fast discrete signal transforms", IEEE Transactions on Signal Processing, Vol. 49, No. 9, September 2001, pp. 1992–2002. 8. Макклеллан Дж., Рейдер Ч. Применение теории чисел в цифровой обработке сигналов. – М.: Радио и связь, 1983. 9 Wang Z., A fast algorithm for the discrete sine transform implemented by the fast cosine transform // IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol. ASSP-30, October 1982, pp. 814–815. 10. Lee P., Huang F.-Y. Restructured recursive DCT and DST algorithms, IEEE Trans. Signal Processing 42 (7) (1994) 1600–1609. 11. Britanak V. The fast DCT-IV/DST-IV computation via the MDCT, Sig. Proc. 83 (2003) 1803–1813. 12. Чуприна О.О. Удосконалений алгоритм ШПФ на базі швидкої згортки // Вісник Нац. ун-ту "Львівська політехніка". – 2008. – № 618. – С. 174–179. 13. Wang Z. and Hunt B. The discrete W-transform, Appl. Math. Comput., 16 (1985), pp. 19–48. 14. Патент 96540 Україна, G06F 17/16 (2006.01), H03M 7/30 (2006.01). Спосіб приведення дискретних гармонічних складових цифрових сигналів до циклічних згорток. / І.О. Процько / Опубл. 10.11.2011, Бюл. № 21.

УДК 004.657

О.В. Столярчук, Р.Я. Шувар, А.М. Продивус
Львівський національний університет імені Івана Франка,
факультет електроніки

КОНЦЕПТУАЛЬНА МОДЕЛЬ ІНФОРМАЦІЙНОЇ СИСТЕМИ РОЗКЛАДУ ЗАНЯТЬ З ДОСТУПОМ ЧЕРЕЗ HTTP-ПРОТОКОЛ

Розглянуто модель інформаційної системи розкладу занять навчального закладу та організацію роботи з нею через HTTP-протокол. Проаналізовано запити, які користувач може створювати, раціональний вибір та оптимальність створення таблиць і зв'язків між ними. Досліджено обмеження, які виникають під час проектування структури таких баз, вимоги до цілісності та несуперечливості даних.

Ключові слова: Інтернет, база даних, веб-інтерфейс.

We consider a model of information system schedule institution and organization work with it via HTTP-protocol. Analysis of queries that the user can create, rational choice and optimization to create tables and relationships between them. Study limitations that arise when designing the structure of such bases, requirements for integrity and consistency of data.

Key words: Internet, database, web-interface.

Вступ. Постановка проблеми

Сьогодні в багатьох галузях виникає потреба систематизації та автоматичної обробки і збереження даних, що вимагає для цього використання баз даних. Використання сучасних комунікаційних систем надає можливість як локального, так і віддаленого доступу до даних, насамперед через http-протокол. Широке використання цього протоколу для доступу до віддалених баз даних зумовлено його універсальністю та підтримкою на різноманітних платформах і операційних системах. Його можна використати в локальних і глобальних мережах. Він не вимагає жодного додаткового програмного забезпечення, крім браузера. Це дає змогу вирішити проблему сумісності та забезпечити доступ до системи з будь-якого робочого місця. Такий підхід є ефективним під час розроблення баз даних для інформаційних систем у багатьох галузях, зокрема, в медицині, освіті тощо, які характеризуються неоднорідністю робочих місць, частою зміною користувачів і великим територіальним заповненням.

Однією з основних проблем роботи з даними є розроблення структури бази даних [1]. Від її раціонального проектування залежатиме в подальшому ефективність роботи. Багато серйозних проблем у подальшій роботі виникає через неправильну або не до кінця продуману організацію даних саме на етапі проектування структури бази даних. Це пов'язано з тим, що під час використання інформаційної системи, яка вже містить багато даних, доводиться докладати багато зусиль і застосовувати нестандартні рішення для виправлення помилок. Такий підхід допомагає частково вирішити проблему, водночас призводячи до зменшення швидкодії та ефективності використання ресурсів. Саме тому на етапі проектування бази даних необхідно максимально врахувати можливі проблеми та різні варіанти використання даних.

Аналіз наявних методів

Сьогодні є багато програмних продуктів, які дозволяють отримати доступ до віддалених баз даних. Часто використовують спеціальне програмне забезпечення (клієнт), що виконується на робочому місці користувача, тобто певну складову частину автоматизованої системи, яка розроблена для вирішення проблем цього користувача. Такий комплекс програм може бути розроблений у різних середовищах програмування, наприклад C++, Паскаль, Delphi, Visual C++ тощо. Деякі виробники СКБД пропонують свої унікальні відкриті і доволі потужні засоби під'єднання до віддалених баз даних. Зокрема корпорація Oracle пропонує для під'єднання до віддалених баз даних свій унікальний продукт Net8, який можна використовувати з великою кількістю мережевих протоколів, як, наприклад, TCP/IP, OSI, IPX/SPX і може працювати під управлінням поширених операційних систем [2].

