про працівника;

-предмет – містить наступну інформацію: назва предмету та його викладач;

-розклад – містить наступну інформацію: день неділі, час початку предмета, назва предмета, назва групи;

-група – містить наступну інформацію: назва групи, її тип, керівник групи;

-дитина – містить інформацію про дитину та назву групи;

-звіт – містить наступну інформацію: індекс дитини, дата , оцінки: активність, харчування та настрій загалом;

-сім'я – містить інформацію про родинні зв'язки дітей та їх батьків чи опікунів

-опікун – містить повну інформацію про опікуна чи батьків

Детальніше сутності та їх властивості з описом обмежень, що потрібні для розв’язання поставлених задач, наведено в таблиці у додатку А.

Між сутностями у базі даних наявні 2 типи зв’язків:

* «один-до-одного»;
* «один-до-багатьох» ;
* «багато-до-багатьох».

Зв'язок «один-до-одного» реалізований у працівника і групи. У кожної групи є свій керівник. Для формалізації зв'язку у таблиці group2\_0 є зовнішній ключ, який посилається на первинний ключ таблиці employee2\_0.

Зв'язок «один-до-багатьох» наявний між декількома парами таблиць у базі даних, їх буде розглянуто по порядку:

-Працівник та предмет. Кожен працівник може викладати один, або декілька предметів. Деякі працівники не викладають жодного предмету;

* Предмет та розклад. В розкладі присутні безліч предметів. Для формалізації зв'язку у таблиці schedule2\_0 є зовнішній ключ, який посилається на первинний ключ таблиці subject2\_0;
* Група та розклад. В одному розкладі присутні безліч груп. Для формалізації зв'язку у таблиці schedule2\_0 є зовнішній ключ, який посилається на первинний ключ таблиці group2\_0;
* Група та діти. В одній групі присутні до 20 дітей. Для формалізації зв'язку у таблиці child2\_0 є зовнішній ключ, який посилається на первинний ключ таблиці group2\_0;
* Діти та звіти. Для однієї дитини існує безліч звітів. Для формалізації зв'язку у таблиці dairy2\_0 є зовнішній ключ, який посилається на первинний ключ таблиці child2\_0.

Зв’язок «багато-до-багатьох» реалізований між батьками чи опікунами та дітьми через проміжню таблицю family2\_0 яка містить зовнішній ключ таблиці family2\_0 та зовнішній ключ таблиці child2\_0. У одного опікуна може бути від одної до безліч дітей. А у дитини може бути до 5 опікунів різних типів.

Схему бази даних, що ілюструє сутності та зв’язки між ними, наведено у додатку Б (рисунок Б.1).

**4 МОДЕЛЬ ПРОГРАМНОГО ДОДАТКУ**

Адміністратори, працівники та опікуни користуються єдиним додатком.

При авторизації у ролі одного з користувачів, додаток завантажує головну сторінку, в залежності від ролі користувача їм буде відображено певний функціонал для виконання задач, виконаного у формі кнопок.

Система гармонічно розподілена на директорії та файли, кожен з яких виконує лише одну певну дію. Клієнт з UI посилає запит контролеру (який збережений в однойменному файлі директорії Controllers), а контролер передає інформацію наступній ланці системи, де вже виконуються операції із БД (Model). Ієрархія сторінок програмного додатку наведена у додатку В.

**5 ВИБІР ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ**

Інформаційна система розроблена у вигляді веб-додатку. Для роботи з базою даних обрано СУБД PostgreSQL, а для розробки програмного додатку використовується платформа Node.js із фреймворком Express.js (серверна частина) та бібліотека React.js із використанням готових бібліотечних компонентів react-bootstrap (клієнтська частина).

Node.js або Node – програмна платформа, заснована на движку V8, що перетворює JavaScript з вузькоспеціалізованої мови на мову загального призначення .

Node.js — середовище виконання коду JavaScript поза браузером, яке дозволяє писати серверний код для веб-сторінок і веб-застосунків, а також для програм командного рядка. За допомогою Node.js реалізується парадигма JavaScript для всього. Вона передбачає використання однієї мови програмування для розробки веб-застосунків замість застосування різних мов для роботи над фронтендом та бекендом.

Node.js – не окрема мова програмування, а платформа для використання JavaScript на стороні сервера. Якщо говорити про мову, то як для фронденда, так і для бекенда використовується той самий JavaScript. Різниця тільки в наборі API, які використовують фронтендери та бекендери.

Браузерний JavaScript використовує Web API, які забезпечують доступ до DOM і інтерфейсу сторінок і веб-додатків. Серверний JavaScript використовує API, що забезпечують доступ до файлової системи додатків, http-запитів, потоків[3].

Express.js – фреймворк web-додатків для Node.js, реалізований як вільне та відкрите програмне забезпечення під ліцензією MIT. Він спроектований для створення веб-додатків та API [4].

Список задач, які реалізує JavaScript:

* фронтенд та бекенд сайтів;
* мобільні додатки;
* десктопні програми;
* ігри;
* веб-сервери;
* презентації.

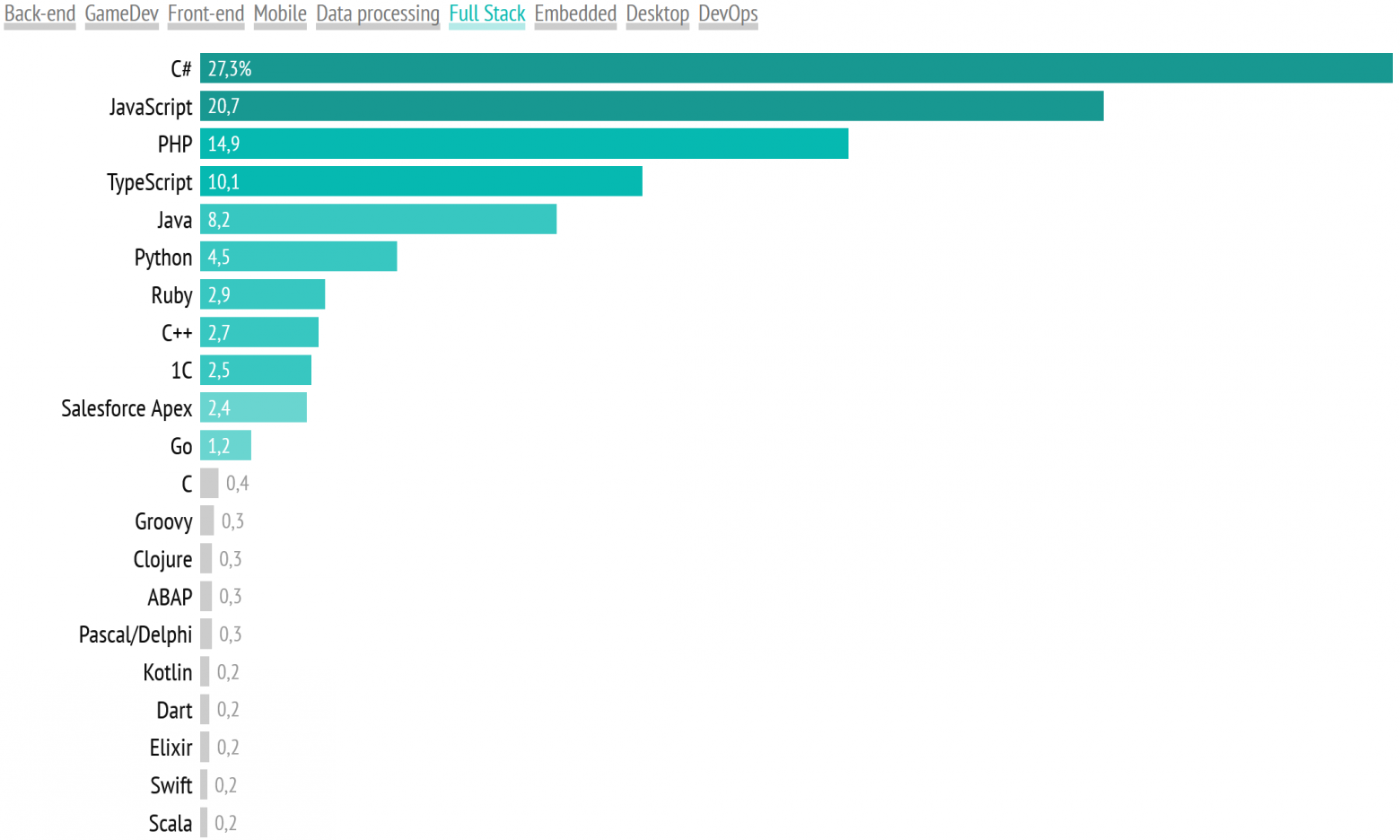


Рисунок 5.1 – Рейтинг мов програмування, якими пишуть fullstack розробники [5]

**6 СТВОРЕННЯ БАЗИ ДАНИХ**

В цьому розділі описано створення основних об’єктів бази даних. Як приклад наведено програмний код створення таблиці employee2\_0 (лістинг 5.1) Повний код створення всіх таблиць БД наведено в додатку Г.

CREATE TABLE IF NOT EXISTS employee2\_0 (

employee\_id SERIAL PRIMARY KEY NOT NULL,

lname\_ VARCHAR(50) NOT NULL,

fname\_ VARCHAR(50) NOT NULL,

fathername\_ VARCHAR(50) NOT NULL,

position\_ VARCHAR(50) NOT NULL,

email VARCHAR(50) NOT NULL UNIQUE,

birth\_date date NOT NULL,

status smallint NOT NULL,

contract\_expiration date NOT NULL,

password\_ VARCHAR(250) NOT NULL,

adress VARCHAR(50) NOT NULL,

phone\_num VARCHAR(20) NOT NULL UNIQUE,

CONSTRAINT lname\_length check (char\_length(lname\_)>=3),

CONSTRAINT fname\_length check (char\_length(fname\_)>=3),

CONSTRAINT fathername\_length check (char\_length(fathername\_)>=3),

CONSTRAINT status\_restriction check (status >=0 and status <2),

CONSTRAINT email\_restriction check (email similar to '([a-z0-9.\_%-]+)@[a-z0-9.\_%-]+\.[a-z]{2,4}'),

CONSTRAINT age\_restrictions check (birth\_date < current\_timestamp - INTERVAL'18 YEAR' and birth\_date > current\_timestamp - INTERVAL'100 YEAR'),

CONSTRAINT phone\_restrictions check (phone\_num similar to '\+38\((050|063|067|093|096)\)+[0-9]{7}'))

Лістинг 5.1 – Створення таблиці employee2\_0

Розглянемо створення таблиці employee2\_0. Створення таблиці відбувається за допомогою команди CREATE TABLE, а потім вказується ім'я таблиці. Кожному полю таблиці зіставляються тип даних обраної СУБД (PostgreSQL) та обмеження цілісності. Поле employee\_id є первинним ключем (PRIMARY KEY) і має тип serial (цей тип є зручним способом створення унікального ідентифікатора шляхом збільшення попереднього значення в даному стовпці на вказане значення за замовчуванням на 1). Поле email є унікальним (unique), за допомогою нього та password\_ реалізується авторизація користувача в додатку. Пароль в базі даних зберігається в HASH-виді за таємним ключем. Це зроблено для додаткового захисту конфіденційної інформації та захисту системи. NOT NULL при визначенні полів таблиці означає, що вони не можуть містити порожні значення. Поле phone\_num також є унікальним, але несе за собою виключно інформаційну функцію для зв’язку з працівниками. CHECK перевіряє чи виконані умови введених значень, наприклад, в полі birth\_date необхідно перевірити, чи є працівнику 18 років. SIMILAR TO дозволяє формувати «шаблон», який у поєднанні з CHECK перевіряє, чи відповідають значення до шаблонних. Наприклад номер телефону має бути відповідним до шаблону +38 – код країни, ((050|063|067|093|096)) – код оператора, та 7 будь-яких цифер після кода опертора.

Інші таблиці в базі даних були створені схожим способом.

**7 ЗАПИТИ ДЛЯ РОЗВ’ЯЗАННЯ ПОСТАВЛЕНИХ ЗАДАЧ**

Далі представлені запити до бази даних, які виконуються у додатку для отримання чи редагування таблиць. Необхідно зазначити, що усі ідентифікатори сховані від користувачів та підставляються системою автоматично. Також перед внесенням змін, перевіряється чи заповнені усі необхідні поля, та чи мають вони коректні дані.

Далі представлені запити до бази даних, які виконуються у додатку для отримання чи редагування таблиць. Необхідно зазначити, що усі ідентифікатори сховані від користувачів та підставляються системою автоматично. Також перед внесенням змін, перевіряється чи заповнені усі необхідні поля, та чи мають вони коректні дані.

Запити для задач адміністратора дитячого садку:

Для задач: А6, А7, А8, А12, А14 використовується запит INSERT INTO table(column1, column2, ...) VALUES (your\_value1, your\_value2*, ...*)

Запити для задач А9,А16,А17,А18 використовується запит DELETE FROM table WHERE id = your\_id

Для задач А1, А2, А3,А4 використовується запит SELECT \* FROM table WHERE id = your\_id.

Для задач А5, А10, А11 використовується запит UPDATE table set column1 = column2, … where column3=column4.

Запити робітників дитячого садку:

Для задач П1,П2,П4,П5 використовується запит SELECT \* FROM table WHERE id = your\_id.

Для задачі П3 використовується запит INSERT INTO table(column1, column2, ...) VALUES (your\_value1, your\_value2*, ...*)

Запити батьків та опікунів дитячого садку:

Опікуни не можуть редагувати ніяку інформацію, лише переглядати її. Це зроблено за для того щоб вся інформація була достовірна і правильно записана. Тому адміністратор перевіряє і сам записує усю інформацію.

Для задач О1,О2,О3,О4 використовується запит SELECT \* FROM table WHERE id = your\_id.

Роздивимося задачі трішки детальніше:

Задачі Адмінстратор:

Задача А1:

Поділяється на два типи, а саме переглянути працюючих працівників та тих що колись працювали.

select \* from employee2\_0 where status = 1 – для працюючих;

select \* from employee2\_0 where status = 0 – для звільнених;

Задача А2:

select employee\_id,lname\_,fname\_,fathername\_,adress,phone\_num,

birth\_date, hiring\_date,contract\_expiration,status,position\_,email from employee2\_0 where employee\_id = '${employee\_id}'

Тут показується вся інформація про працівника окрім його паролю, шукаємо працівника за його унікальним індексом. Список працюючих та звільнених працівників знаходяться на сторінці http://localhost:3000/employee При натисненні на ПІБ працівника, вікдкривається шаблон сторінки в яку записуються данні по індексу працівника ;

Задача А3:

Поділяється на два типи:

-Показати всіх дітей:

select \* from child2\_0 inner join group2\_0 on

child2\_0.fk\_group\_id = group2\_0.group\_id

-Показати дітей які належать до конкретної групи:

select \* from child2\_0 inner join group2\_0 on

child2\_0.fk\_group\_id = group2\_0.group\_id where group\_name = '${group\_name}'

Задача А4:

Подивитися інформацію про конкретну дитину можна абсолютно так же само як і працівника. При натисненні на ПІБ дитини, вікдкривається шаблон сторінки в яку записуються данні по індексу дитини ;

select \* from child2\_0 full outer join group2\_0 on

child2\_0.fk\_group\_id = group2\_0.group\_id where child\_id = '${child\_id}'

Задача А5:

Перевести дитину в іншу групу. Нам потрібен індекс групи та індекс дитини.

update child2\_0 set child\_lname = '${child\_lname}',child\_fname = '${child\_fname}',child\_fathername='${child\_fathername}',receipt\_date='${receipt\_date}',child\_birth\_date='${ '${child\_adress}',medical\_comment='${medical\_comment}',release\_date='${release\_date}',gender='${gender}', fk\_group\_id = '${fk\_group\_id}' where child\_id ='${child\_id} '

Було зроблено редактор в якому адміністратор може замінити будь-яку інформацію про дитину. Існуючі дані автоматично записуються до форми, таким чином можна змінити тільки один параметр і зберегти, замість того щоб заново записувати всю інформацію про дитину.

Задача А6

Створити профіль працівника

Insert into employee2\_0 VALUES (DEFAULT, '${lname\_}', '${fname\_}', '${fathername\_}','${adress}','${phone\_num}','${birth\_date}',' ${hiring\_date}','${contract\_expiration}',DEFAULT,'${position\_}','${email}','${hashPassword}'

Заноситься вся інформація про працівника. Його індекс проставляється автоматично командою DEFAULT також статус робітника назначений в табличці як DEFAULT = 1 адже не можна зареєструвати вже звільненого робітника. Пароль в контроллері хеширується по таємному ключу і записується в базі даних вже в зашифрованому вигляді. Це зроблено для того щоб будь-хто хто має доступ до бази даних не міг бачити паролі всіх користувачів.

Задача А7

Створити нову групу дітей

insert into group2\_0 VALUES(DEFAULT,'${group\_name}','${type\_}','${fk\_employee\_id}')

Адміністратор може сам прописати лише ім’я групи. Її тип можна обрати лише серед запропонованих типів, а індекс працівника також обрати лише з запропонованих працівників зі status = 1 і position\_ = 'Викладач'

select employee\_id , lname\_, fname\_, fathername\_ from employee2\_0 where position\_ = 'Викладач' and status = 1 `

Задача А8

Створити профіль дитини

insert into child2\_0 VALUES (DEFAULT, '${child\_lname}', '${child\_fname}' ,'${child\_fathername}','${child\_birth\_date}','${receipt\_date}','${release\_date}',' ${child\_adress}','${medical\_comment}','${gender}','${fk\_group\_id}')

Адміністратор сам заповнює всі поля окрім поля з групою. Там він може обрати лише запропоновані існуючі групи до якої буде належати дитина після реєстрації.

Задача А9

Видалити групу

Delete from group2\_0 where group\_id = '${group\_id}'

Адміністратор може видалити лише порожню групу. Для цього створено окремий Select для того щоб можна було обирати лише порожні групи.

select \* from group2\_0 full outer join child2\_0 on

child2\_0.fk\_group\_id = group2\_0.group\_id where child\_id is null`

А11

Звільнити працівника

update employee2\_0 set status = 0 where employee\_id = '${employee\_id}'

Змінюється статус працівника по індексу.

А12

Додати предмет

insert into subject2\_0 values(DEFAULT, '${subject\_name}', '${fk\_employee\_id}')

Індекс записується на 1 більше автоматично, назву предмету адміністратор записує у формі, а індекс працівника з запропонованих працівників зі status = 1 і position\_ = 'Викладач'

А13

Створення розкладу

insert into schedule2\_0 values('${day\_week}', '${time\_}', '${fk\_subject\_id}', '${fk\_group\_id}' ,DEFAULT)`

Адміністратор обирає запропонований день, час , предмет і групу чий розклад буде оновлено.

А14 Реєстрація батьків або опікунів

insert into parent2\_0 VALUES (DEFAULT, '${lname\_}', '${fname\_}', '${fathername\_}', '${birth\_date}', '${parent\_type}', '${adress}',' ${work\_place}', '${phone\_num}', '${email}', '${hashPassword}')

Адміністратор сам заповнює всі поля. Пароль також хеширується по таємному ключу.

А15

Подивитися профіль опікуна

При натисненні на ПІБ опікуна відкривається шаблон сторінки в яку підзавантажується потрібна інформація по індексу опікуна.

select parent\_id,lname\_, fname\_, fathername\_, birth\_date, parent\_type ,adress,work\_place ,phone\_num,email від parent2\_0 where parent\_id = '${parent\_id}'

А16

Видалити предмет

delete from subject2\_0 where subject\_name = '${subject\_name}'

Адміністратор видаляє предмет по його імені. Список імен запропонований в модальному вікні

А17

Видалити опікуна

Опікун видаляється по індексу

delete from parent2\_0 where parent\_id = '${parent\_id}`

А18

Випустити дитину

delete from child2\_0 where child\_id = '${child\_id}'

А19

Додати новину

VALUES(DEFAULT,DEFAULT,'${topic}','${text\_}')

Індекс і дата підставляються автоматично

Задачі працівника:

П1

Подивитися список дітей за групами

select \* from child2\_0 inner join group2\_0 on

child2\_0.fk\_group\_id = group2\_0.group\_id where group\_name = '${group\_name}'

Будь який працівник може продивитися всіх дітей які зараз навчаються в дитячому садку.

П2

Подивитися інформацію про дитину

select \* from child2\_0 full outer join group2\_0 on

child2\_0.fk\_group\_id = group2\_0.group\_id where child\_id = '${child\_id}'

Завантажиться шаблонна сторінка з інформацією дитини вибраної за індексом.

П3

Зробити добовий звіт дитини:

insert into dairy2\_0 VALUES (DEFAULT, '${fk\_child\_id}', DEFAULT, '${nutrition}', '${activity}','${gen\_mood}')

Викладач обирає 3 оцінки від 1 до 5 по трьом пунктам: харчування, активність та загальний настрій. Індекс дитини береться з URL адреси.

П4

Подивитися розклад

select \* from schedule2\_0 where fk\_group\_id = '${fk\_group\_id}'

Отримати інформацію про росклад вибраної групи дітей

П5

Подивитися профіль робітника

При натисненні на ПІБ опікуна відкривається шаблон сторінки в яку підзавантажується потрібна інформація по індексу опікуна.