Однак у разі використання цих технологій виникають проблеми, пов'язані з сумісністю робочих місць. Кожна з них вимагає наявності свого програмного забезпечення на клієнтській машині, тобто до кожної з цих систем потрібно налаштовувати окремий модуль. Крім того, використання такого програмного забезпечення вимагає спеціальної підготовки користувачів.

Цих недоліків позбавлена технологія використання http-протоколу для доступу до віддалених баз даних. Вибір конкретних рішень під час розроблення веб-інтерфейсів залежить від особливостей самої інформації та потреб роботи з нею [3].

Формулювання мети статті

У цій статті розглянуто модель інформаційної системи розкладу занять вищого навчального закладу та організацію доступу до неї через http-протокол. Запропоновано реляційну модель бази даних, проаналізовано оптимальність створення зв'язків між таблицями та запити, які можна здійснювати. Досліджено вимоги до цілісності й несуперечливості даних. Вибір інформаційної системи розкладу занять зумовлений актуальністю та характером проблем, які виникають під час її розроблення, що найповніше охоплюють особливості створення баз даних з http-доступом до них. Під час формування інформаційної системи розкладу занять у вищому навчальному закладі необхідно врахувати особливості організації навчального процесу [5, 6]. Зокрема заняття можуть проводитися курсами, потоками, групами, підгрупами, проходити протягом цілої або півпари, щотижня або раз на два тижні (по знаменнику чи по чисельнику). Потрібно відслідковувати наявність часового збігу проведення занять по викладачах та аудиторіях. Тому виникає потреба у структуруванні розкладу, приведенні його до логічного й однозначного вигляду та створенні інформаційної системи, яка дозволила б ефективно формувати розклад занять з урахуванням всіх згаданих вище особливостей. Крім того, під час розробки структури бази даних необхідно враховувати те, що доступ буде здійснюватись через http-протокол. Тож формування її має відбуватися так, щоб можна було в разі потреби без особливих труднощів розподілити її вміст по декількох місцях у мережі і отримувати доступ до даних через веб-інтерфейс. Потрібно також створити зручний та інтуїтивно зрозумілий користувачам без особливої підготовки веб-інтерфейс для роботи з розкладом, який надавав би можливість отримувати різноманітні варіанти інформації щодо розкладу занять у стандартній формі [7].

Аналіз отриманих результатів

На етапі проектування до інформаційної системи висувалися такі вимоги: можливість імпорту даних про предмети, навчальні групи та потоки з зовнішніх файлів, перевірка під час введення

даних на збіг по викладачах та аудиторіях, виведення даних у зручному для сприйняття вигляді, забезпечення швидкого пошуку необхідної інформації в базі даних, можливість контролю кількості введених годин для кожної навчальної групи та предмета.

Веб-інтерфейс до бази даних повинен складатися з двох частин. Адміністративна частина має забезпечувати формування розкладу, внесення поправок і змін. Доступ до неї повинні мати лише користувачі з відповідними повноваженнями. Користувацька частина має надавати можливість будь-якому користувачеві переглянути потрібну йому інформацію. Крім того, необхідно забезпечити доступ до інформаційної системи з мережі Інтернет, веб-інтерфейс повинен бути уніфікованим і використання його не має вимагати в користувачів особливого програмного забезпечення, крім веб-браузера.

Запропоновано використовувати структуру бази даних (рис. 1) [8]. Вона складається з восьми таблиць: auditory, day, rozklad, group, group_teach, stream, teachers, subject.

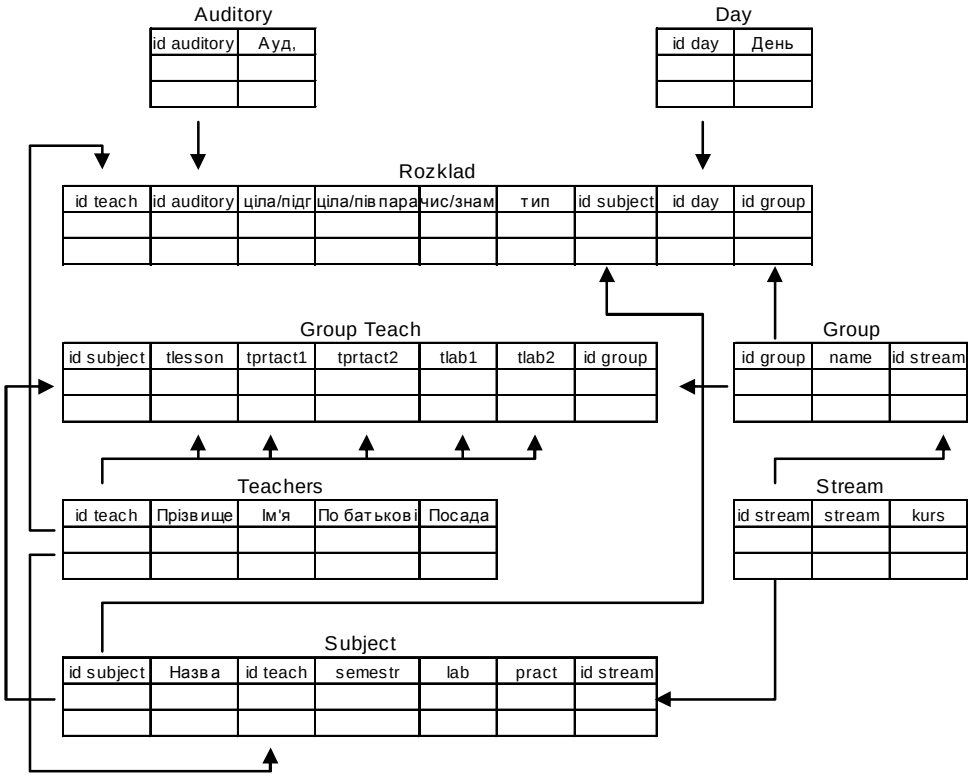

Рис. 1. Структура бази даних системи "Розклад"

Таблиці auditory, day, stream, teachers повністю незалежні і дані в них заносяться без використання інших таблиць. Таблиця auditory має два поля: idauditory і Аудиторія. У полі Аудиторія зберігаються номери всіх аудиторій, в яких проходять заняття, а в полі idauditory первинний ключ для таблиці. У таблиці day зберігаються дні тижня, в які проходить навчання. Поле id day є первинним ключем для неї. Таблиця stream має такі поля: idstream, stream, kurs. В ній міститься інформація про потоки груп (набір груп однієї спеціальності, для яких можуть бути спільні лекції). Поле stream показує, який це потік, а kurs – номер курсу. idstream є первинним ключем. Відповідно на кожному курсі для кожної спеціальності є свій потік і для груп, що входять до нього, можуть бути спільні пари. Це потрібно буде враховувати під час перевірки збігів викладачів та аудиторій. Таблиця teachers містить дані про викладачів (прізвище, ім'я, по батькові, посада, кафедра).

Таблиці auditory, day, teachers та stream зручно заповнити на початку створення бази даних, оскільки під час підготовки розкладу на семестр відомо, в які дні відбувається навчання, які аудиторії наявні та які групи і потоки є на факультеті. Також в основному відомі прізвища викладачів, залучених до навчального процесу. Для цього розроблено модуль, який після створення бази дає змогу імпортувати в неї дані про

прізвища викладачів та доступні аудиторії з зовнішніх текстових файлів або з бази даних розкладу попереднього семестру. Для внесення змін у ці таблиці під час формування розкладу розроблено модуль, який дозволяє додавати чи редагувати інформацію в таблицях teachers та stream.

Таблиці rozklad, group, group_teach, subject заповнюються з використанням даних інших таблиць. Таблиця group містить список груп, які є на факультеті. Поле name містить назву групи визначеного формату, яка включає в себе спеціальність (напрям підготовки), курс і групу. Наприклад, значення І–31 означає, що це напрям підготовки «Комп'ютерні науки», третій курс, перша група. Поле idstream містить код потоку. Через поле idstream робиться зв'язок один до багатьох між таблицями stream і group, тобто idstream є зовнішнім ключем. У ньому вказується, до якого потоку належить група.

Отже, коли нам потрібно буде внести пару для цілого потоку, то через цей зв'язок ми можемо вибрати групи, яким її читатимуть.

Subject – таблиця предметів. У ній міститься інформація про кожний предмет (назва, хто викладає, в якому семестрі читають (у першому, другому, першому і другому), наявність для цього предмета практичних і лабораторних занять, для якого потоку читають). Вона зв'язана з таблицями rozklad, group_teach, stream. У полі stream зберігається список потоків чи груп, для яких читають предмет. Наприклад запис І-1, П-1, М-1, І-2, П-31 означатиме, що цей предмет читають на першому курсі для потоків І, П, М, на другому курсі лише для потоку І, та третьому курсі лише для групи П-31. У полі idteacher зберігаються ідентифікатори викладачів, які можуть читати цей предмет. Це дає змогу під час введення в базу пари з цього предмета вибирати лише з тих викладачів, хто читає цей предмет, а не з усіх.