Задачі опікуна:

Всі є однотипними реалізуються через Select з підстановкою вірних таблиць і значень

О1: select \* from schedule2\_0 where fk\_group\_id = '${fk\_group\_id}'

О2: select dairy\_num,date\_,nutrition,activity,gen\_mood from dairy2\_0 where fk\_child\_id = '${fk\_child\_id}' and date\_ = current\_date `

О4: SELECT fk\_child\_id ,avg(gen\_mood)::numeric(10,2) as avg\_mood , avg(activity)::numeric(10,2) as avg\_activity, avg(Nutrition)::numeric(10,2)

as avg\_nutrition FROM dairy2\_0 where fk\_child\_id = '${fk\_child\_id}'

**8 ПРОГРАМНА РЕАЛІЗАЦІЯ ІНТЕРФЕЙСУ**

Кожна модель реагує на команди контролерів в яких йде звернення до цієї моделі. Так, як кожен користувач системи має свій окремий простір, контролери поділено між ролями. Список контролерів:

-childApi – забезпечує зв’язок з childController, де отримується інформація про дітей для працівників та користувачів.

-employeeApi - забезпечує зв’язок з employeeController, де отримується інформація про працівників для працівників та користувачів.

-parentApi - забезпечує зв’язок з parentController, де отримується інформація про опікунів для працівників та користувачів.

-scheduleApi – забезпечує зв’язок з scheduleController, де отримується інформація для всіх користувачів про розклад

-userApi – забезпечує зв’язок з employeeController та parentController, а саме відповідає за реєстрацію, авторизацію та перевірку JWT ( JSON WEB TOKEN) користувачів.

-adminApi – заберзпечує зв’язок з рядом контролерів для адмін панелі.

Таблиця 8.1 – Структура ПО

|  |  |  |
| --- | --- | --- |
| View | Controller | Model |
| FrontPage | userApi | employeeController,  parentController |
| childPanel | childApi | childController,  groupController |
| employeePanel | employeeApi | employeeController,  dairyController |
| parentPanel | parentApi | parentController |
| schedule | scheduleApi | scheduleController |
| adminPanel | adminApi | employeeController,  parentController,  childController,  groupController,  scheduleController,  subjectController,  dairyController |

**9 БЕЗПЕКА КЛІЄНТА ТА ІНФОРМАЦІЙНОЇ СИСТЕМИ**

Безпека БД забезпечується шляхом створення ролей. Ролі записані в представлені в ІС

Код створення представлення:

CREATE OR REPLACE VIEW authinfo

AS

SELECT employee2\_0.employee\_id,

employee2\_0.email,

'1' AS "Roles",

employee2\_0.password\_,

employee2\_0.position\_ AS type,

employee2\_0.phone\_num

FROM employee2\_0

WHERE employee2\_0.position\_ = 'Адміністратор' AND employee2\_0.status = 1

UNION

SELECT employee2\_0.employee\_id,

employee2\_0.email,

'2' AS "Roles",

employee2\_0.password\_,

employee2\_0.position\_ AS type,

employee2\_0.phone\_num

FROM employee2\_0

WHERE employee2\_0.position\_ = 'Викладач' AND employee2\_0.status = 1

UNION

SELECT parent2\_0.parent\_id AS employee\_id,

parent2\_0.email,

'3' AS "Roles",

parent2\_0.password\_,

parent2\_0.parent\_type AS type,

parent2\_0.phone\_num

FROM parent2\_0;

Таким чином ми отримуємо інформацію про всіх активних користувачів. Звільнені працівники не зможуть авторизуватися, адже в представлення записуються дані лише тих працівників які мають статус 1, тобто активний.

При авторизація користувача в його JWT записуються наступні дані:

Роль: вона може бути 1,2 або 3.

Якщо роль = 1 то авторизувався адміністратор. На рівні клієнта йому буде відображена адмін панель. Іншим ролям адмін панель буде схована, також доступ до адмін панелі заборонений. Навіть якщо користувач вручну введе в URL потрібний адрес, він не зможе перейти на сторінку.

Якщо роль = 2 то авторизувався працівник. Він може передивлятися будь-яку інформацію. А також записувати дітям раз в день добовий звіт.

Якщо роль =3 то авторизувався опікун. Він може лише передивлятись інформацію про дітей, розклад, і новини.

const generateJwt = (Role, EMAIL, position\_) => {

return jwt.sign(

{ Role, EMAIL, position\_ },

process.env.SECRET\_KEY,

{ expiresIn: '24h' }

)

}

Код запису даних в JWT.

Токен дійсний 24 години після, цього часу користувачу потрібно знову авторизуватися. Також дані шифруються по таємному ключу який знаходься в процесних змінних.

Інший рівень захисту знаходиться на серверному рівні.

router.post('/post',checkRole('Адміністратор'),groupController.post)

Перед кожним запросом до функції контроллеру провіряється тип користувача який також закодований в JWT

const jwt = require('jsonwebtoken')

module.exports = function(position\_){

return function (req, res, next) {

if (req.method === "OPTIONS") {

next()

}

try {

const token = req.headers.authorization.split(' ')[1] // Bearer asfasnfkajsfnjk

if (!token) {

return res.status(401).json({ message: "The user is not authorized" })

}

const decoded = jwt.verify(token, process.env.SECRET\_KEY)

if(decoded.position\_ == 'Administrator' || decoded.position\_ == 'Teacher' ){

req.user = decoded

}

else {

return res.status(403).json({message:'No access'})

}

next()

} catch (e) {

res.status(401).json({ message: "The user is not authorized" })

}

};

}

Лістинг коду файла role-middleware.js.

Він насамперед провіряє наявність токену в локальному сховищі. Якщо Token = null то на клієнт повертається код похибки 401 (Не Авторизований)



Рисунок 10.1- приклад не авторизованого користувача

Якщо ж токен є в сховищі. То він по секретному слову розкодовує його і провіряє поле position\_. Якщо воно містить слово Адміністратор, Викладач то запрос пропускається на сервер. Якщо ні то повертає код похибки 403(Немає доступу)



Рисунок 10.2- приклад на недостатні права доступа

**10 ІНСТРУКЦІЯ КОРИСТУВАЧА**

В даному розділі розглядається розв’язання користувацьких задач за допомогою розробленого інтерфейсу.

## Батьки або опікуни

Після підключення до веб-адреси додатку, користувач бачить головну сторінку (рисунки 10.1 та 10.2).



Рисунок 10.1 – Початкова сторінка

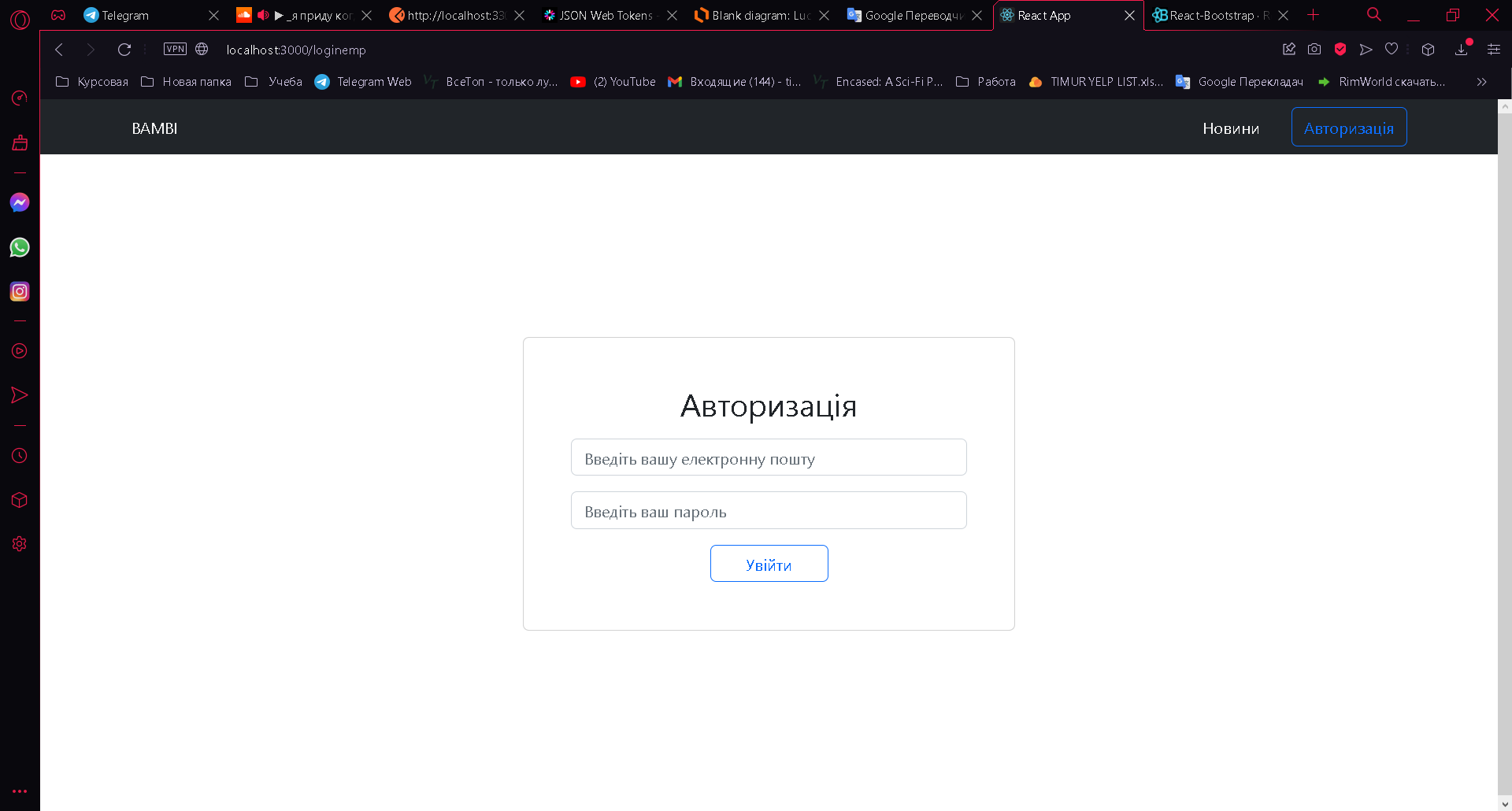


Рисунок 10.2 – Вікно авторизації

Користувач може увести свої логін та пароль, якщо він має аккаунт. Якщо ні то людина яка хоче записати свою дитину в дитячий сад повинна перейти на сторінку Новини і зв’язатися з дитячим садком.

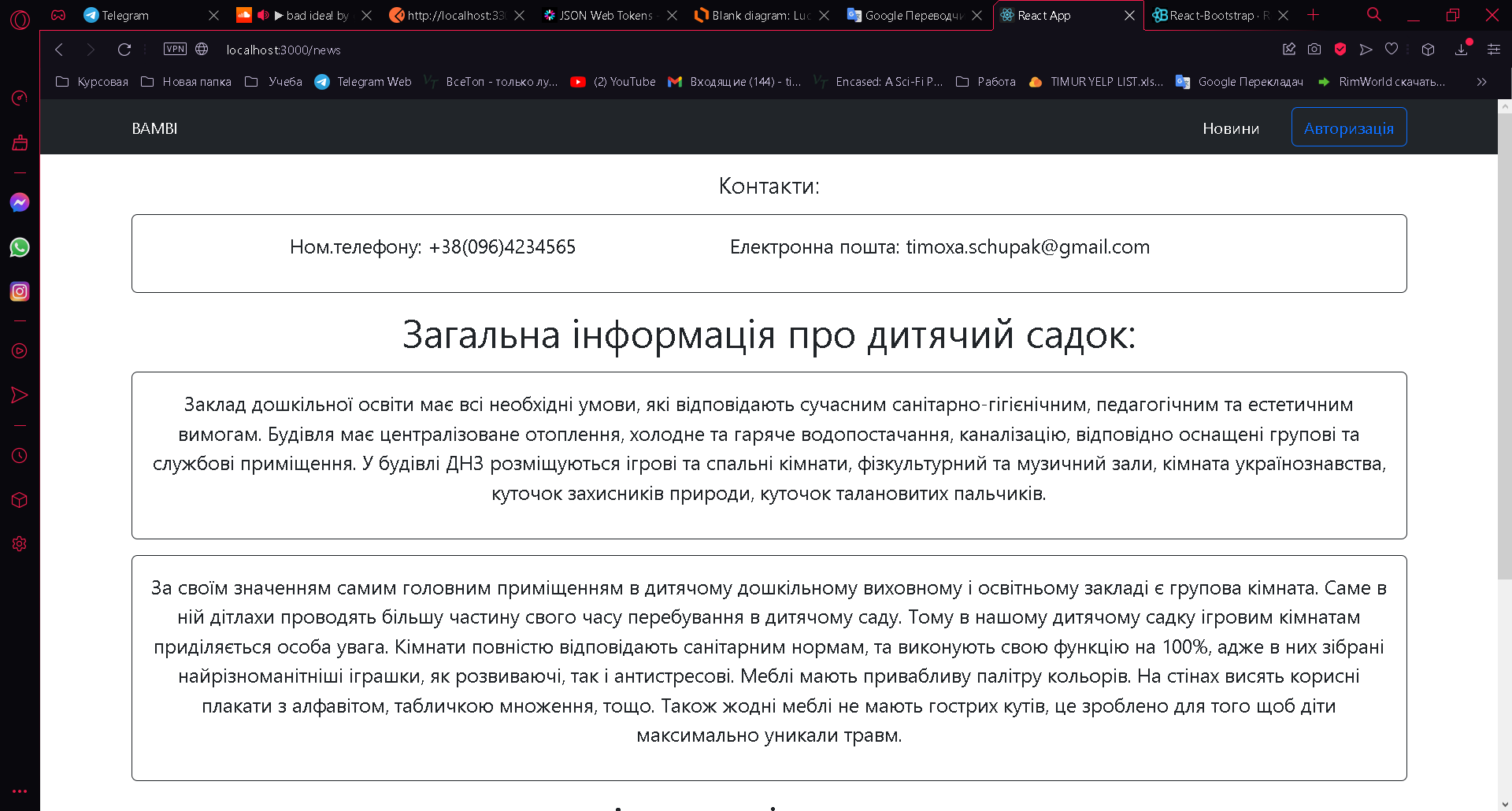


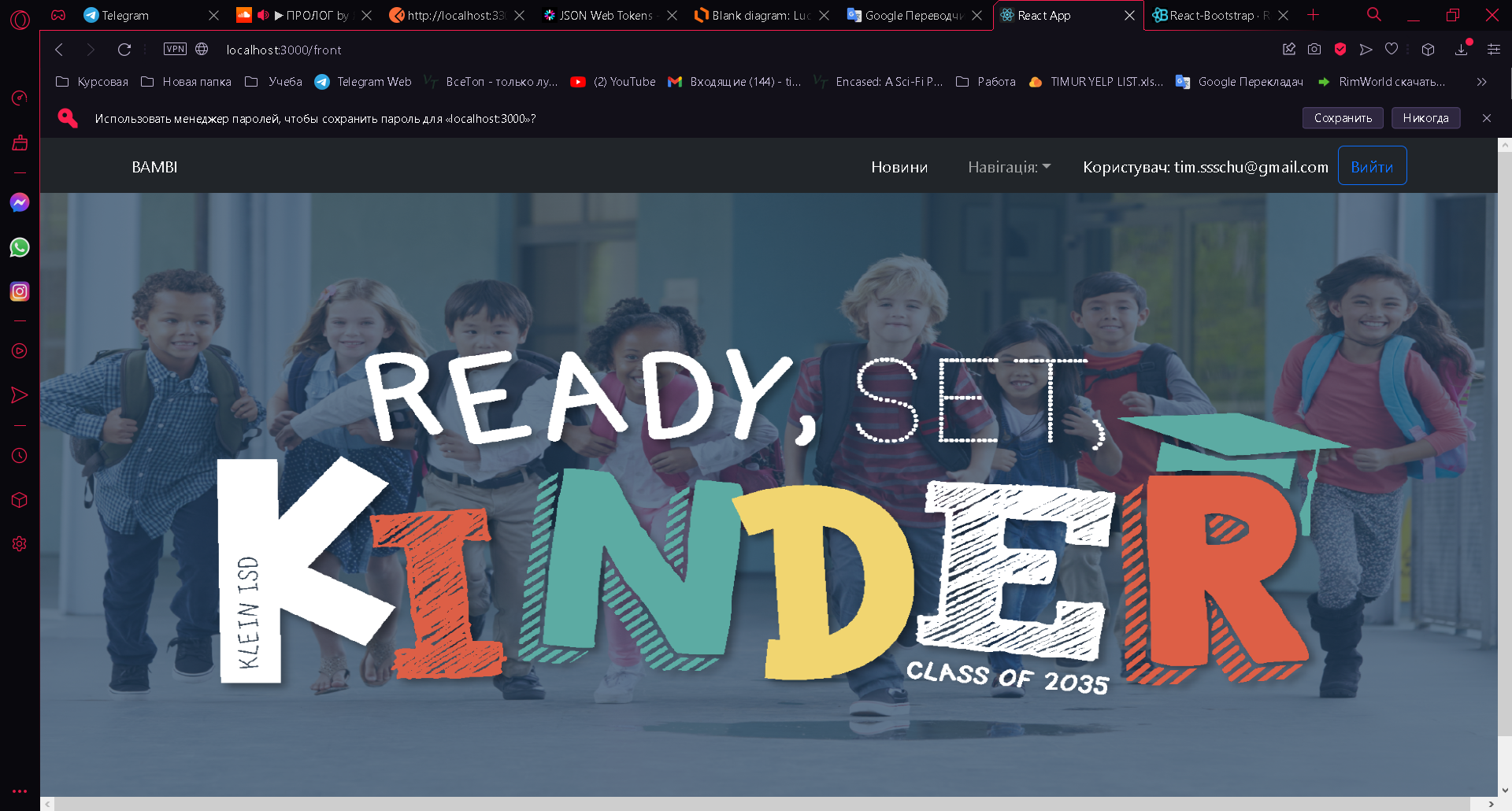
Рисунок 10.3 – Вікно новин та загальної інформації 

Рисунок 10.3 – Головна сторінка авторизованого користувача.

З’являється підпис Користувач: ЕMAIL користувача. Це означає що користувач авторизований. Також замість кнопки авторизації, з’явилась кнопка Вийти. Вона виконує ініціацію виходу з аккаунту.

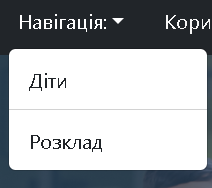


Рисунок 10.4 - Вкладка навігація

Показує куди користувач може перейти. Навігація змінюється в залежності від ролі. Наприклад Ролі 3 доступно дві сторінки:

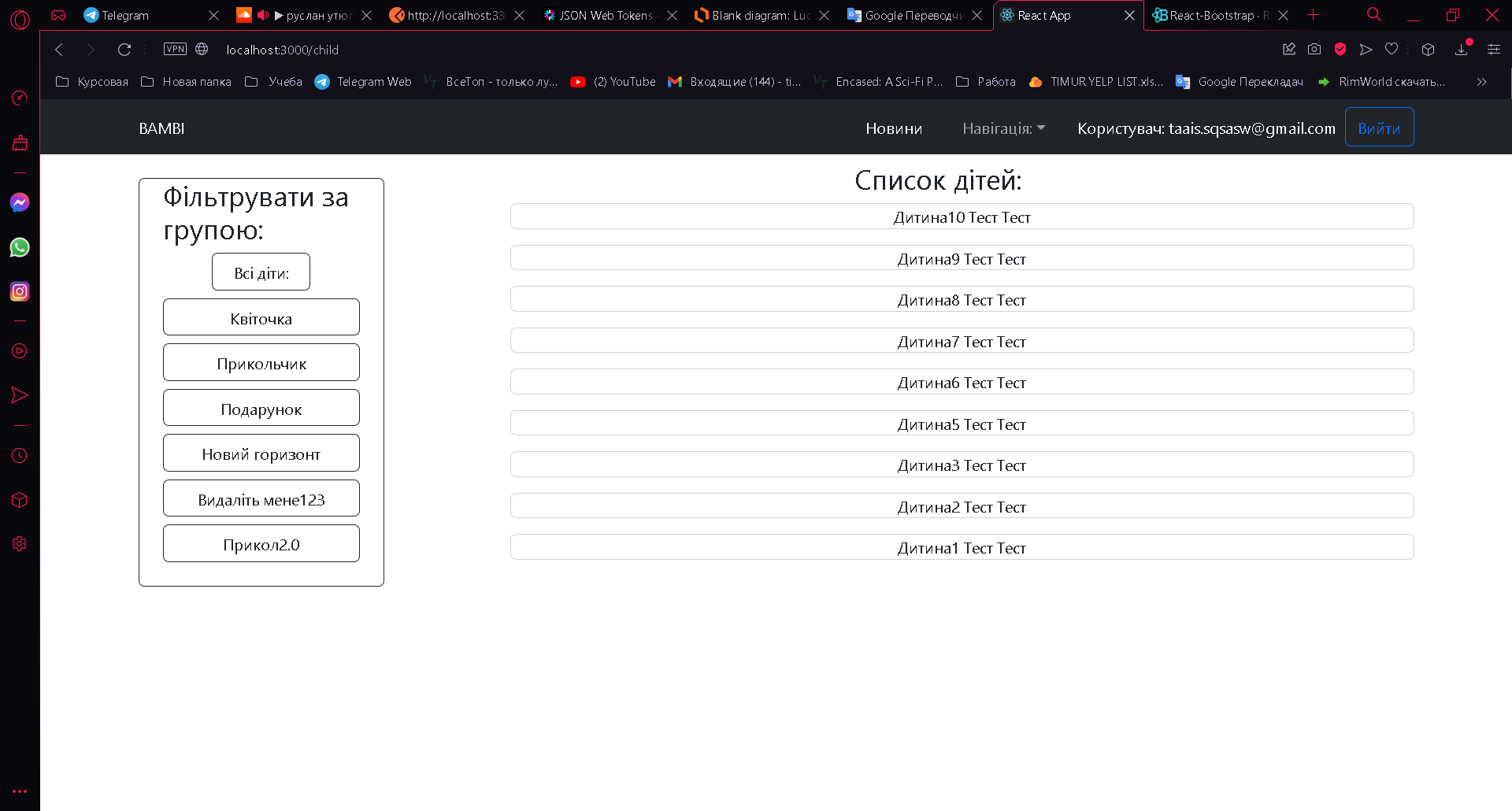


Рисунок 10.4 - Сторінка Діти

На цій сторінці спочатку будуть показані усі діти, їх можна сортувати за групами, клікаючи на назву групу зліва.