Основним документом, який використовується під час формування розкладу занять, є робочі навчальні плани, які складаються на основі навчальних планів. Інформація в робочих навчальних планах представлена в чітко визначеному форматі. Для імпорту цієї інформації в базу даних розкладу розроблено модуль, який дозволяє автоматизувати внесення інформації про навчальні предмети і про те, для яких потоків, курсів чи груп їх викладають. Це істотно скорочує час для наповнення бази та полегшує роботу. Крім того, за потреби можна імпортувати також інші параметри навчальних предметів, які є в навчальних планах, для подальшого їх оброблення. Це може бути кількість годин на семестр, на тиждень, кафедра, яка забезпечує читання предмета, форма звітності тощо. Для цього необхідно створити відповідні поля в таблиці subject і в модулі імпорту внести необхідні виправлення. Для оновлення інформації про навчальні предмети розроблено модуль, який дає змогу додавати й редагувати таку інформацію.

Під час проектування структури бази даних розкладу необхідно врахувати різні форми проведення занять з навчального предмета: лекції, практичні, лабораторні, які можуть вести різні викладачі. Крім того, практичні та лабораторні заняття можуть проходити в підгрупах, заняття в яких теж можуть вести різні викладачі. Для цього запропоновано використати таблицю group_teach, яка пов'язує викладачів з предметами і групами або підгрупами. Поле idsubject містить ідентифікатор предмету, а поля tlesson, tpract1, tprakt2, tlab1, tlab2 – ідентифікатори викладачів, які читають лекції, практичні й лабораторні для першої та другої підгрупи, В полі idgroup міститься ідентифікатор групи. Під час формування розкладу потрібно для кожного предмета, що передбачений для тієї чи іншої групи, кожній підгрупі задати викладачів, які будуть викладати його. Коли предмет викладається лише для цілої групи, то буде один запис. Однак коли ми вводитимемо інформацію про лекцію, яка читається для потоку (кілька груп), то для кожної групи потрібно внести свій запис. Це пов'язано з тим, що навіть якщо лектор один, то практичні чи лабораторні для різних груп можуть вести різні викладачі. Такий підхід дає нам змогу задавати викладачів для кожної групи незалежно.

У таблиці rozklad розміщені записи, необхідні для формування розкладу. Поля idteach, idauditory, idsubject, idday, idgroup містять інформацію про викладача, аудиторію, предмет, день, групу відповідно. Ці поля є зовнішніми ключами, які разом з відповідними первинними ключами таблиць auditory, day, rozklad, group, teachers, subject зв'язують таблицю rozklad із цими таблицями. Поля ціла/підг, чис/знам, ціла/півпари, тип, №пари містять у собі всі атрибути різновиду пари. Поле ціла/підг визначає, чи ця пара читається для всієї групи, чи для підгрупи. Кожну групу можна ділити на дві підгрупи, для яких

одночасно можуть бути різні пари або лише в однієї підгрупи буде пара, а в іншої – ні. Поле чис/знам вказує, чи пару читають щотижня, чи через тиждень. Поле ціла/півпари показує, чи це ціла пара, чи половина. Зокрема, доволі часто лабораторні роботи проводяться по півтори пари, тому вказаний параметр дає змогу описати цю особливість. Поле тип визначає форму проведення заняття. Це може бути лекція для групи, лекція для потоку, лекція для курсу, лабораторне заняття, практичне, практичне заняття спільно з іншою групою. Кількість типів визначається особливістю організації занять у конкретному навчальному закладі. Для кожного типу може бути особлива поведінка запису пари. Якщо, скажімо, лекцію читають для потоку, то потрібно зробити запис для кожної з груп, що входять у потік. Коли якась група має по підгрупах одночасно заняття по чисельнику і знаменнику, то в rozklad для цієї групи на один і той же час буде внесено 4 записи, тобто кожний різновид пари вноситься окремо. Під час внесення нових пар враховується сукупність цих параметрів для вже введених записів, що дає змогу уникнути збігу за часом викладачів, аудиторій і пар.