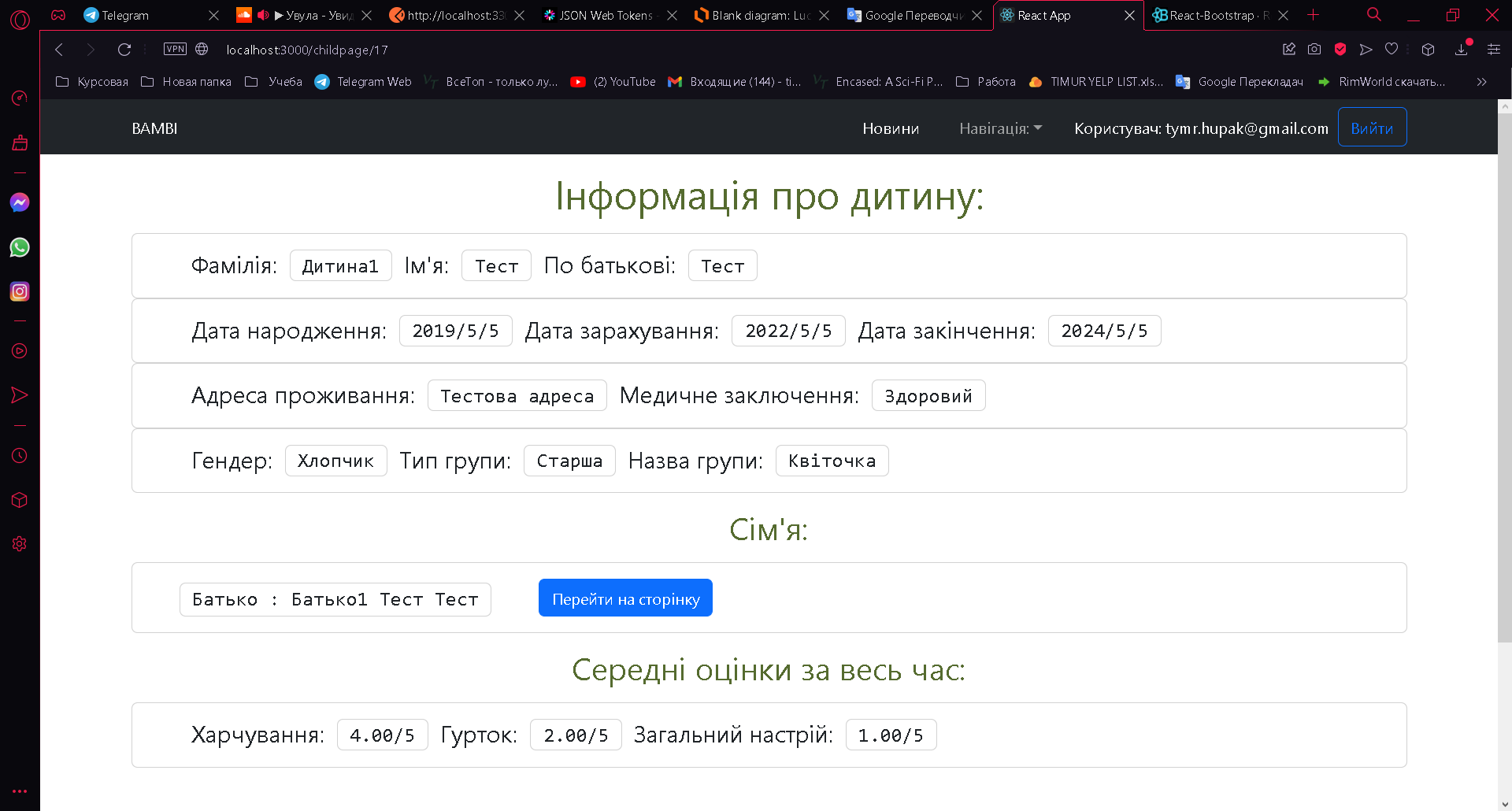


Рисунок 10.5 - Сторінка Інформації про дитину

На цій сторінці є повна інформація про дитину,її сім’ю, середні оцінки дитини за весь час. А також щоденник.

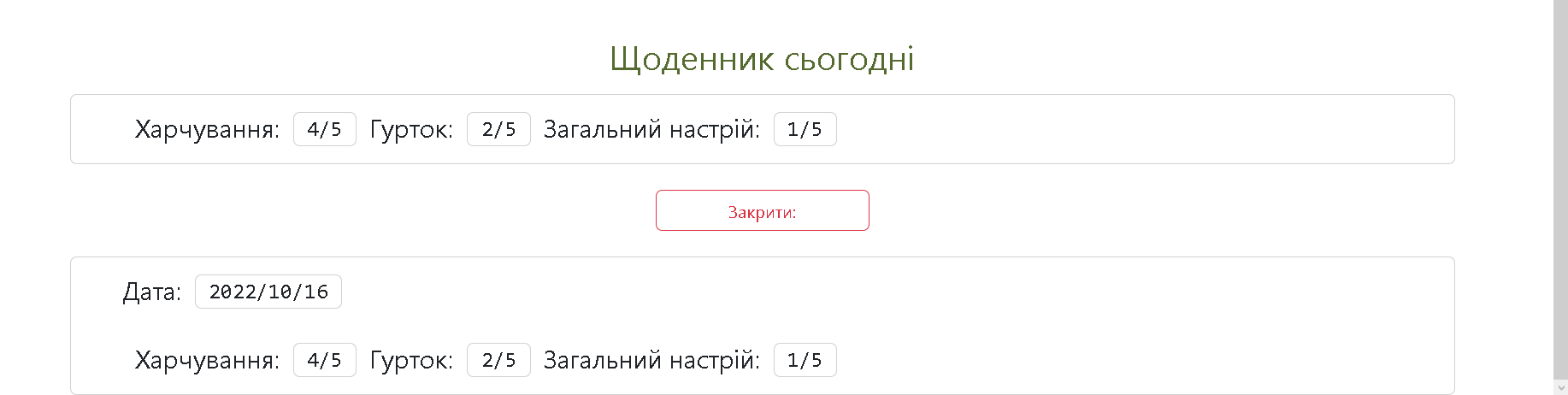


Рисунок 10.6 – Щоденник дитини

Вище буде показано Щоденник сьогодні, але лише в тому випадку, якщо викладач уже зробив запис в щоденник сьогодні. Зазвичай після 4 годин всі щоденники повинні бути виставлені.

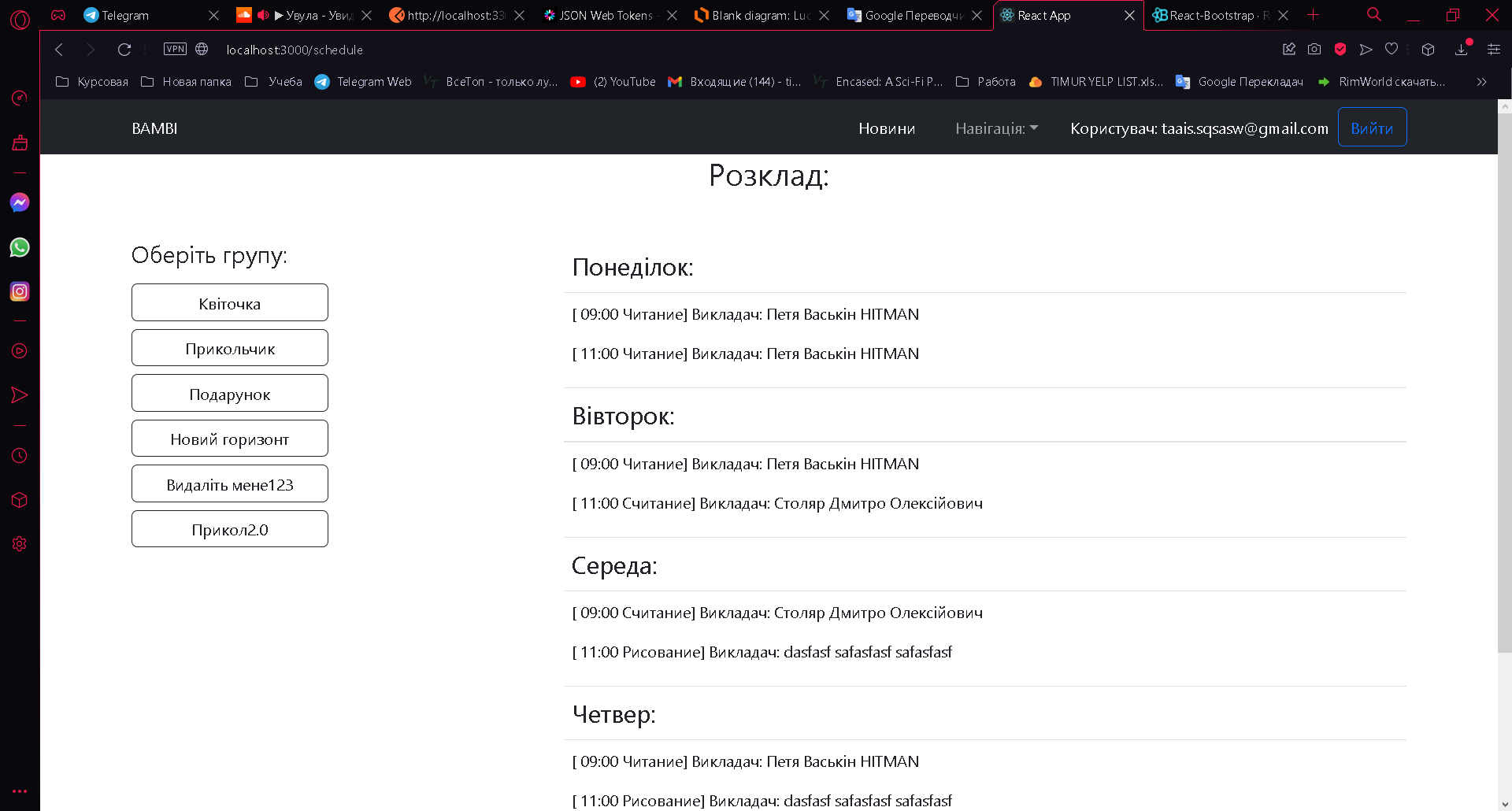


Рисунок 10.7 - Сторінка Розклад

Тут в формі таблиці показані дані з розкладу по групам. Групи розташовані зліва розкладу.

На цьому функціонал опікуна закінчується.

Викладачі

Функціонал викладачів мало чим відрізняється від функціоналу опікунів. Але є одна велика відмінність. Працівники можуть робити добовий звіт один раз на добу для дитини.

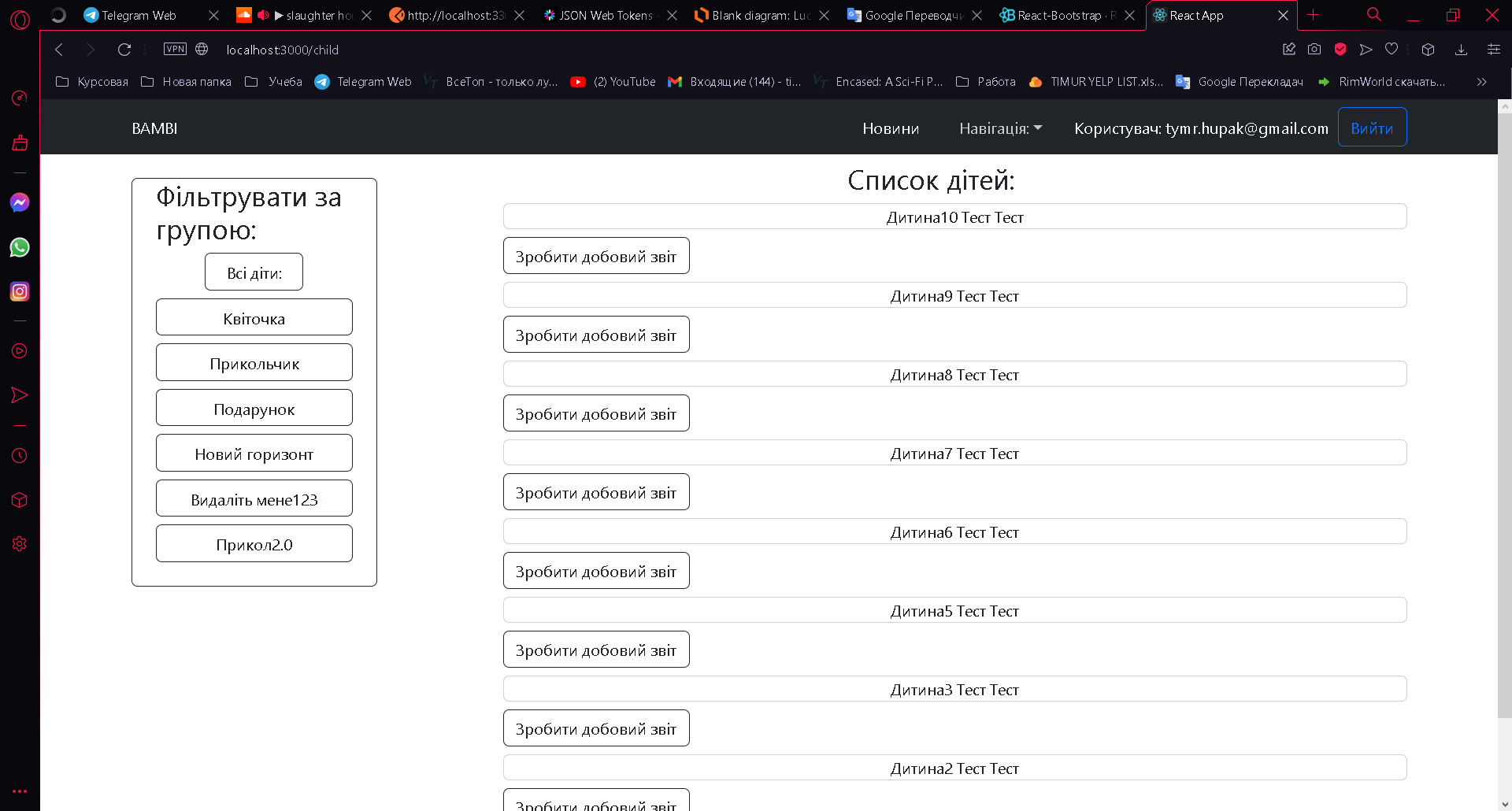


Рисунок 10.8- Сторінка дітей для викладачів

Це одна й та ж сама сторінка що й для опікунів, але є один виняток. Під кожною дитиною є кнопка «Зробити добовий звіт». Якщо її натиснути то ми перейдемо на сторінку на якій можемо оцінити: активність, харчування, загальний настрій дитини.

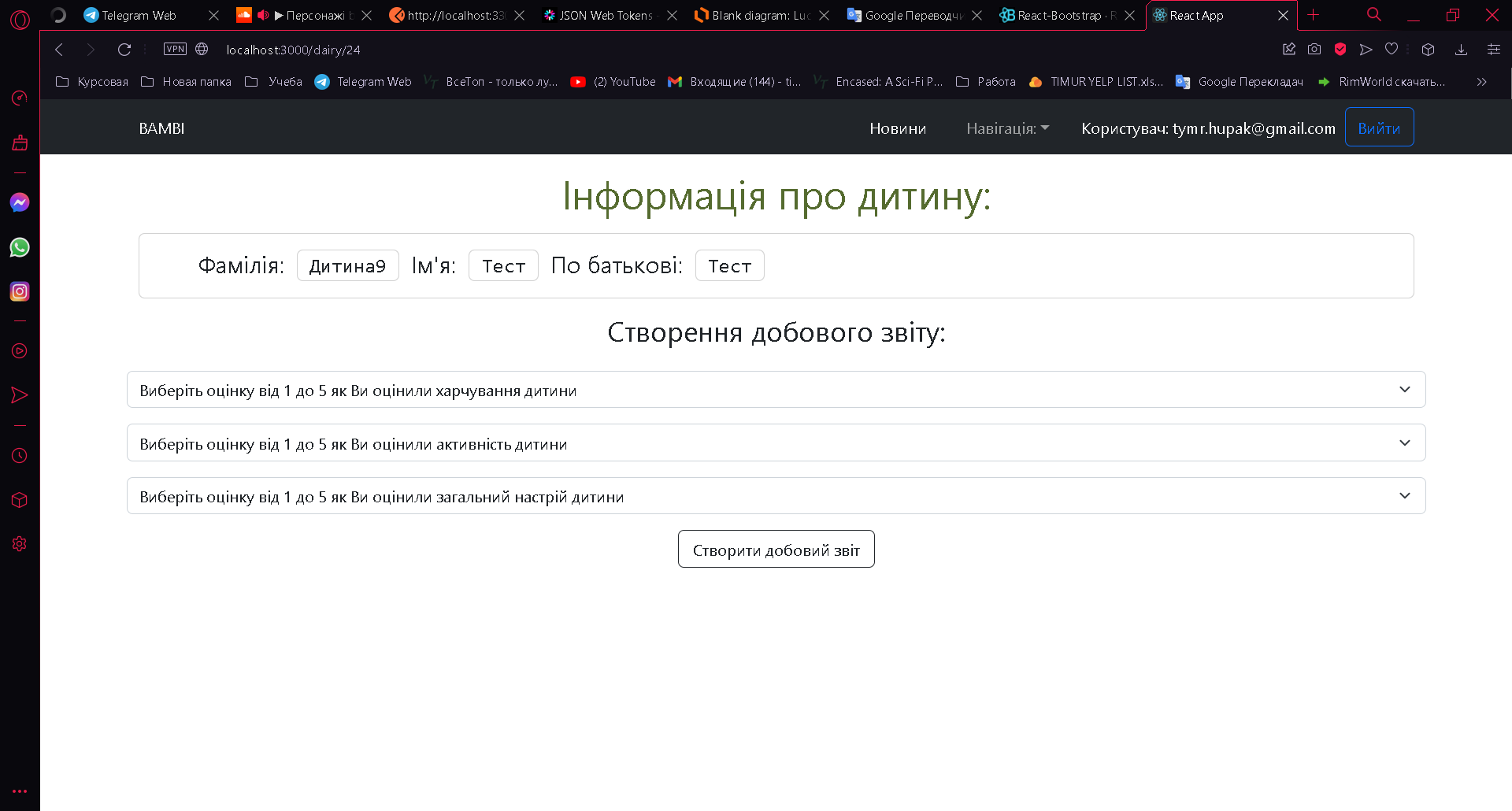


Рисунок 10.8- Сторінка щоденника для дитини.