Проаналізовано, які запити реалізуються через веб-інтерфейс доступу до бази даних, а також, як забезпечується цілісність та несуперечливість даних. Система дозволяє внести чи видалити новий предмет і задати, для яких груп він читається, додати викладача і присвоїти та змінювати йому перелік предметів, які він може викладати. Для кожної групи можна задати викладачів, які будуть проводити різні типи занять з відповідного навчального предмета. Забезпечено можливість внесення пари у вільну аудиторію, проводиться перевірка, чи група в цей час не має вже пари і чи вільний викладач. Надано можливість редагувати та видаляти інформацію про внесені в базу пари.

Розроблено модуль, який надає можливість вибрати з бази інформацію про розклад відповідно до заданих користувачем критеріїв пошуку: групи, дні, назви предметів, прізвища викладачів. Для забезпечення цілісності даних перед внесенням будь яких змін у базу даних (командами INSERT, UPDATE, DELETE) виконується перевірка на допустимість проведення таких змін. Існує два механізми оновлення даних у зв’язаних між собою таблицях.

При каскадному оновленні даних, якщо змінюється запис в материнській таблиці, то аналогічно змінюються і записи в дочірніх таблицях, котрі мають зв’язок з записом з материнської таблиці. У цьому випадку при видаленні запису з головної таблиці, відбувається видалення всіх зв’язаних записів з дочірніх таблиць.

При спробі змінити дані ключового поля в дочірній таблиці, зміна допускається лише за умови, що в материнській таблиці, існує запис, котрий відповідає змінюваному в дочірній.

При другому підході, до забезпечення оновлення, забороняється змінювати дані в головній таблиці, якщо в дочірніх таблицях існують пов’язані з даним записом записи. Аналогічно не можна витерти запис з материнської таблиці. В обох випадках не можна вставити запис в дочірню таблицю, якщо йому не існує аналогу в материнській.

У нашій методиці запропоновано і створено веб-інтерфейс, який просто не дозволяє користувачу проводити запити, що можуть порушити цілісність даних. Наприклад, під час вставляння даних у дочірню таблицю заборонено прямий ввід ключового поля і користувачеві надано можливість вибору його значення тільки з допустимих величин. Це реалізовано розміщенням на формі об’єктів вибору, таких, як SELECT, CHECKBOX, RADIOGROUP, елементами котрих є значення ключового поля материнської таблиці.

У випадках, коли описаний вище метод використати не можна, запропоновано перед внесенням змін в дочірню таблицю перевіряти наявність відповідного ключового значення в материнській, і або вивести повідомлення про неможливість внесення даних змін, або спробувати автоматично створити запис в головній таблиці з потрібним ключовим значенням, за потреби запитавши в користувача необхідні додаткові дані.

Наприклад, у нашій системі, щоб вставити запис у таблицю group_teach, тобто задати викладачів, які ведуть заняття з якогось предмета для певної групи, надається можливість вибирати лише тих викладачів, що можуть його читати, тобто ідентифікатор яких внесений у поле idteacher таблиці subject для запису з idsubject цього предмета. При цьому передбачено можливість автоматичного внесення його для запису з idsubject з попереднім запитом у користувача необхідних додаткових даних і подальшим внесенням цих даних у таблицю teachers (рис. 2). Такий підхід не знімає, однак, усіх проблем, пов’язаних з

цілісністю бази даних. У таблиці rozklad також міститься інформація про викладача, котрий проводить пару. Якщо змінити, наприклад, лектора з якогось предмета, то в таблиці rozklad ці зміни не будуть відображатися, що може призвести до внутрішньої суперечливості даних. Тому для уникнення цієї проблеми під час внесення змін до таблиці group_teach з таблиці rozklad знищуються всі змінені дані. Забезпечення цілісності й несуперечливості інформації, яка зберігається в базі даних, є найважливішою задачею під час проектування баз даних та програм, що їх обслуговують. Якщо не забезпечити дотримання цих умов, то в базі даних можливе накопичення непотрібних і недостовірних даних, збої в роботі програмного забезпечення, і рано чи пізно це призведе до краху всієї системи і втрати інформації.