Якщо добовий звіт дитини на цю добу уже створено то ця сторінка має інший вид, а саме:

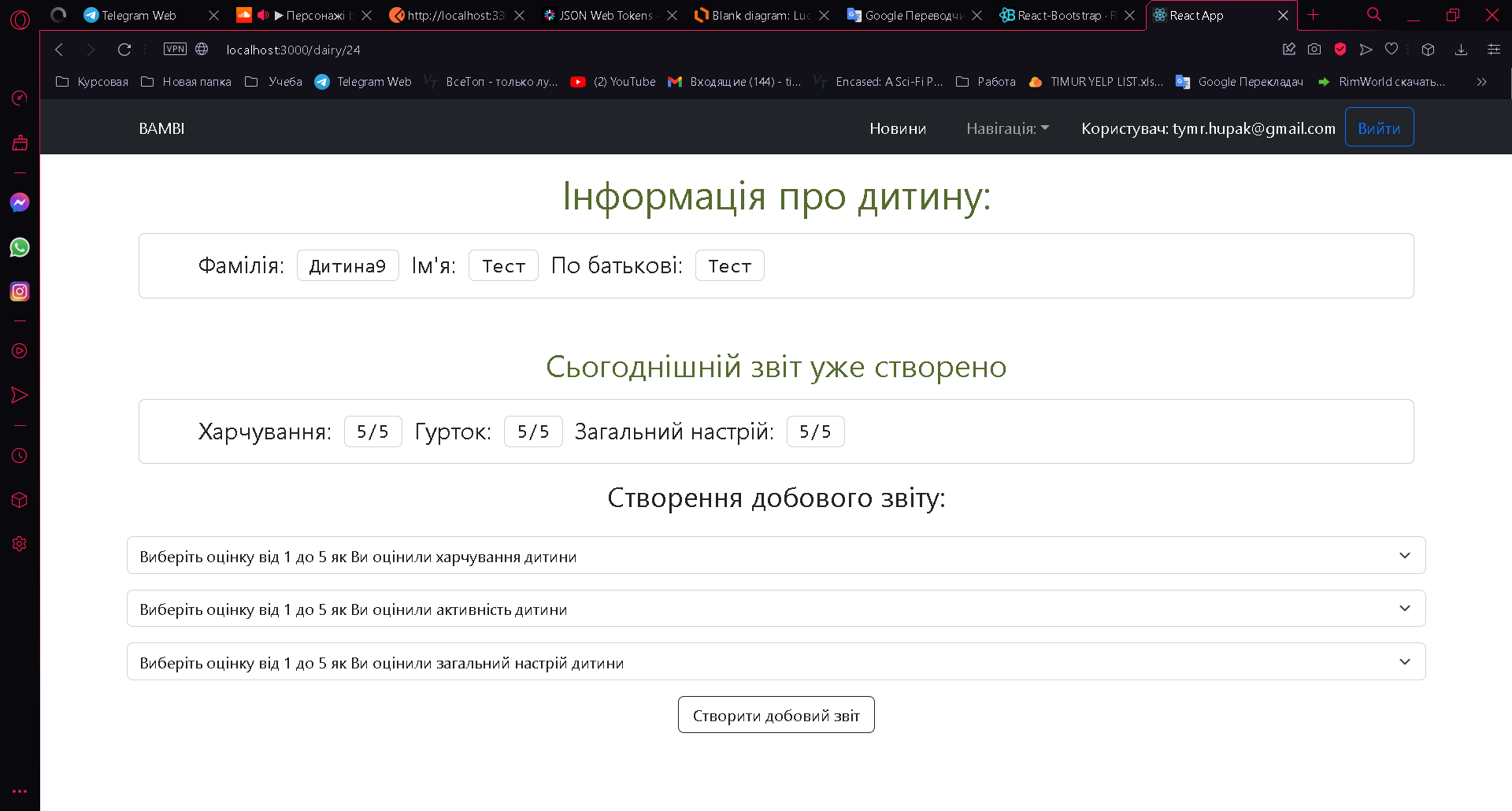


Рисунок 10.9- Сторінка щоденника для дитини, у якої уже є сьогоднішній звіт.

З’являється інформація про те що звіт уже заповнений і є інформація які оцінки отримала дитина.

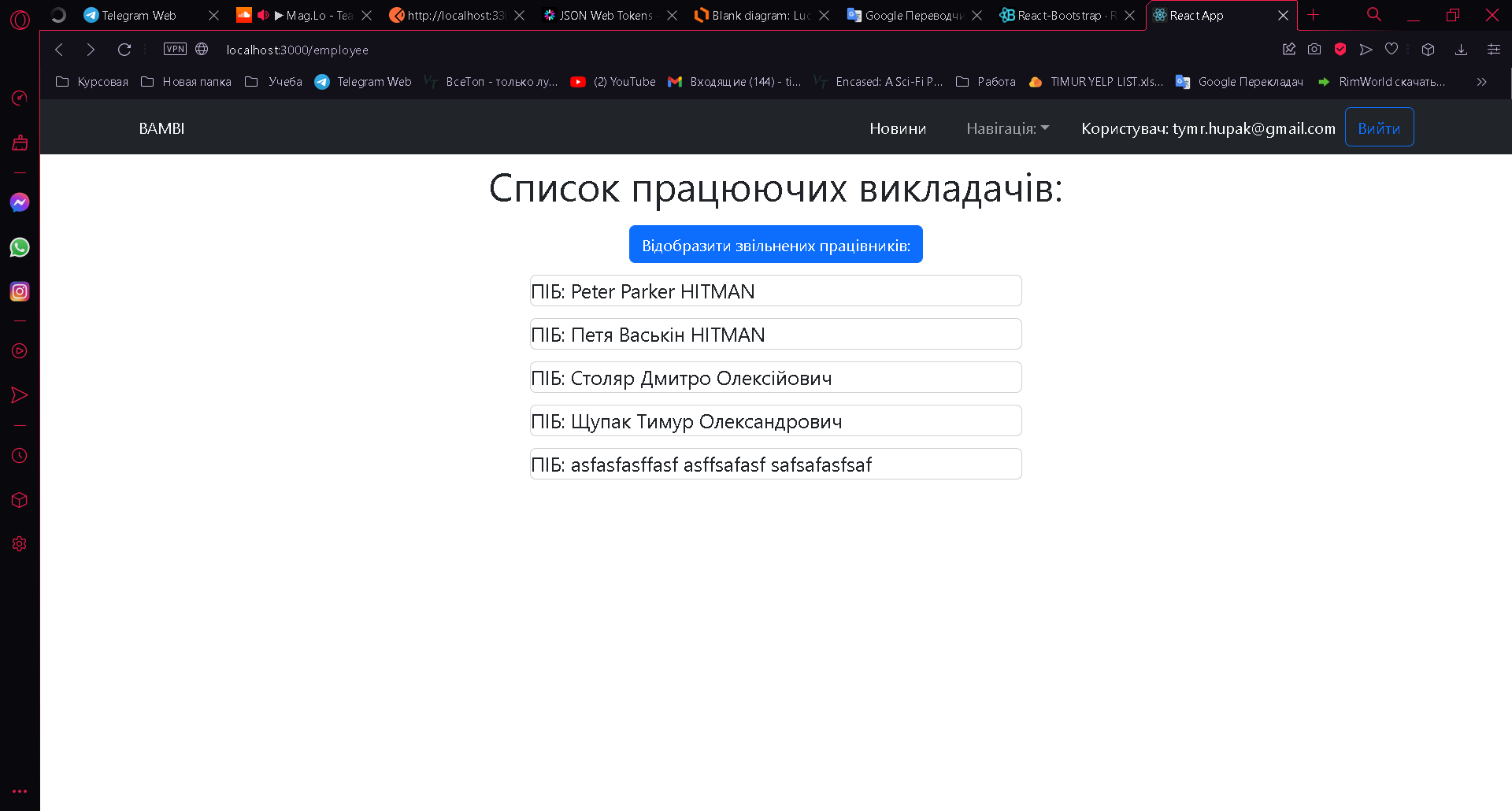


Рисунок 10.9- Сторінка зі списком працівників

При відкриті сторінки по стандарту буде список працюючих працівників, але якщо є необхідність можна відобразити звільнених працівників. Це можна зробити натиснувши кнопку «Відобразити звільнених працівників»

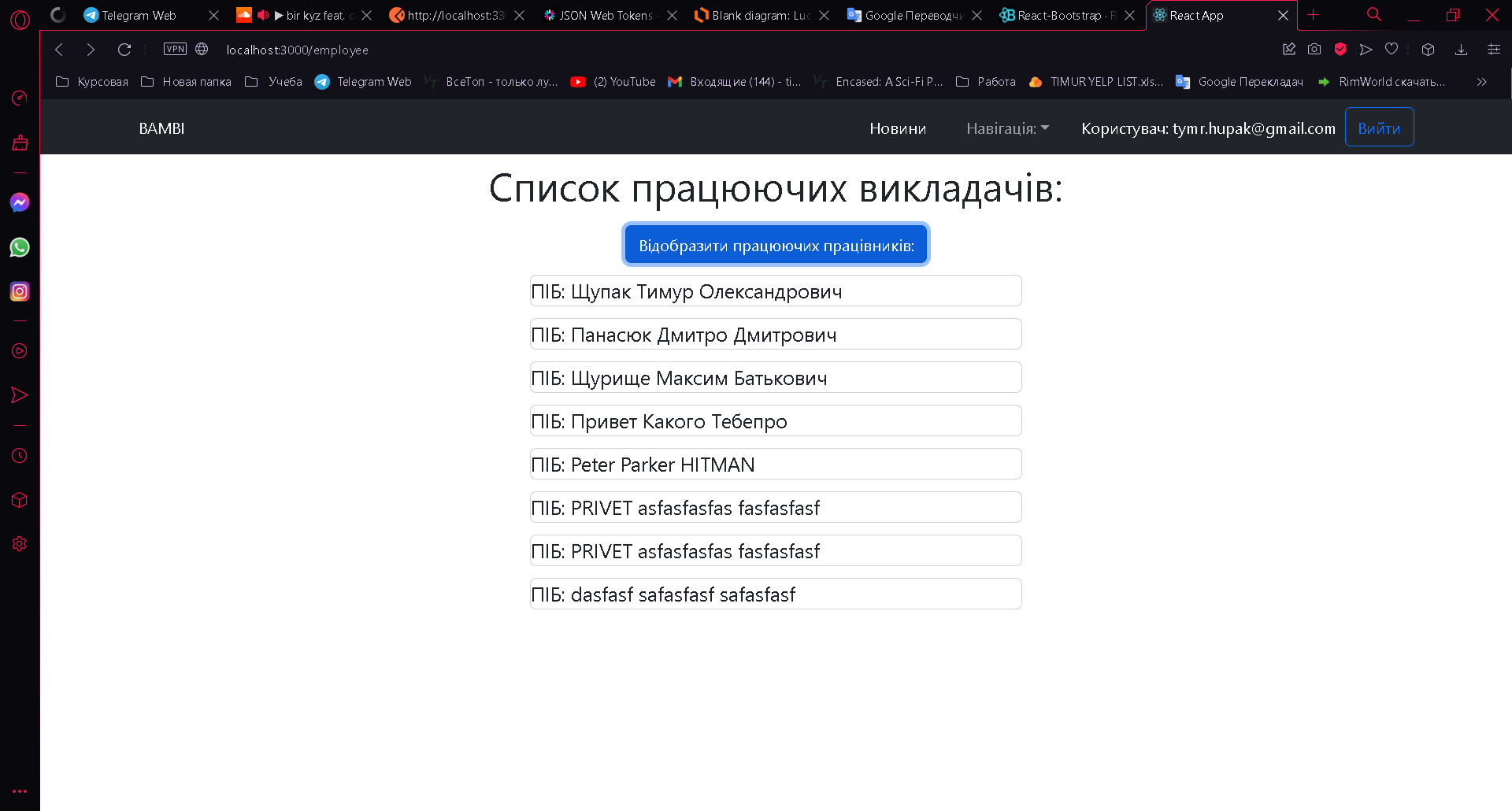


Рисунок 10.9- Сторінка зі списком звільнених працівників

Список замінюється звільненими працівниками і також змінюється кнопка на «Відобразити працюючих працівників»

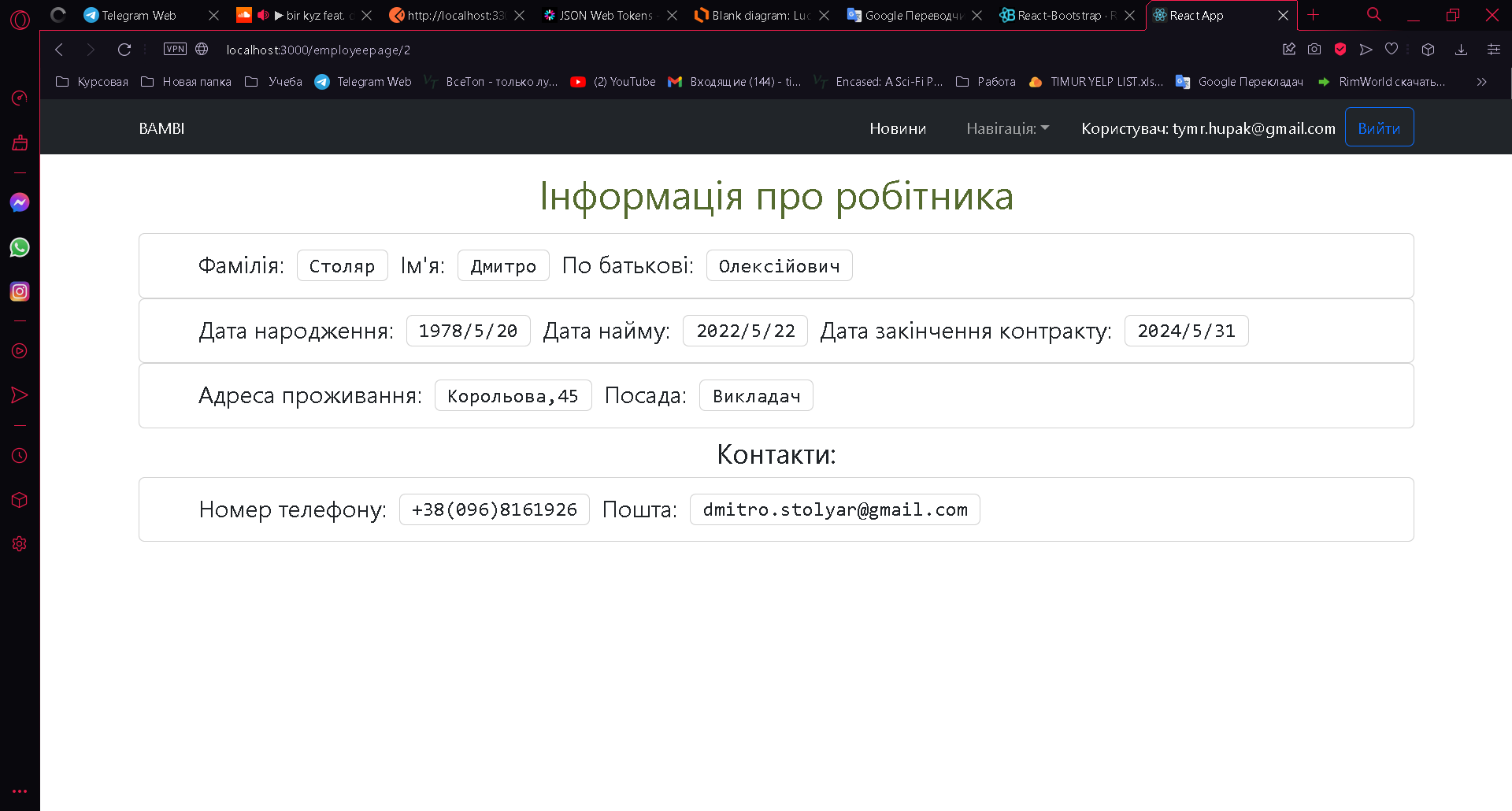


Рисунок 10.10- Сторінка з повною інформацією працівника

Адміністратор

Адміністратор має доступ до всіх сторінок, а також на Навігаціоній панелі з’являється кнопка адмін панель. Натиснувши її ми переходимо на сторінку з усім функціоналом адміністратора. Категорії функціоналу поділені на категорії:

-реєстрація: працівника, опікуна, дитини а також зв’язати дитину і опікуна;

-взаємодія з групами: додати групу, видалити групу;

-редагування або видалення: опікуни, працівники, діти;

-предмети та розклад: додати предмет, редагувати або видалити предмет, додати або видалити розклад

-новини: додати новину (Адміністратор видаляє новини на сторінці «Новини». Там тільки для нього з’являється кнопка «Видалити» біля кожної новини. Натискаючи на неї, запис новини видаляється з ІС).

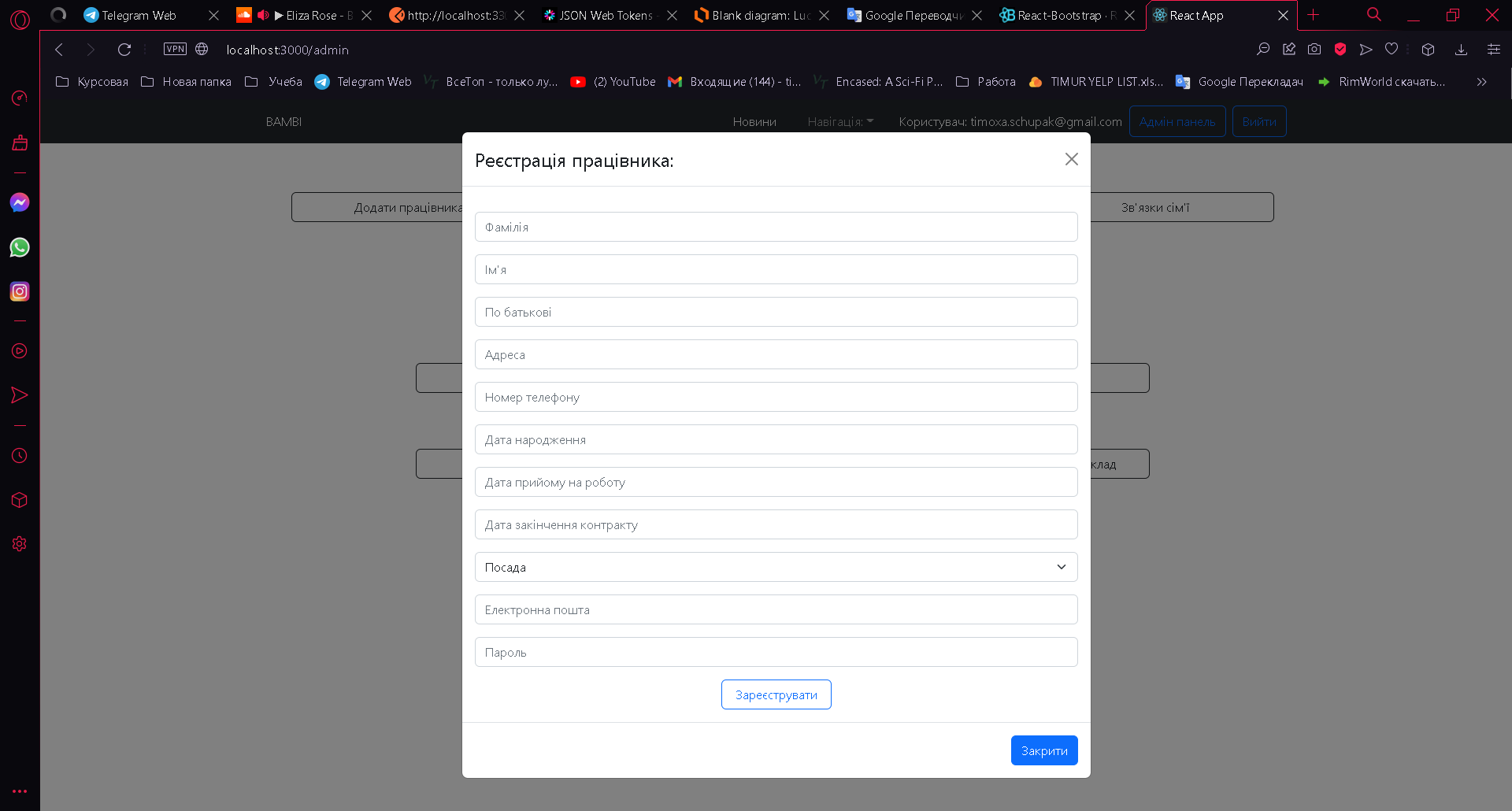


Рисунок 10.11- Реєстрація працівника

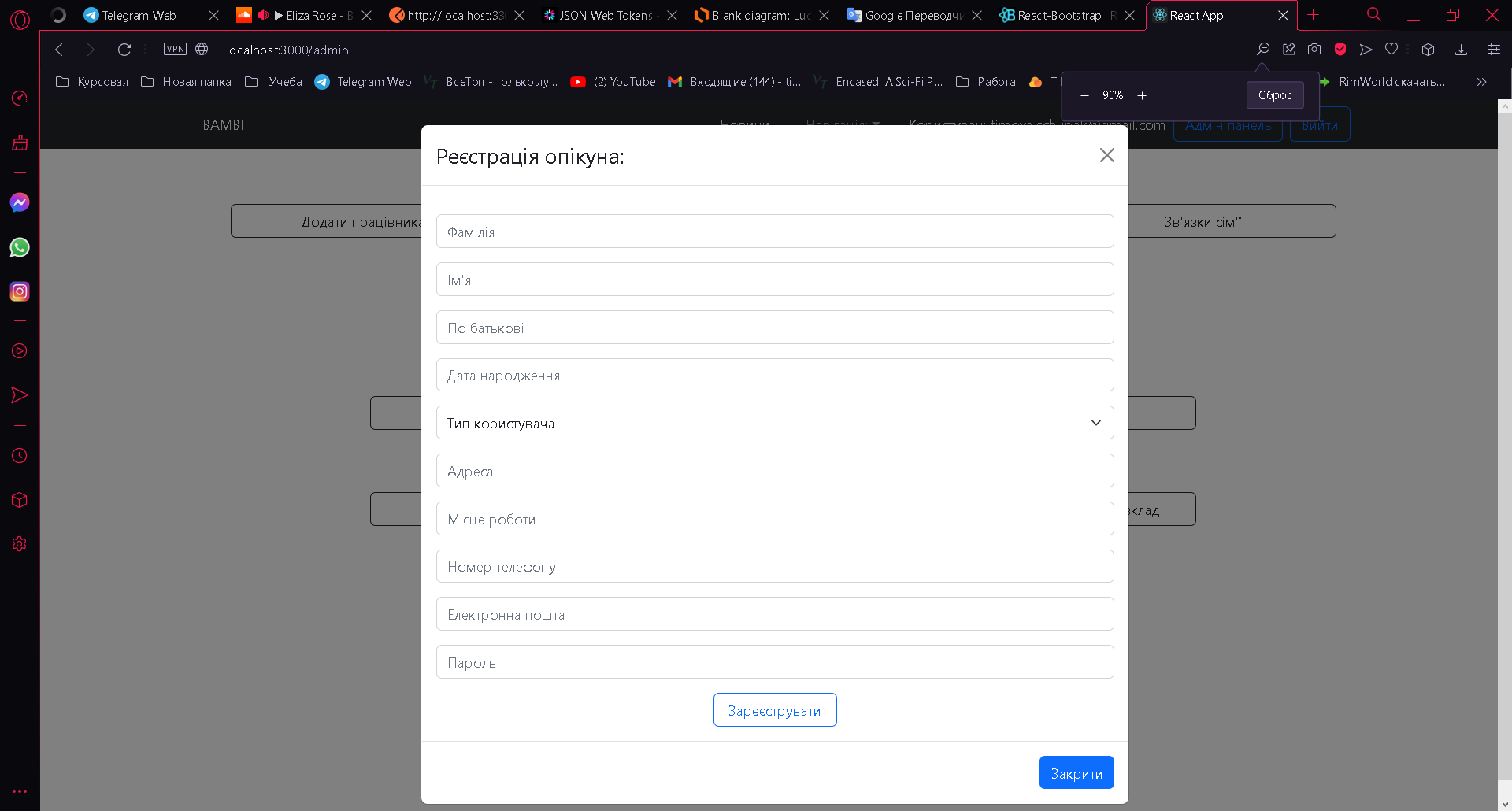


Рисунок 10.12- Реєстрація опікуна

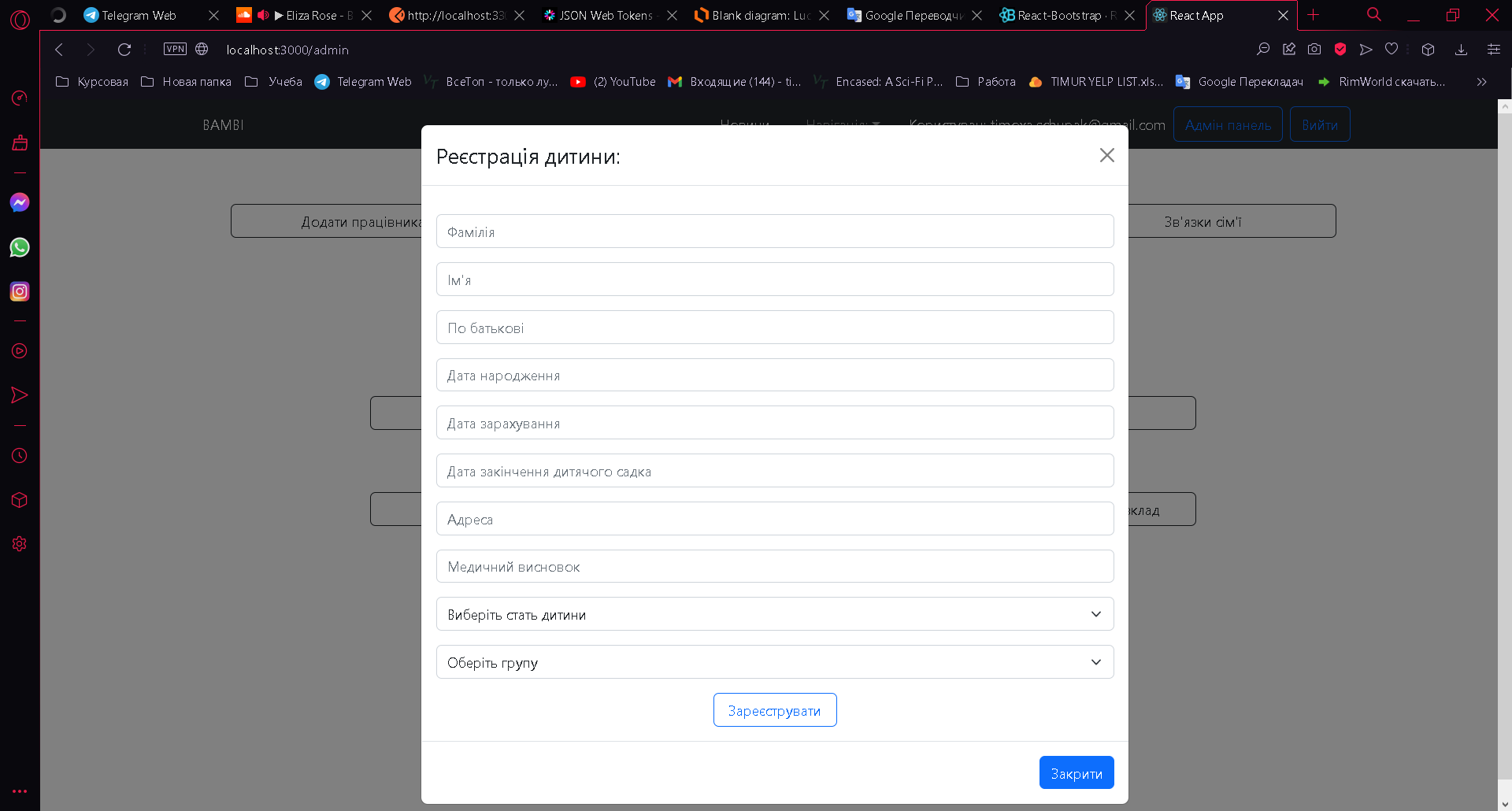


Рисунок 10.13- Реєстрація дитини

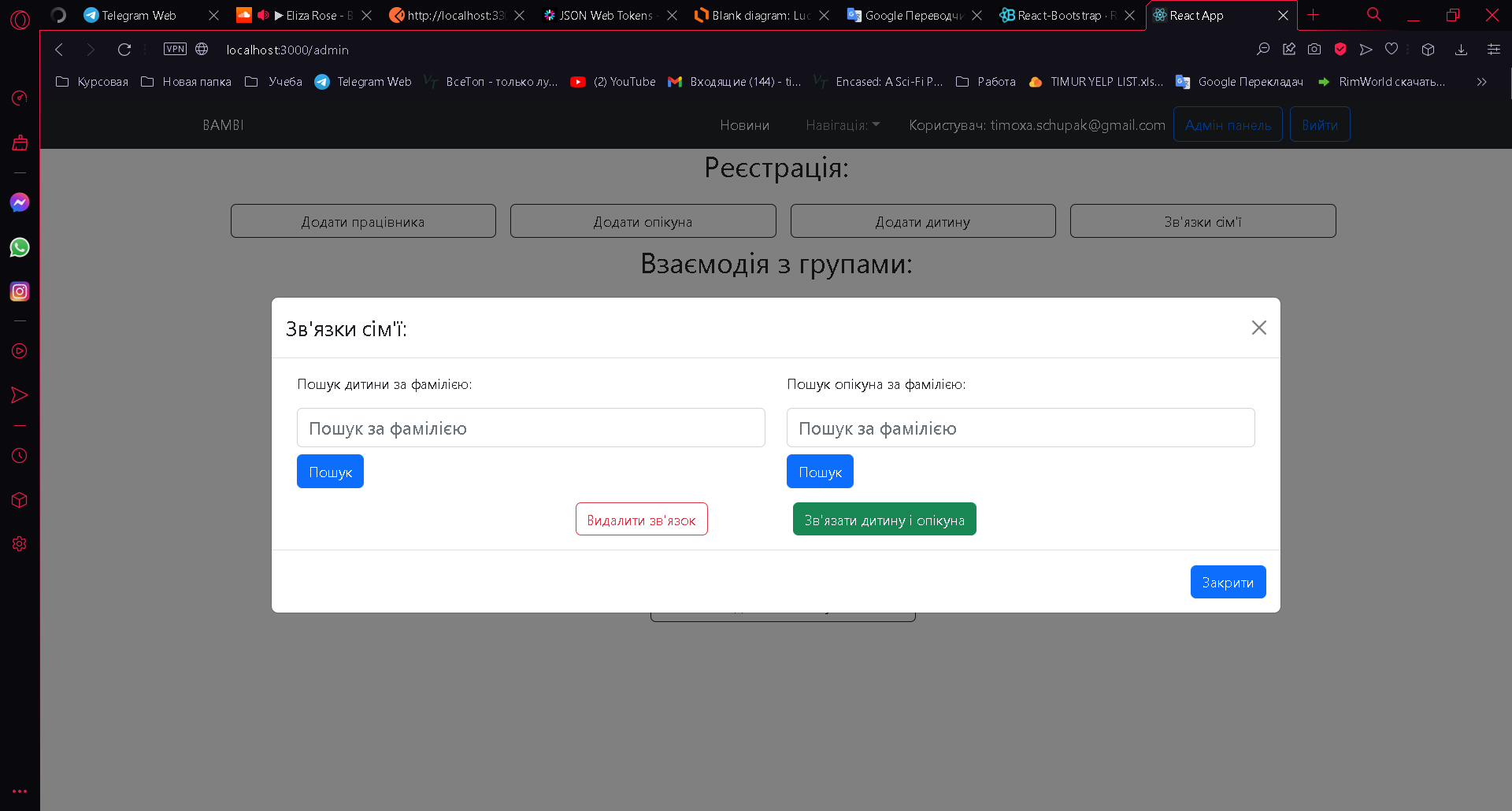


Рисунок 10.14- Додати сім’ю

Тут реалізований пошук за фамілією як дітей так і опікунів.

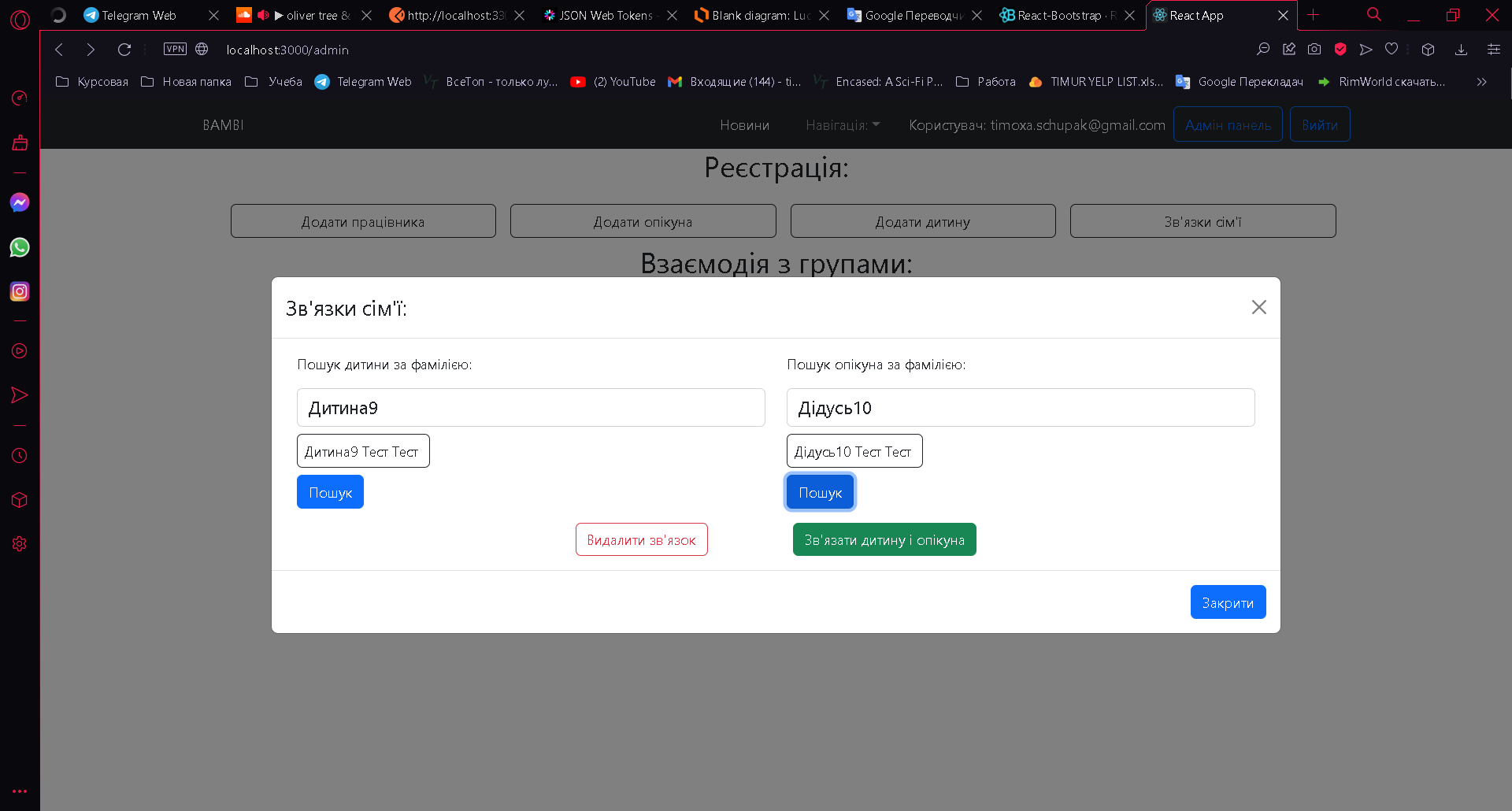


Рисунок 10.15- Додати сім’ю

При натиснені на дитину чи опікуна, у випадку якщо у них уже існує з кимось зв’язок, він видобразиться нижче

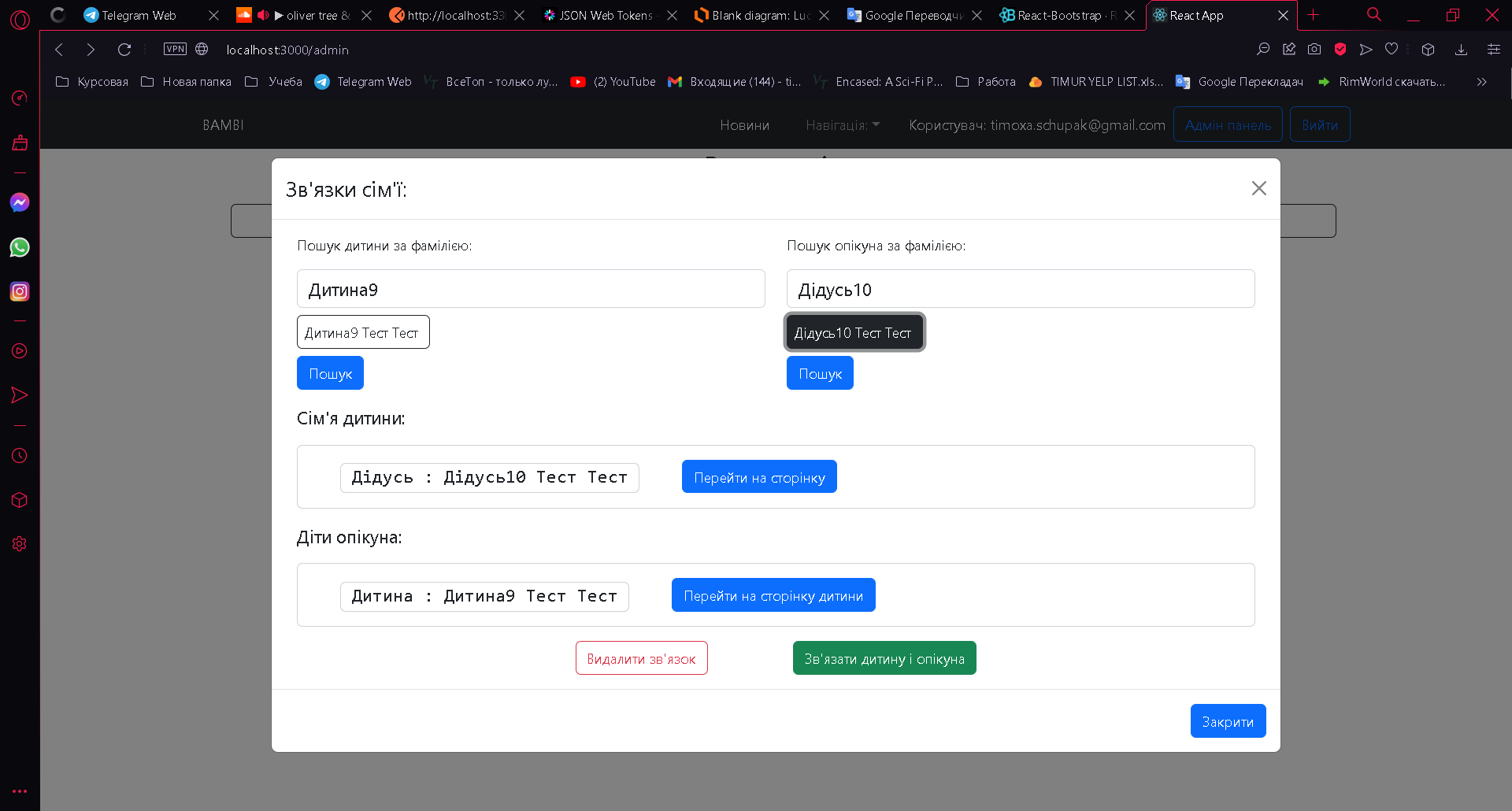


Рисунок 10.16- Додати сім’ю

Тут ми можемо побачити що «Дитина9» та «Дідусь10» є родичами, ми можемо видалити цей зв’язок натиснувши кнопку «Видалити зв’язок»

Обавши дитину та опікуна які не зв’язані за натиснувши кнопку «Зв’язати дитину і опікуна» ми створимо їх зв’язок.