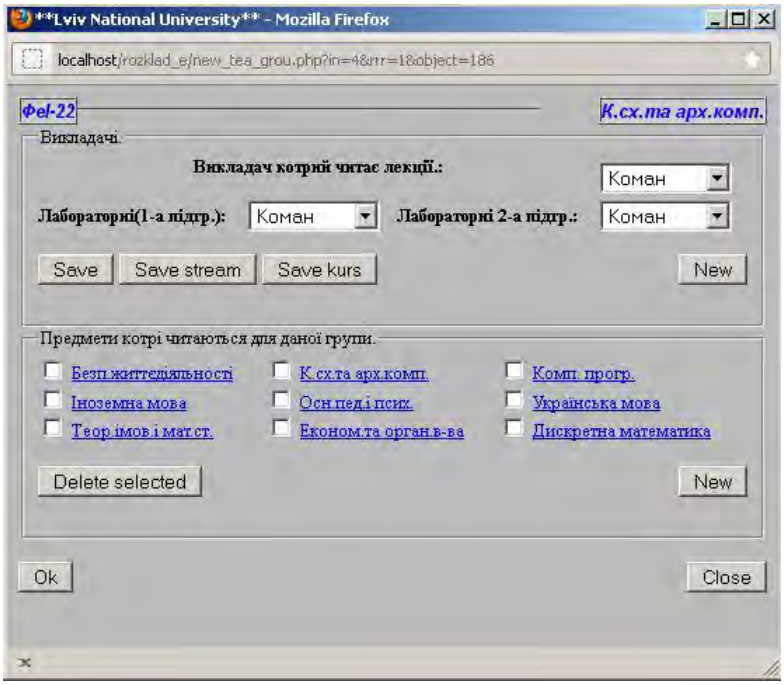

Рис. 2. Веб-інтерфейс зіставлення предметів та викладачів

Якщо змінюється запис у материнській таблиці subject, то аналогічно змінюються і записи в дочірніх таблицях group_teach та rozklad, котрі мають зв'язок із нею. Наприклад, у разі видалення предмета з subject забезпечується видалення всіх записів, пов'язаних з цим предметом у таблицях group_teach та rozklad.

Під час формування розкладу занять інформація вноситься переважно в таблицю rozklad. При цьому передбачено перевірку правильності введення записів. Наприклад, коли змінюється викладач, який веде заняття для певної групи, тобто змінюється значення полів tlesson, tpract1, tprakt2, tlab1, tlab2 у таблиці group_teach для якогось idsubject, то з таблиці rozklad видаляються записи, що стосуються попереднього значення idteach для idsubject цієї групи.

Під час внесення пари відбувається перевірка, чи в таблиці rozklad вже наявні записи з аудиторією, в якій пропонується проводити пару, а також чи викладач не має вже пари в цей час. При цьому враховуються особливості описаних вище параметрів таблиці. Наприклад, аудиторія може бути вільною лише по чисельнику, а по знаменнику зайнята. Тоді вноситься два окремих записи з однаковими параметрами, крім значень чис/знам та idauditory.

При видаленні пари для якоїсь групи враховується, чи не є ця пара спільною парою для інших груп, потоків чи курсів. Якщо є, то в такому випадку коректно видаляються записи про цю пару для всіх груп, для яких вона читається, тобто з таблиці rozklad видаляється для кожної групи свій запис про цю пару.

Розроблення користувацького веб-інтерфейсу є одним з важливих етапів розробки будь-якого програмного продукту. Якою хорошою не була б система, які б необхідні функції вона не виконувала, якщо її веб-інтерфейс є незручним для користувача, той швидко втомлюється під час

роботи з нею, не розуміє, чому виконуються якісь дії, то її використовувати не будуть, а намагатимуться знайти інший продукт, можливо навіть менш досконалий, але з зручнішим веб-інтерфейсом.

Коли веб-інтерфейс є продуманим, функціональним та ергономічним одночасно, користувач, зазвичай, цього не помічає, але коли інтерфейс незручний, то це помічається відразу.

У роботі з базами даних, як зазначалося вище, необхідно забезпечити цілісність та несуперечливість даних. Для реалізації цього завдання використано, зокрема, і можливості користувацького веб-інтерфейсу.

Необхідно також зазначити, що у веб-програміста набагато менший вибір елементів керування, котрі можна використати в розробці форм, ніж у програміста, який розробляє свої продукти під конкретну операційну систему, йому необхідно враховувати, що користувач може використовувати різні браузери, працювати з різними операційними системами. Це все викликає великі труднощі та накладає певні обмеження під час розробки зовнішнього веб-інтерфейсу до бази даних.

Створення користувацького веб-інтерфейсу, призначеного для роботи з базами даних, можна розбити на декілька етапів:

- аналіз запитів, які користувач може створювати, використовуючи ту чи іншу форму;
- вибір елементів керування для форми;
- створення самої форми та розміщення на ній елементів керування.

Розглянемо побудову користувацького веб-інтерфейсу на прикладі створення форми введення розкладу.

Ця форма є основною формою системи, тому доцільно побудувати її так, щоб за її допомогою можна було розв’язувати переважну більшість задач, пов’язаних зі створенням розкладу, а не тільки вводити сам розклад, а саме:

- вибір та можливість створення нової групи;
- вибір дня;
- вибір номера пари;
- вибір типу пари:
- вибір підгрупи;
- вибір частоти проведення пари;
- вибір типу пари (лекція тощо);
- вибір аудиторії;
- вибір та можливість створення запису для нової пари;
- вибір та можливість створення запису для нового викладача;
- можливість редагувати та видаляти наявні записи пар;
- можливість переглядати поточний розклад активної пари.