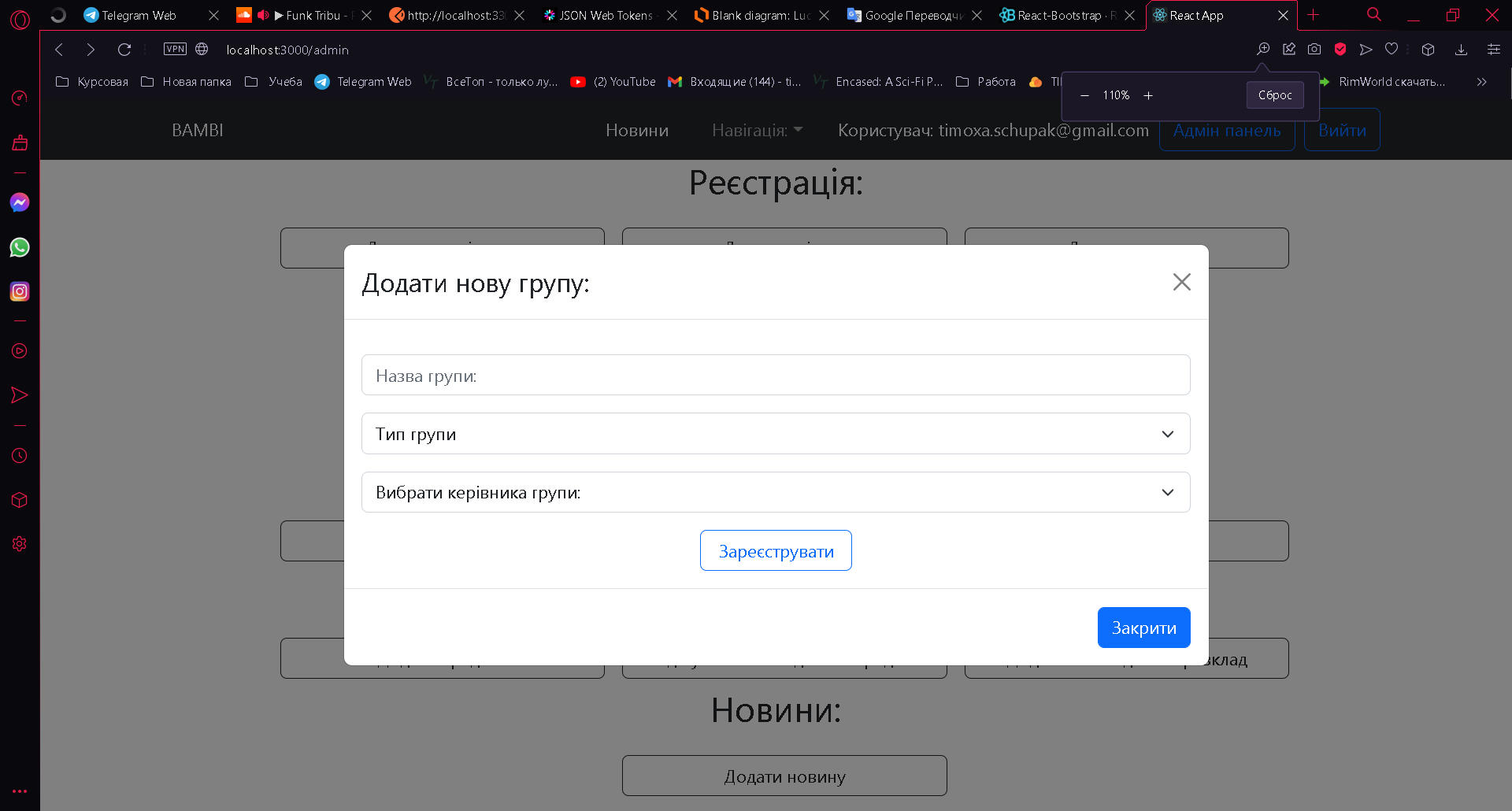


Рисунок 10.17- Додати групу

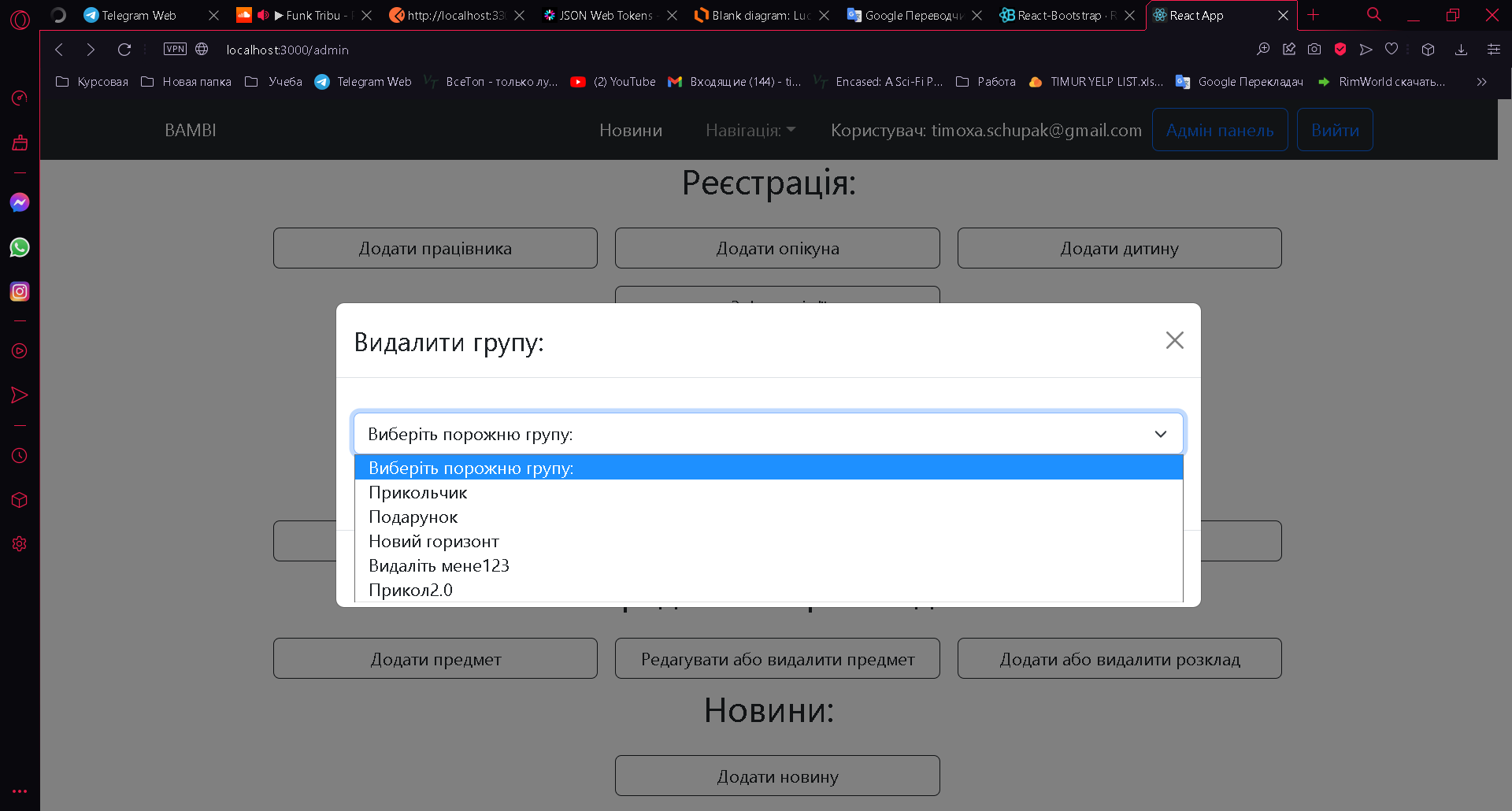


Рисунок 10.18- Видалити групу

На вибір даються лише порожні групи, в яких не має дітей. Перед тим як видаляти групу Адміністратор повинен перевести всіх дітей в іншу групу, або ж випустити їх.



Рисунок 10.19- Список батьків для адміністратор

Якщо натиснути кнопку «Видалити» то запис про опікуна буде видалений з ІС. Натиснувши кнопку «Редагувати» ми перейдемо на сторінку де можна змінити будь-яку інформацію.

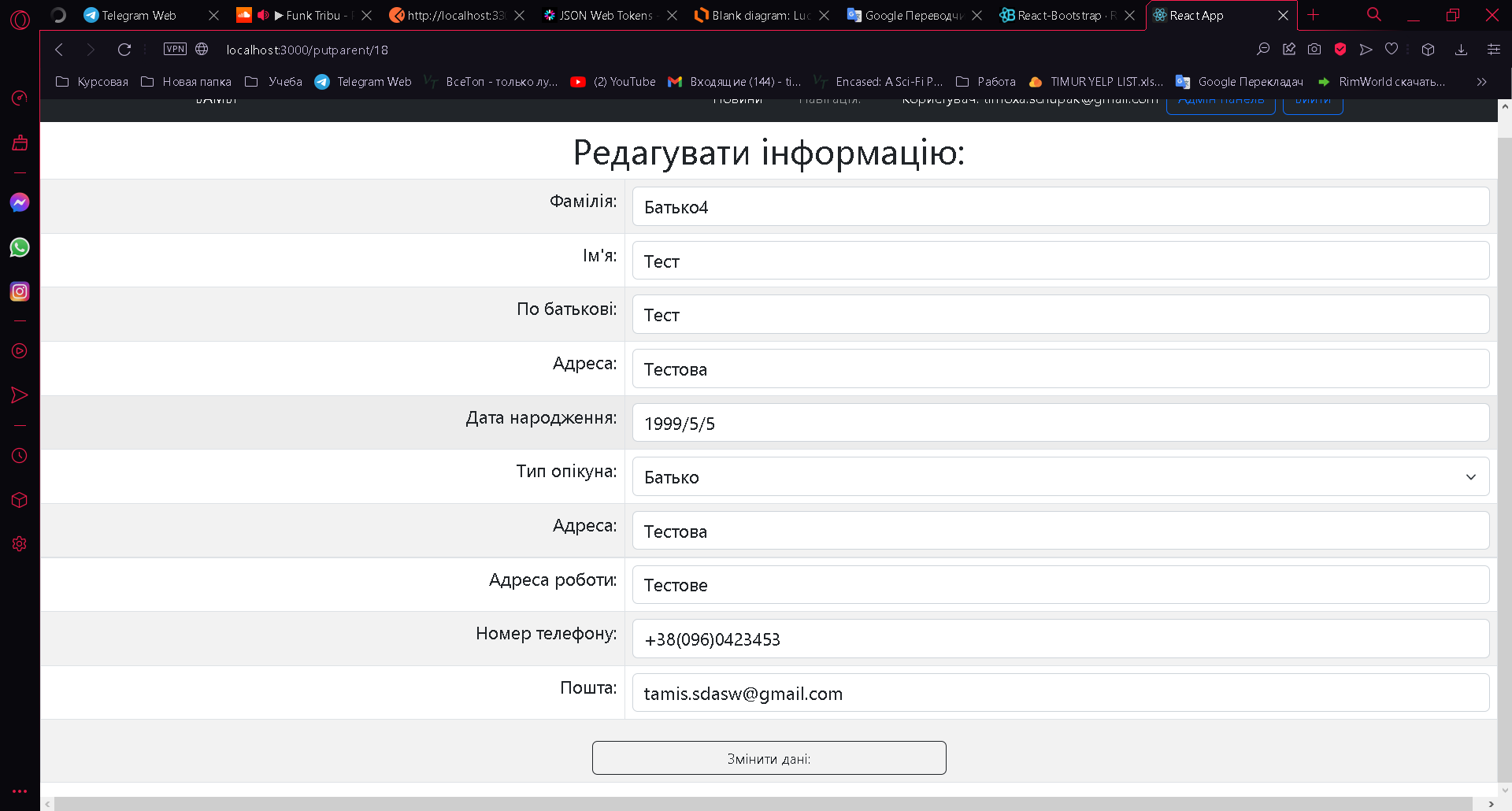


Рисунок 10.20- Панель редагування опікунів

Дані опікуна автоматично записуються в форми. Таким чином ми можемо змінити від одного до всіх полів з інформацією.

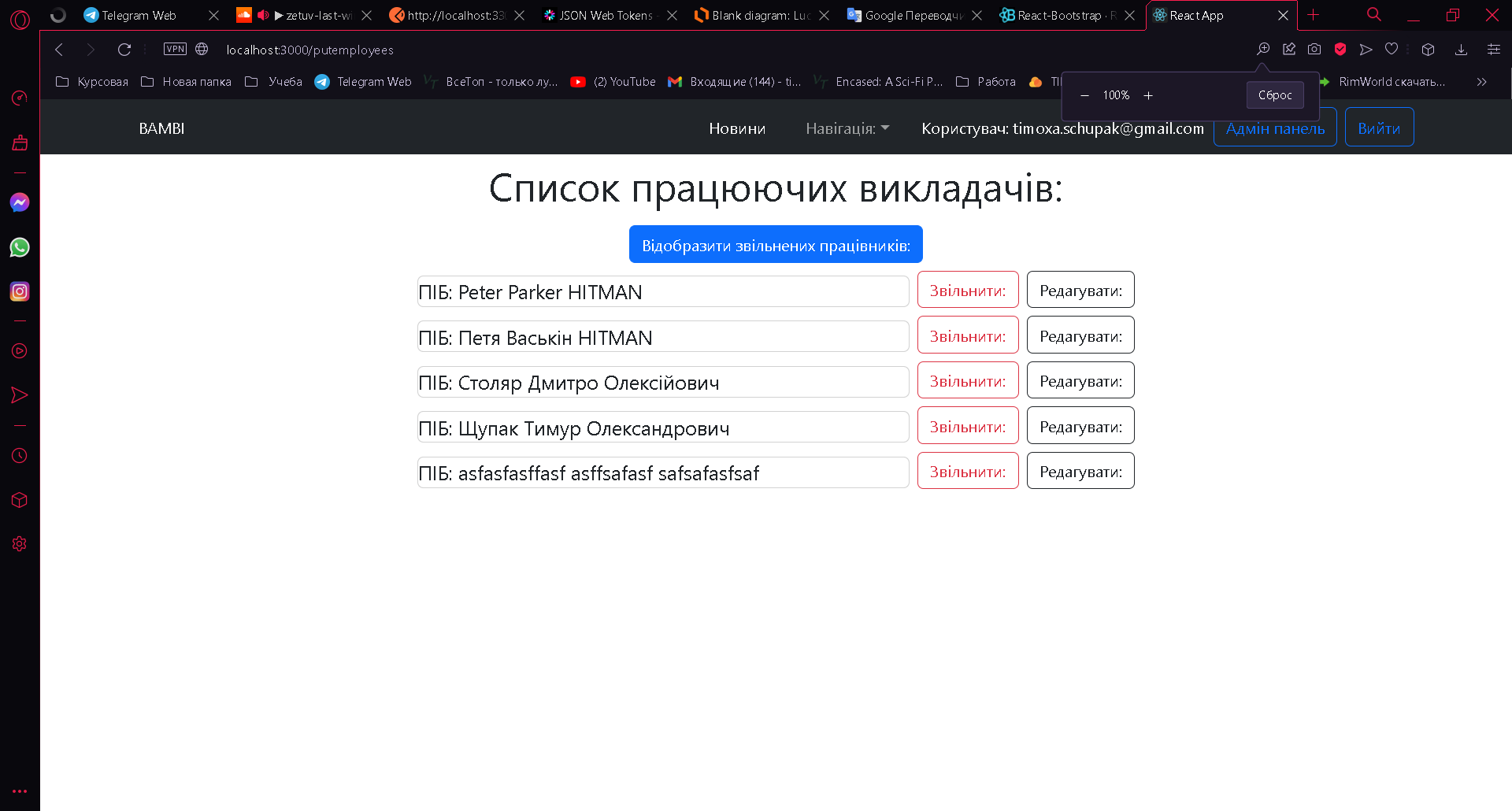


Рисунок 10.21- Список працівників для адміністратора

Напроти кожного працівника є кнопка «Звільнити» натискаючи її статус працівника змінюється на 0. І так як в нашому представлені для авторизації обрані лише працівники зі статусом 1, то звільнений працівник не може авторизуватися до нашого додатку.