Робота в цій формі відбувається переважно з таблицею rozklad. При цьому реалізовано перевірку правильності введених користувачем даних з метою уникнення пересилки по мережі наперед некоректних даних (наприклад, коли користувач не вказав пару), тому за допомогою JavaScript організовано перевірки на допустимість введених даних перед відсиланням їх на сервер, де проводиться вже детальніший аналіз на коректність цих даних.

Як уже зазначалося вище, для забезпечення цілісності даних під час їх введення в дочірні таблиці краще користуватися елементами, які надають можливість вибору потрібних значень, аніж дозволяти користувачеві самому вводити ці значення, тим більше, що такий варіант заповнення форми є зручнішим і для самого користувача, бо йому не потрібно нічого згадувати, боятися помилок набору тощо.

Для вибору групи зроблено список з таких посилань, що після вибору одного з них користувачу відразу завантажується розклад певної групи, без будь-яких додаткових операцій з його боку. Це пришвидшує роботу користувача з формою, до того ж стало вже звичним, що після вибору посилання завантажується нова сторінка, тому призначення цього елемента легко зрозуміле.

Все це справедливо також для вибору дня та номера пари, тому зроблено список робочих днів закладу (перелік береться з бази даних) і список допустимих номерів пар.

Ввід атрибутів запропоновано зробити у вигляді вибору одного з можливих положень перемикача radio, щоб відразу було видно всі можливі варіанти вибору. Оскільки цих варіантів небагато, то вони не займають надто багато місця на формі.

У протилежність до цього списки аудиторій, типу пари та самої пари, у зв'язку з тим, що кількість елементів в них більша і, крім того, різна для кожної групи, реалізовано у вигляді випадного списку. Цим самим забезпечується однакове розміщення елементів на формі незалежно від кількості предметів для кожної окремої групи, кількості доступних аудиторій тощо.

Відображення викладача має тільки інформативний характер, тому його можна виводити навіть простим текстом, але оскільки, в принципі, з цієї форми його можна поміняти, краще виводити його в полі редагування. Але користувач може неправильно зрозуміти такий елемент керування, можливо він просто введе прізвище викладача в поле і думатиме (цілком справедливо), що саме це прізвище запишеться в базу даних. Але це не так. Одним зі шляхів вирішення цієї суперечності запропоновано використати випадний список тільки з одним елементом, в якому автоматично стоятиме прізвище того викладача, який читає поточний предмет для цієї групи чи підгрупи.

Відображення поточного розкладу реалізовано у вигляді таблиці з чотирьох комірок, котрі за потреби можуть об'єднуватися і кожна з яких відповідає парі з певними атрибутами. Біля кожної пари розміщено кнопки, за допомогою яких можна вилучити її з розкладу або перевести програму в режим редагування запису цієї пари.

Правильний вибір розмірів і кольору форми, зручне розміщення елементів на ній є одним з найвідповідальніших етапів створення форми.

Вище описано, які елементи мають обов'язково бути розміщеними на формі. Їх, як бачимо, доволі багато. Можна, звичайно, створити велику форму, щоб на ній можна було розмістити всі елементи, але, по-перше, велику форму важче зробити „приємною" для користувача, а по-друге, розміщення на одній формі великої кількості елементів керування робить її незручною для заповнення. Адже для того, щоб заповнити таку форму, потрібно пройти по всіх елементах, а оскільки вони займатимуть у цьому випадку велику площу, доведеться здійснювати рухи мишкою на порівняно великі відстані, що, своєю чергою, спричиняє втому.

Враховуючи вищенаведене, запропоновано процес введення розкладу проводити за два кроки, для чого, відповідно, створено дві форми. У першій формі організовано введення атрибутів та типу пари, а в другій – введення самого предмета та аудиторії. У другій формі забезпечено також можливість контролю та зміни даних, введених на першому кроці. З урахуванням цих міркувань було створено веб-інтерфейс, котрий проводить заповнення бази даних у два кроки. При цьому генеруються форми, зображені на рис. 3 та 4.