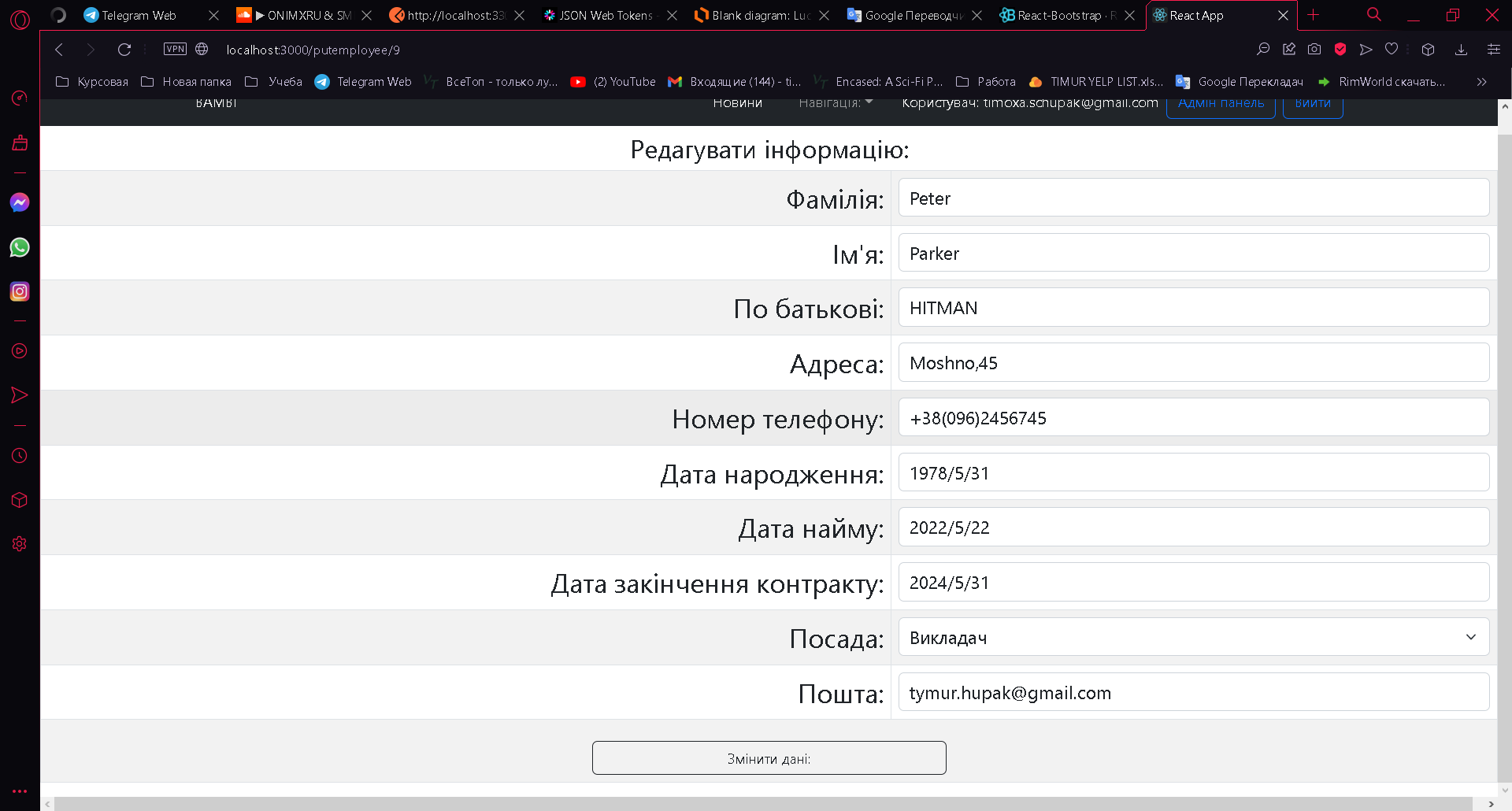


Рисунок 10.22- Панель редагування працівників

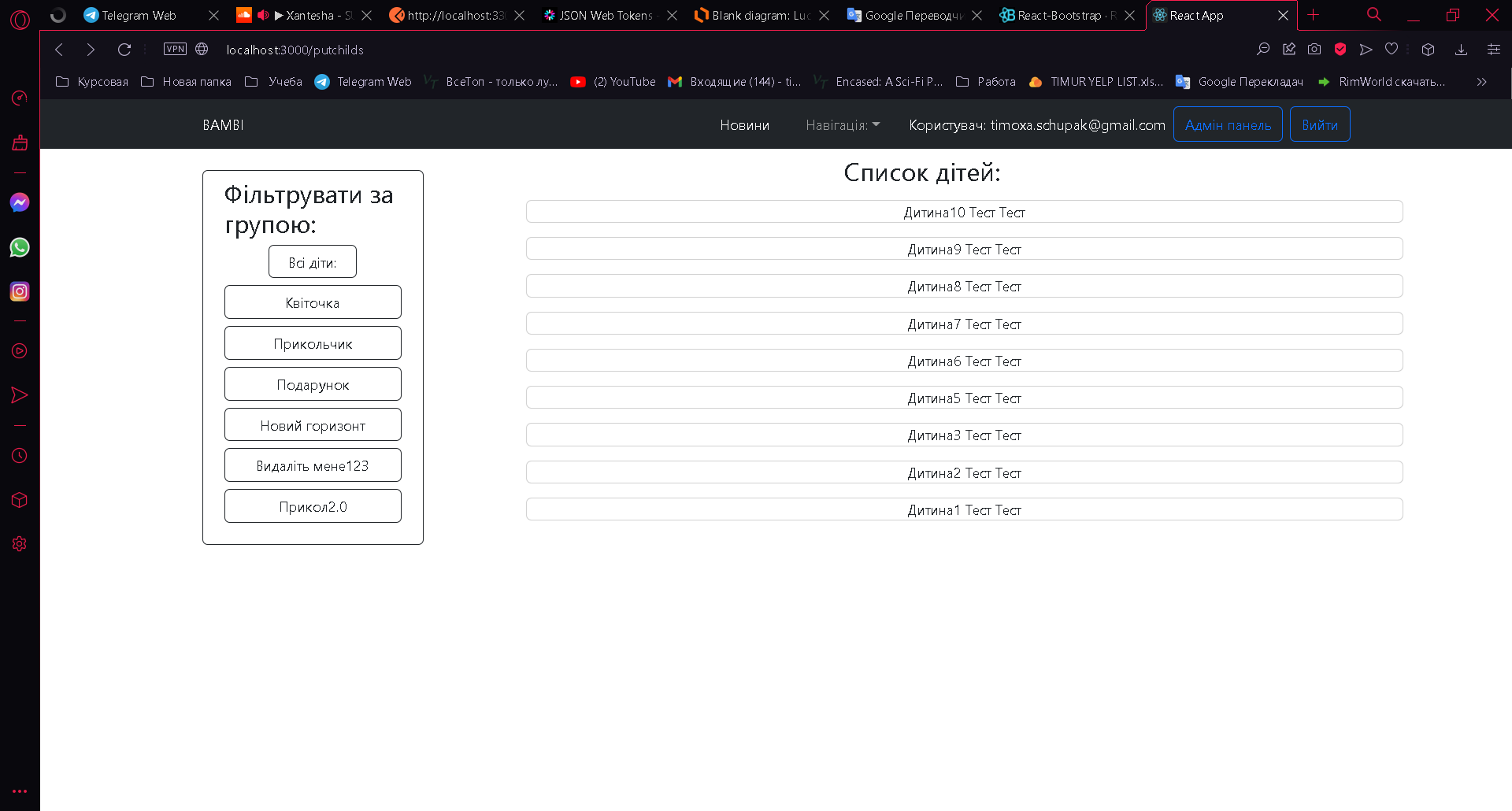


Рисунок 10.23- Список дітей для адміністратора

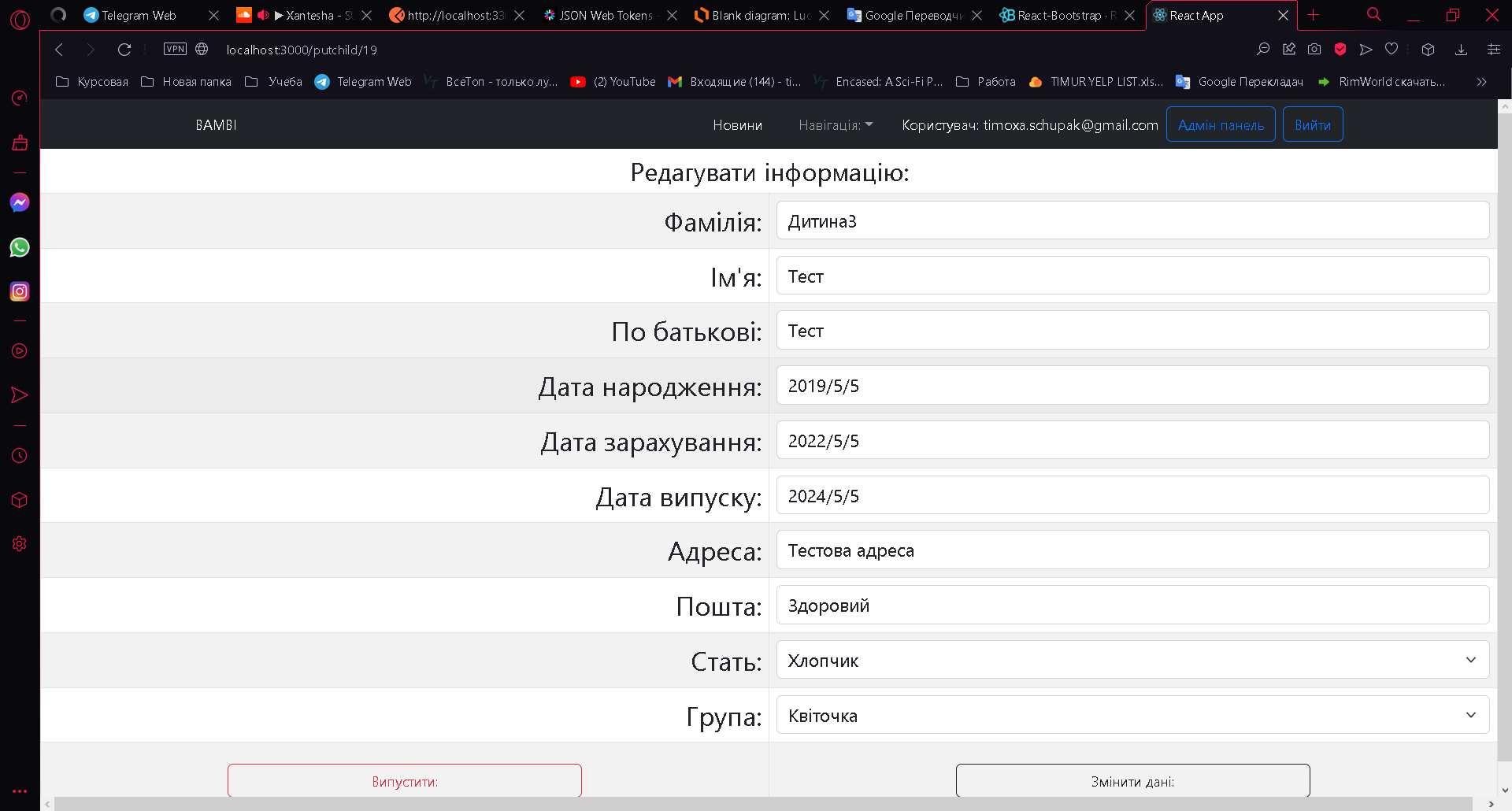


Рисунок 10.24- Панель редагування та випуску дітей

На цій панелі також є кнопка «Випустити» яка видаляє запис про дитину з ІС. Вся інформація про дитину з її середніми оцінками атестату зберігається в іншій таблиці автоматично.

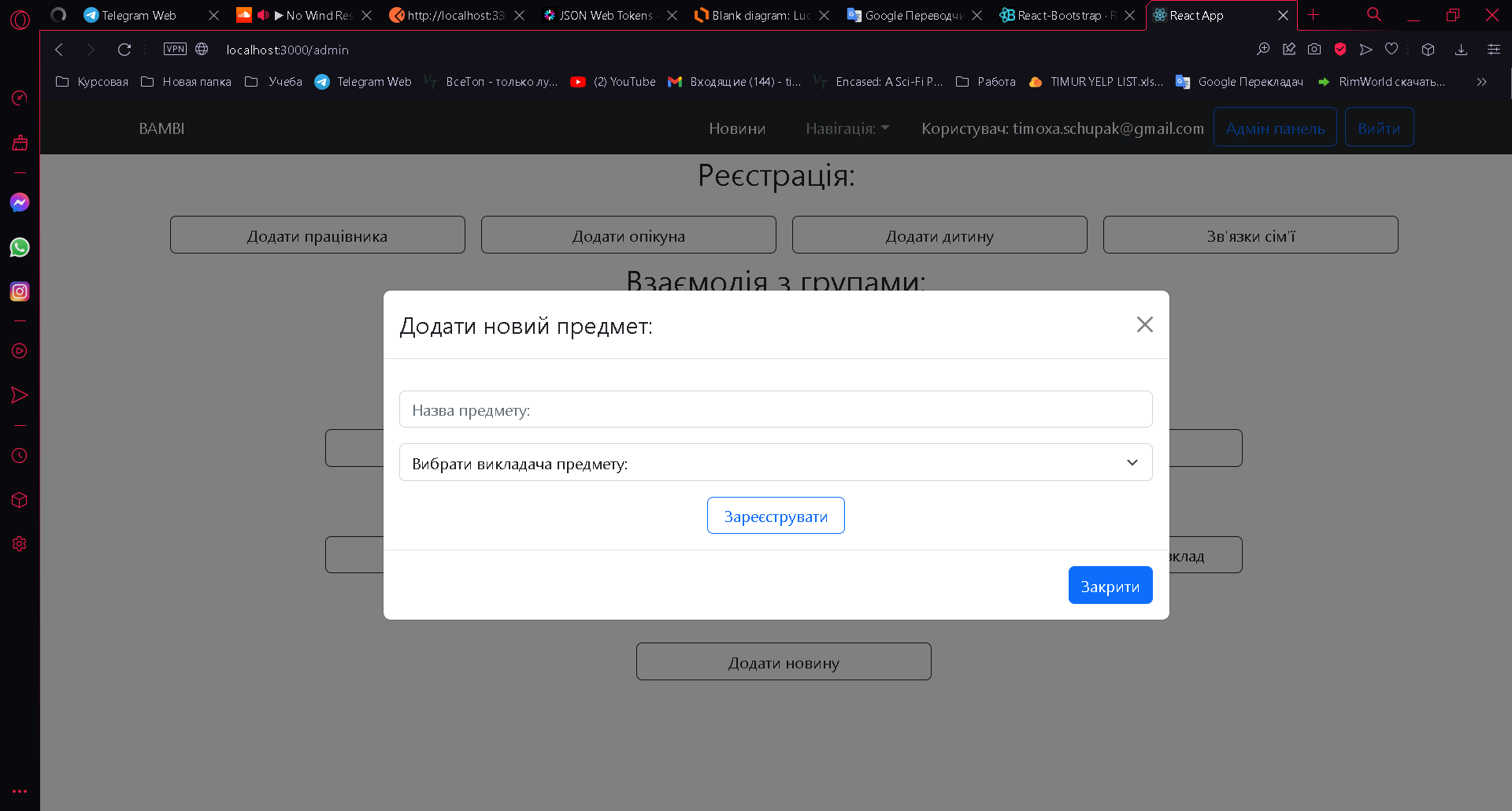


Рисунок 10.25- Додати новий предмет

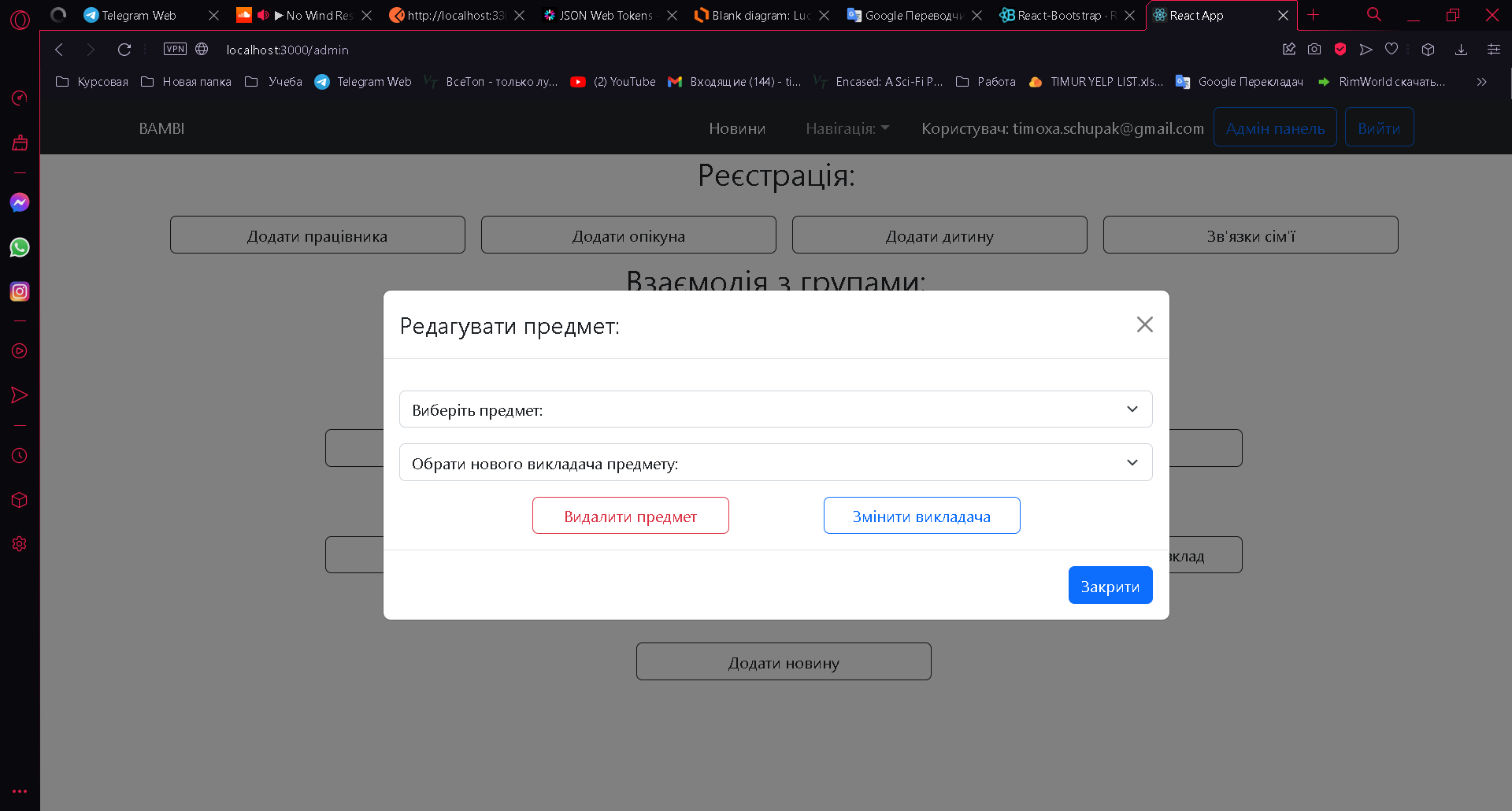


Рисунок 10.26- Редагувати викладача предмету



Рисунок 10.27- Додати або видалити запис з розкладу

Зліва можна обрати групу чий розклад ми будемо змінювати. В середній таблиці сам розклад з кнопкою «Видалити» біля кожного запису з розкладу.

В правій таблиці ми можемо обрати день, час, предмет та групу в яку додамо запис до розкладу.

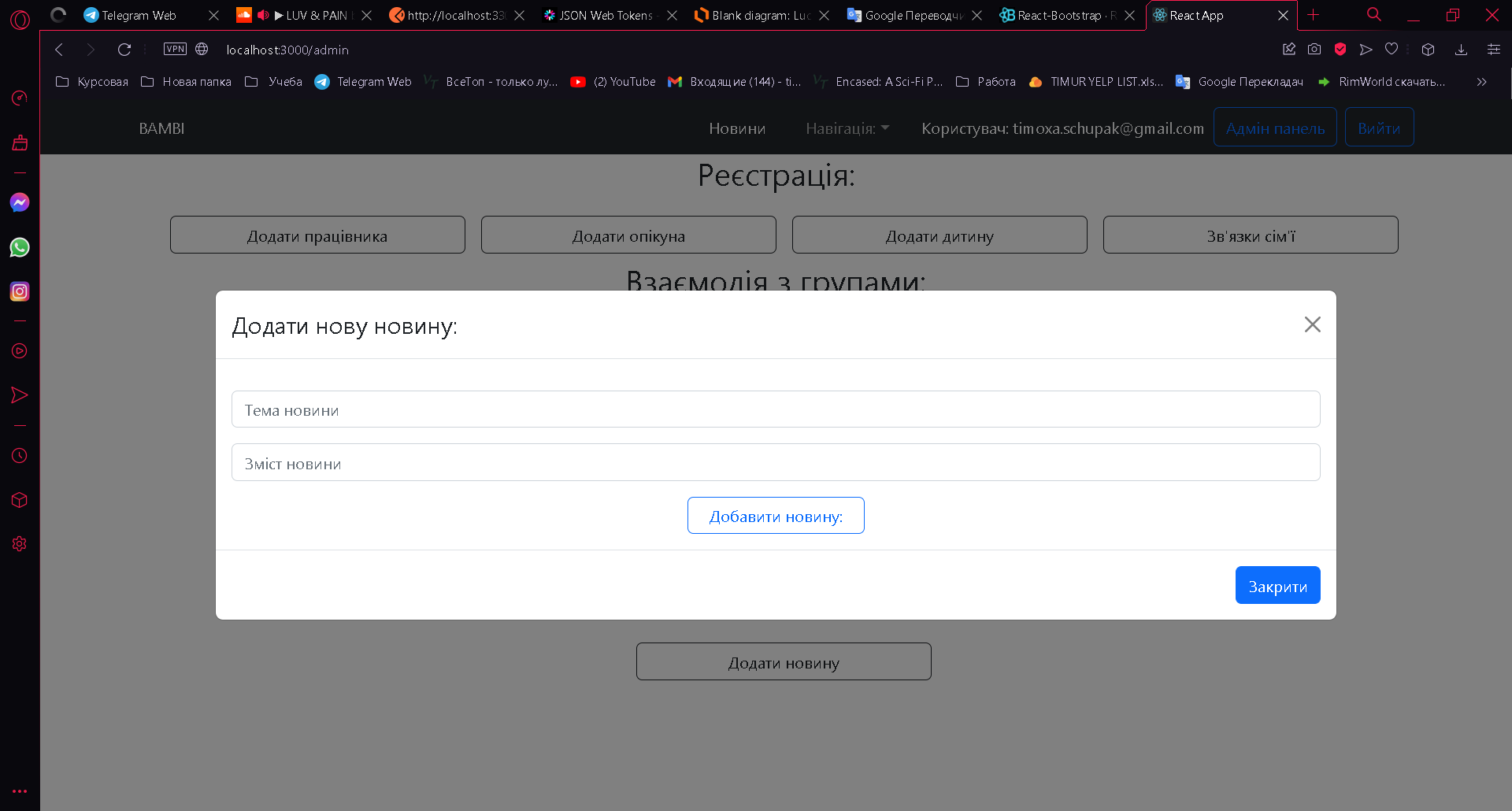


Рисунок 10.28- Додати новину

В першій формі ми пишемо тему новини, а в нижній зміст новини.

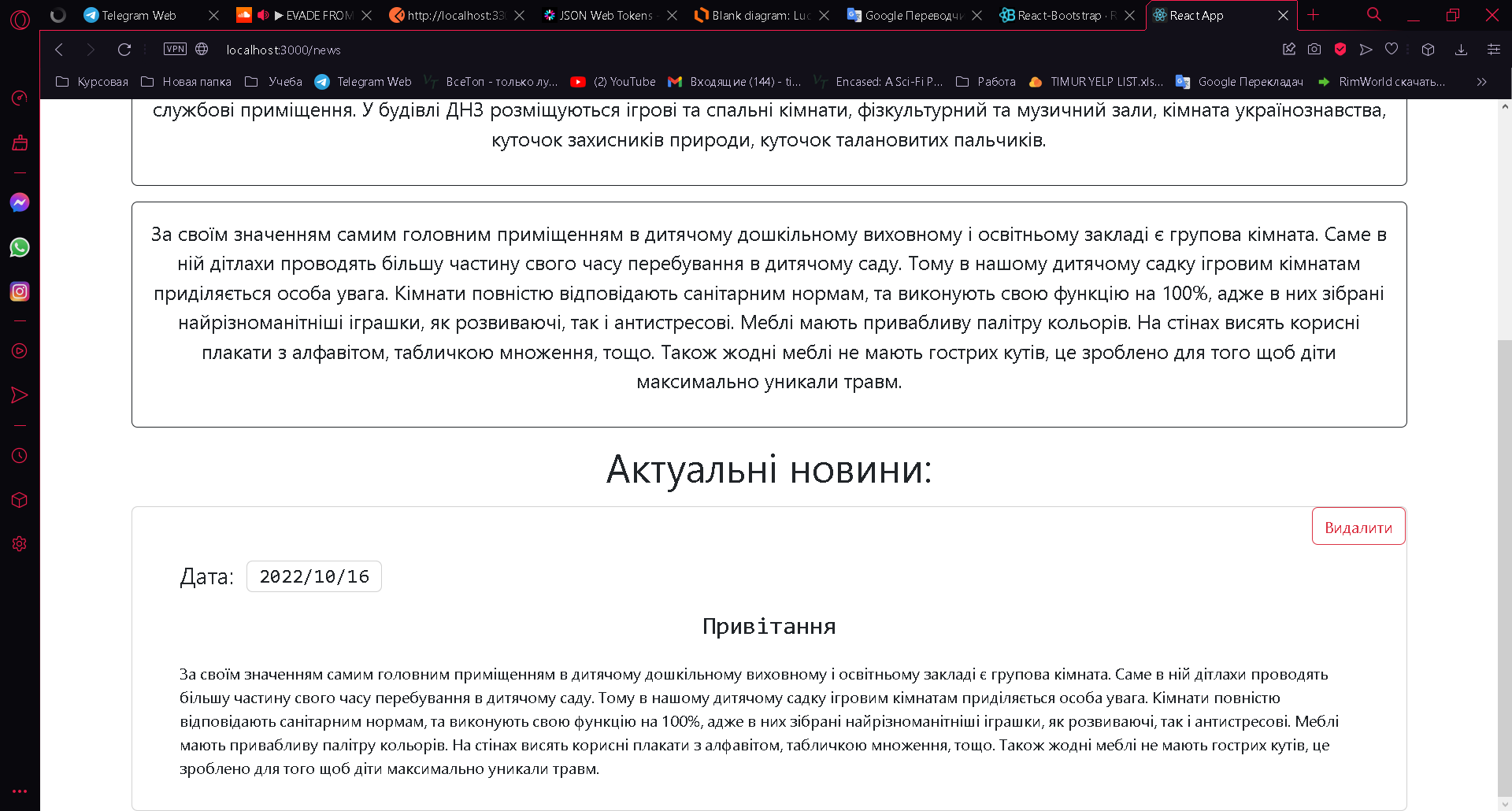


Рисунок 10.29- Сторінка новин для адміністратора

В правому верхньому куті є кнопка «Видалити» її бачать тільки користувачі з ролю Адміністратора. Дата при створені новини автоматично генерується сьогоднішнім числом місяцем і роком.

**ВИСНОВКИ**

В даному курсовому проекті виконано аналіз предметної області, в результаті якого визначено користувачів системи та їх задачі, спроектовано і розроблено базу даних на основі СУБД PostgreSQL яка реалізує реляційну модель даних; розроблено веб-додаток з використанням сучасних програмних засобів та бібліотек, таких як MobX, Axios, React, Express, pg-npm.

Інтерфейс клієнтської програми розроблений з урахуванням вимог кожної категорії користувачів і надає необхідний функціонал для вирішення відповідних задач. Доступ до даних з боку різних категорій користувачів розмежований за допомогою механізму ролей і прав доступу. Захист від несанкціонованого доступу реалізований шляхом використання механізму автентифікації і авторизації.

Інформаційна система дитячого садка покликана полегшити роботу та адміністрування дитячого садка. Спростити доступ користувачів до загальної та конфіденційної інформації, яка надається відповідно до рівня доступу, який залежить від типу користувача. Також ця інформаційна система полегшить збирання та зберігання інформації та звітів виховної установи. Дасть швидкий доступ батькам до всієї інформації, що їх цікавить, про власних дітей, про виховну установу т

а її новини.

**СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ**

1. Архітектура REST [Електронний ресурс] – Режим доступу: https://habr.com/ru/post/38730/
2. Патерни для новачків: MVC vs MVP vs MVVM. [Електронний ресурс] – Режим доступу: https://habr.com/ru/post/215605/
3. Node.js. Документація [Електронний ресурс] – Режим доступу: https://nodejs.org/uk/docs/
4. Node.js, Express та MongoDB: API за півгодини. [Електронний ресурс] – Режим доступу: https://habr.com/ru/company/ruvds/blog/321104/
5. Рейтинг мов програмування 2022 [Електронний ресурс] – Режим доступу: https://dou.ua/lenta/articles/language-rating-2022/

# ДОДАТОК А Опис сутностей та їх властивостей

Таблиця А.1 – Опис сутностей та їх властивостей

|  |  |  |
| --- | --- | --- |
| Властивість | Опис | Обмеження |
| Об’єкт employee2\_0 |
| Employee\_id | Індекс працівника | Первинний ключ, тип BIGSERIAL, не порожнє |
| Lname\_ | Фамілія | 50 символів VARCHAR не порожнє |
| Fname\_ | Ім’я | 50 символів VARCHAR не порожнє |
| Fathername\_ | По батькові | 50 символів VARCHAR не порожнє |
| Position\_ | Посада (Адміністратор/Викладач) | 50 символів VARCHAR не порожнє |
| email | Електронна пошта | 50 символів VARCHAR не порожнє |
| Password\_ | Пароль | 250 символів VARCHAR не порожнє |
| status | Працює чи звільнений | Smallint DEFAULT 1 |
| Birth\_date | Дата народження | Date не порожнє |
| Hiring\_date | Дата найму | Date не порожнє |
| Contract\_expiration | Дата закінчення контракту | Date не порожнє |
| Phone\_num | Номер телефону | VARCHAR(20) NOT NULL UNIQUE |
| Властивість | Опис | Обмеження |
| Parent2\_0 |
| parent\_id | Індекс опікуна | BIGSERIAL PRIMARY KEY NOT NULL |
| lname\_ | Ім’я | VARCHAR(30) NOT NULL |
| fname\_ | Фамілія | VARCHAR(30) NOT NULL |
| fathername\_ | По батькові | VARCHAR(30) NOT NULL |
| birth\_date | Дата народження | DATE NOT NULL |
| parent\_type | Тип опікуна | VARCHAR(20) NOT NULL |
| adress | Адреса проживання | VARCHAR(50) NOT NULL |
| work\_place | Адреса роботи | VARCHAR(100) NOT NULL |
| phone\_num | Номер телефону | VARCHAR(20) NOT NULL |
| email | Електронна пошта | VARCHAR(50) NOT NULL |
| password\_ | Пароль | VARCHAR(250) NOT NULL |
| Об’єкт child2\_0 |
| child\_id | Індекс дитини | BIGSERIAL PRIMARY KEY NOT NULL |
| child\_lname | Ім’я | VARCHAR(30) NOT NULL |
| child\_fname | Фамілія | VARCHAR(30) NOT NULL |
| child\_fathername | По батькові | VARCHAR (30) NOT NULL |
| child\_birth | День народження | date DATE NOT NULL |
| receipt\_date | Дата зарахування | date DATE NOT NULL |
| release\_date | Дата випуску | date DATE NOT NULL |
| child\_adress | Адреса проживання | VARCHAR(50) NOT NULL |
| medical\_comment | Медичне заключення | VARCHAR(350) NOT NULL |
| gender | Гендер | VARCHAR(10) NOT NULL |
| fk\_group\_id | Індекс групи в якій навчається дитина | INTEGER REFERENCES group2\_0(group\_id) on delete no action on update cascade NOT NULL |
| Об’єкт group2\_0 |
| group\_id | Індекс групи | SERIAL PRIMARY KEY NOT NULL |
| group\_name | Назва групи | VARCHAR(20) NOT NULL UNIQUE |
| type\_ | Тип групи | VARCHAR(20) NOT NULL |
| fk\_employee\_id | Індекс працівника який є керівником групи | INTEGER REFERENCES employee2\_0(employee\_id) on delete cascade on update cascade NOT NULL |
| Об’єкт family2\_0 |
| fk\_parent\_id | Індекс опікуна, у якого сімейні відносини з дитиною | INTEGER REFERENCES parent2\_0(parent\_id) on delete cascade NOT NULL |
| fk\_child\_id | Дитина | INTEGER REFERENCES child2\_0(child\_id) on delete cascade NOT NULL |
| family\_id | Індекс запису в сім’ї | SERIAL PRIMARY KEY NOT NULL |
| Об’єкт dairy2\_0 |
| dairy\_num | Індекс записа в щоденник | BIGSERIAL PRIMARY KEY NOT NULL |
| fk\_child\_id | Індекс дитини якій належить запис | INTEGER REFERENCES child2\_0(child\_id) on delete cascade on update cascade NOT NULL |
| date\_ | Дата запису | DATE NOT NULL default current\_date |
| nutrition | Оцінка від 1 до 5 по харчуванню | smallint NOT NULL |
| activity | Оцінка від 1 до 5 по активності | smallint NOT NULL |
| gen\_mood | Оцінка від 1 до 5 по загальному настрою | smallint NOT NULL |
| Об’єкт subject2\_0 |
| subject\_id | Індекс предмету | SMALLSERIAL PRIMARY KEY NOT NULL |
| subject\_name | Назва предмету | VARCHAR(30) NOT NULL |
| fk\_employee\_id | Індекс працівника який веде предмет | INTEGER REFERENCES employee2\_0(employee\_id) on delete no action on update cascade NOT NULL |
| Об’єкт schedule2\_0 | | |
| day\_week | День неділі  (Понеділок -1,  Вівторок -2 …..  П’ятниця -5) | SMALLINT NOT NULL |
| time\_ | Час початку уроку | VARCHAR(10) NOT NULL |
| fk\_subject\_id | Індекс предмета | INTEGER REFERENCES subject2\_0(subject\_id) |
| fk\_group\_id | Індекс групи у якої предмет в цей час | INTEGER REFERENCES group2\_0(group\_id) on delete no action on update cascade NOT NULL |
| schedule\_id | Індекс запису в розклад | SERIAL PRIMARY KEY NOT NULL |
| Об’єкт child\_audit | | |
| child\_id | Індекс дитини | BIGINT PRIMARY KEY NOT NULL |
| child\_lname | Фамілія дитини що випустилася | VARCHAR(30) NOT NULL |
| child\_fnamе | Ім’я дитини що випустилася | VARCHAR(30) NOT NULL |
| child\_fathername | По батькові дитину що випустилася | VARCHAR(30) NOT NULL |
| receipt\_date | Дата зарахування | DATE NOT NULL |
| child\_birth\_date | Дата народження | DATE NOT NULL |
| release\_date | Дата випуску | DATE NOT NULL |
| avg\_gen\_mood | Оцінка в атестаті за настрій | smallint not null |
| avg\_activity | Оцінка в атестаті за активність | smallint not null, |
| avg\_nutrition | Оцінка в атестаті за харчування | smallint not null, |
| Об’єкт news\_table | | |
| news\_id | Індекс новини | SERIAL PRIMARY KEY NOT NULL |
| date\_ | Дата новини | date NOT NULL default current\_date |
| topic | Тема новини | VARCHAR(100) not null |
| text\_ | Зміст новини | VARCHAR(2000) not null |

# ДОДАТОК Б Схема бази даних

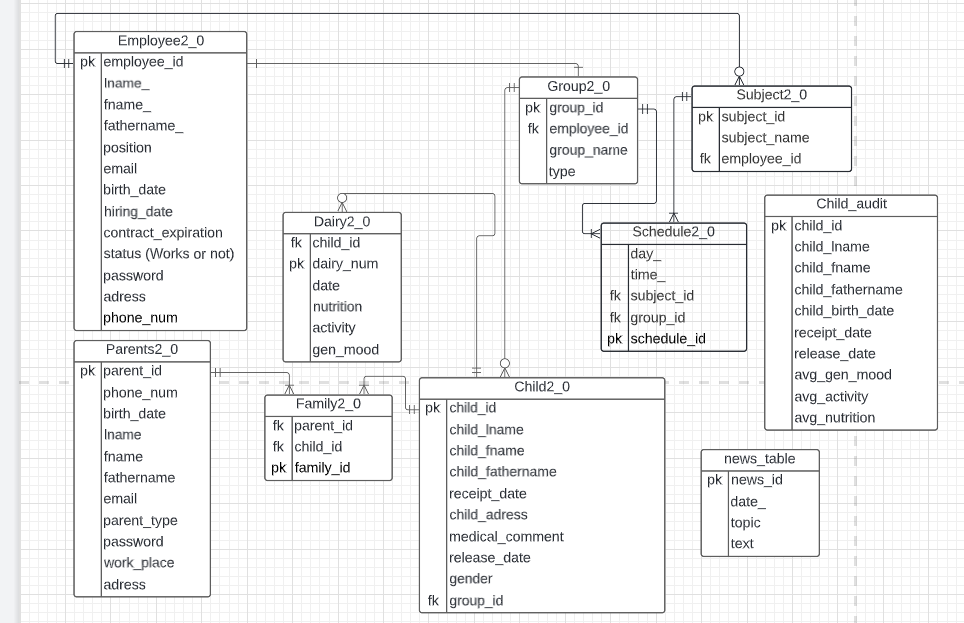


Рисунок Б.1 – Схема бази даних

# ДОДАТОК В Запити для створення бази даних

**Таблиці**

1. Child\_audit

CREATE TABLE IF NOT EXISTS child\_audit (

child\_id BIGINT PRIMARY KEY NOT NULL,

child\_lname VARCHAR(30) NOT NULL,

child\_fname VARCHAR(30) NOT NULL,

child\_fathername VARCHAR (30) NOT NULL,

child\_birth\_date DATE NOT NULL,

receipt\_date DATE NOT NULL,

release\_date DATE NOT NULL,

avg\_gen\_mood smallint not null,

avg\_activity smallint not null,

avg\_nutrition smallint not null

)

2. NEWS\_TABLE

CREATE TABLE IF NOT EXISTS NEWS\_TABLE (

news\_id SERIAL PRIMARY KEY NOT NULL,

date\_ date NOT NULL default current\_date,

topic VARCHAR(100) not null,

text\_ VARCHAR(2000) not null,

CONSTRAINT topic\_ check (char\_length(topic)>=5),

CONSTRAINT text\_ check (char\_length(text\_)>=20)

)

3. employee2\_0

CREATE TABLE IF NOT EXISTS employee2\_0 (

employee\_id SERIAL PRIMARY KEY NOT NULL,

lname\_ VARCHAR(50) NOT NULL,

fname\_ VARCHAR(50) NOT NULL,

fathername\_ VARCHAR(50) NOT NULL,

adress VARCHAR(50) NOT NULL,

phone\_num VARCHAR(20) NOT NULL UNIQUE,

birth\_date date NOT NULL,

hiring\_date date NOT NULL,

contract\_expiration date NOT NULL,

status smallint NOT NULL DEFAULT 1,

position\_ VARCHAR(50) NOT NULL DEFAULT 'Викладач',

email VARCHAR(50) NOT NULL UNIQUE,

password\_ VARCHAR(250) NOT NULL

CONSTRAINT lname\_length check (char\_length(lname\_)>=3),

CONSTRAINT fname\_length check (char\_length(fname\_)>=3),

CONSTRAINT fathername\_length check (char\_length(fathername\_)>=3),

CONSTRAINT status\_restriction check (status >=0 and status <2),

CONSTRAINT email\_restriction check (email similar to '([a-z0-9.\_%-]+)@[a-z0-9.\_%-]+\.[a-z]{2,4}'),

CONSTRAINT age\_restrictions check (birth\_date < current\_timestamp - INTERVAL'18 YEAR' and birth\_date > current\_timestamp - INTERVAL'100 YEAR'),

CONSTRAINT phone\_restrictions check (phone\_num similar to '\+38\((050|063|067|093|096)\)+[0-9]{7}'),

CONSTRAINT password\_lenght check (char\_length(password\_)>=8)

)

4. group2\_0

CREATE TABLE IF NOT EXISTS group2\_0 (

group\_id SERIAL PRIMARY KEY NOT NULL,

group\_name VARCHAR(20) NOT NULL UNIQUE,

type\_ VARCHAR(20) NOT NULL,

fk\_employee\_id INTEGER REFERENCES employee2\_0(employee\_id) on delete cascade on update cascade NOT NULL,

CONSTRAINT type\_restriction check (type\_ similar to 'Junior|Senior')

)

5. subject2\_0

CREATE TABLE IF NOT EXISTS subject2\_0 (

subject\_id SMALLSERIAL PRIMARY KEY NOT NULL,

subject\_name VARCHAR(30) NOT NULL,

fk\_employee\_id INTEGER REFERENCES employee2\_0(employee\_id) on delete no action on update cascade NOT NULL

)

1. schedule2\_0

CREATE TABLE IF NOT EXISTS schedule2\_0 (

day\_week SMALLINT NOT NULL,

time\_ VARCHAR(10) NOT NULL,

fk\_subject\_id INTEGER REFERENCES subject2\_0(subject\_id) on delete no action on update cascade NOT NULL,

fk\_group\_id INTEGER REFERENCES group2\_0(group\_id) on delete no action on update cascade NOT NULL,

schedule\_id SERIAL PRIMARY KEY NOT NULL,

CONSTRAINT day\_restrition check (day\_ similar to 'monday|tuesday|wednesday|thursday|friday'),

CONSTRAINT day\_week\_restriction check (day\_week > 0 and day\_week < 6 ))

1. child2\_0

CREATE TABLE IF NOT EXISTS child2\_0 (

child\_id BIGSERIAL PRIMARY KEY NOT NULL,

child\_lname VARCHAR(30) NOT NULL,

child\_fname VARCHAR(30) NOT NULL,

child\_fathername VARCHAR (30) NOT NULL,

child\_birth\_date DATE NOT NULL,

receipt\_date DATE NOT NULL,

release\_date DATE NOT NULL,

child\_adress VARCHAR(50) NOT NULL,

medical\_comment VARCHAR(350) NOT NULL,

gender VARCHAR(10) NOT NULL,

fk\_group\_id INTEGER REFERENCES group2\_0(group\_id) on delete no action on update cascade NOT NULL,

CONSTRAINT child\_fname check (char\_length(child\_fname)>=3),

CONSTRAINT child\_lname check (char\_length(child\_lname)>=3),

CONSTRAINT child\_fathername check (char\_length(child\_fathername)>=3),

CONSTRAINT age\_restrictions check (child\_birth\_date > current\_timestamp - INTERVAL'6 YEAR'),

CONSTRAINT release\_restrictions check (release\_date > receipt\_date),

CONSTRAINT receipt\_restrictions check (receipt\_date > child\_birth\_date + INTERVAL'2 YEAR'),

CONSTRAINT gender\_restriction check (gender similar to 'Хлопчик|Дівчинка')

)

1. parent2\_0

CREATE TABLE IF NOT EXISTS parent2\_0 (

parent\_id BIGSERIAL PRIMARY KEY NOT NULL,

lname\_ VARCHAR(30) NOT NULL,

fname\_ VARCHAR(30) NOT NULL,

fathername\_ VARCHAR(30) NOT NULL,

birth\_date DATE NOT NULL,

parent\_type VARCHAR(20) NOT NULL,

adress VARCHAR(50) NOT NULL,

work\_place VARCHAR(100) NOT NULL,

phone\_num VARCHAR(20) NOT NULL,

email VARCHAR(50) NOT NULL,

password\_ VARCHAR(250) NOT NULL

CONSTRAINT lname\_length check (char\_length(lname\_)>=3),

CONSTRAINT fname\_length check (char\_length(fname\_)>=3),

CONSTRAINT fathername\_length check (char\_length(fathername\_)>=3),

CONSTRAINT email\_restriction check (email similar to '([a-z0-9.\_%-]+)@[a-z0-9.\_%-]+\.[a-z]{2,4}'),

CONSTRAINT parent\_type\_restriction check (parent\_type similar to 'Батько|Мати|Бабуся|Дідусь|Опікун|Соціальний робітник'),

CONSTRAINT phone\_restrictions check (phone\_num similar to '\+38\((050|063|067|093|096)\)+[0-9]{7}'),

CONSTRAINT password\_lenght check (char\_length(password\_)>=8),

CONSTRAINT age\_restrictions check (birth\_date < (current\_timestamp - INTERVAL'18 YEAR') and birth\_date > (current\_timestamp - INTERVAL'100 YEAR'))

)

1. family2\_0

fk\_parent\_id INTEGER REFERENCES parent2\_0(parent\_id) on delete cascade NOT NULL,

fk\_child\_id INTEGER REFERENCES child2\_0(child\_id) on delete cascade NOT NULL,

family\_id SERIAL PRIMARY KEY NOT NULL,

unique(fk\_parent\_id,fk\_child\_id)

);

10. dairy2\_0

CREATE TABLE IF NOT EXISTS dairy2\_0 (

dairy\_num BIGSERIAL PRIMARY KEY NOT NULL,

fk\_child\_id INTEGER REFERENCES child2\_0(child\_id) on delete cascade on update cascade NOT NULL,

date\_ DATE NOT NULL default current\_date,

nutrition smallint NOT NULL,

activity smallint NOT NULL,

gen\_mood smallint NOT NULL,

CONSTRAINT nutrition\_restriction check (nutrition > 0 and nutrition < 6),

Constraint activity\_restriction check (activity > 0 and activity < 6),

CONSTRAINT gen\_mood\_restriction check (gen\_mood > 0 and gen\_mood <6)

)

**Додаток Г**

**Представлення:**

authinfo

CREATE OR REPLACE VIEW authinfo

AS

SELECT employee2\_0.employee\_id,

employee2\_0.email,

'1't AS "Roles",

employee2\_0.password\_,

employee2\_0.position\_ AS type,

employee2\_0.phone\_num

FROM employee2\_0

WHERE employee2\_0.position\_ = 'Адміністратор' AND employee2\_0.status = 1

UNION

SELECT employee2\_0.employee\_id,

employee2\_0.email,

'2' AS "Roles",

employee2\_0.password\_,

employee2\_0.position\_ AS type,

employee2\_0.phone\_num

FROM employee2\_0

WHERE employee2\_0.position\_ = 'Викладач' AND employee2\_0.status = 1

UNION

SELECT parent2\_0.parent\_id AS employee\_id,

parent2\_0.email,

'3' AS "Roles",

parent2\_0.password\_,

parent2\_0.parent\_type AS type,

parent2\_0.phone\_num

FROM parent2\_0;

**Тригери**

1.Тригер який провіряє чи залишається при видаленні опікуна і дитини хоча б одні родинні зв’язки

CREATE or REPLACE FUNCTION check\_family() returns trigger

AS $$

DECLARE kid\_id\_check INTEGER;

DECLARE parent\_count INTEGER;

BEGIN

Select child\_id from parent2\_0

inner join family2\_0 on family2\_0.fk\_parent\_id = parent2\_0.parent\_id

inner join child2\_0 on child2\_0.child\_id = family2\_0.fk\_child\_id

where old.parent\_id = parent\_id

into kid\_id\_check;

SELECT count(parent\_id) from parent2\_0

inner join family2\_0 on family2\_0.fk\_parent\_id = parent2\_0.parent\_id

inner join child2\_0 on child2\_0.child\_id = family2\_0.fk\_child\_id

where child\_id = kid\_id\_check

into parent\_count;

if

parent\_count = 1

then

update parent2\_0

set parent\_id = old.parent\_id, phone\_num = '+38(096)4236456',birth\_date='1989/06/04',lname\_='Колаковський',fname\_='Артем', fathername = 'Альбертович',email = 'artem.sc@gmail.com',parent\_type = 'Соціальний робітник',password\_ = old.password\_ ,adress = 'Генуеская,5',work\_place = 'Софиевская,4/Соціальный робітник'

where parent\_id = old.parent\_id;

return new;

else

return old;

end if;

END;

$$ LANGUAGE plpgsql;

CREATE TRIGGER check\_family\_trigger

BEFORE UPDATE OR DELETE ON parent2\_0

FOR EACH ROW

WHEN (pg\_trigger\_depth() < 1)

EXECUTE PROCEDURE check\_family();

1. Тригер який провіряє перед створенням дитини кількість дітей в групі. Якщо число більше 20 то дитина не створюється

CREATE or REPLACE FUNCTION check\_group\_amount() returns trigger

AS $$

Declare

kidscount smallint;

BEGIN

select count(child\_id) from child2\_0

into kidscount

where new.fk\_group\_id = fk\_group\_id;

if kidscount >= 20

then

raise exception 'Группа может состоять максимум из 20-ти детей!';

return null;

elsif

kidscount <= 20

then

return new;

end if;

END;

$$ LANGUAGE plpgsql;

CREATE TRIGGER check\_group\_amount\_trigger

BEFORE UPDATE or insert ON child2\_0

FOR EACH ROW

EXECUTE PROCEDURE check\_group\_amount();

1. Тригер який при видаленні дитини записує її інформацію та середні оцінки зі щоденника в таблицю Child\_audit

CREATE or REPLACE FUNCTION child\_audit() returns trigger

AS $$

BEGIN

insert into child\_audit

select child\_id,child\_lname,child\_fname,child\_fathername,child\_birth\_date,receipt\_date,

release\_date, avg(gen\_mood)::numeric(10,2) as Общее\_настроение , avg(activity)::numeric(10,2) as Активности, avg(Nutrition)::numeric(10,2) as Питание

from child2\_0 inner join dairy2\_0 on

child2\_0.child\_id = dairy2\_0.fk\_child\_id where child\_id = old.child\_id

group by child\_id;

return old;

END;

$$ LANGUAGE plpgsql;

CREATE TRIGGER child\_audit\_trigger

BEFORE DELETE ON child2\_0

FOR EACH ROW

EXECUTE PROCEDURE child\_audit();