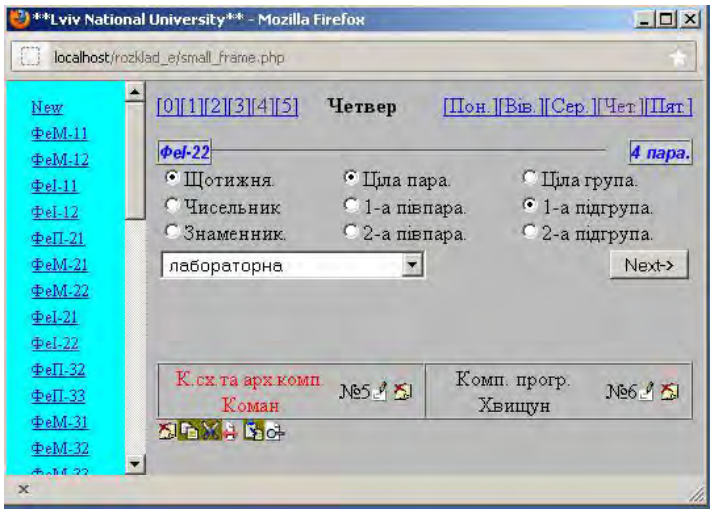

Рис. 3. Форма введення розкладу – перший крок

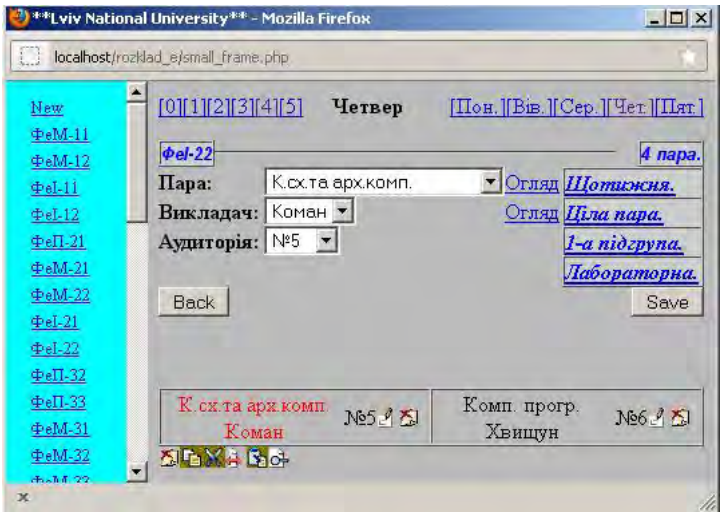

Рис. 4. Форма введення розкладу – другий крок

Під час розробки веб-інтерфейсів для роботи з базою даних розкладу насамперед було враховано особливості створеної бази даних і наявні вимоги до електронної системи розкладу занять для забезпечення зручності користувачів та функціональності. З цією метою розроблено кілька незалежних веб-інтерфейсів. Інтерфейс керування списками викладачів дає змогу додати, редагувати чи видалити інформацію про викладачів, які проводять заняття. Інтерфейс керування навчальними предметами дозволяє додати, редагувати чи видалити інформацію про навчальні предмети. Враховано також можливість інтеграції системи з наявними зовнішніми базами даних викладачів і навчальних предметів. Забезпечено можливість використання розподілених баз даних розкладу, працівників і предметів, які можуть розміщуватися на різних комп'ютерах і редагуватись та супроводжуватись незалежно одна від одної. Веб-інтерфейс забезпечує доступ до цих даних, об'єднання і дає змогу користувачеві ефективно їх використовувати. При цьому для забезпечення швидкодії можна використовувати кешування інформації, яка змінюється нечасто, що істотно збільшить ефективне використання мережевих ресурсів.

Для роботи з веб-інтерфейсами передбачено два типи користувачів. Адміністратори мають можливість вносити та редагувати інформацію про розклад занять через відповідні, доступні тільки їм, веб-інтерфейси. Звичайні користувачі можуть переглядати через веб-інтерфейс інформацію про весь розклад чи якусь частину відповідно до заданих ними критеріїв.

Веб-інтерфейс імпорту навчальної інформації дозволяє внести в базу даних з зовнішніх файлів інформацію про навчальні предмети, прізвища викладачів, доступні аудиторії ти лабораторії. Веб-інтерфейс навчальних дисциплін дає можливість внесення інформації про прізвища викладачів, які ведуть заняття з відповідного навчального предмету та можливі форми проведення (лекційне, практичне, лабораторне заняття). У цьому модулі доступні інструменти для розподілу викладачів по навчальних предметах, внесення інформації про те, для яких курсів, потоків, груп чи підгруп будуть проводитись заняття з цих навчальних предметів. При цьому надано можливість за одну операцію вносити інформацію про заняття для всього курсу чи потоку, що істотно прискорює роботу з наповнення бази даних.

Для формування розкладу занять розроблено окремий веб-інтерфейс, який містить сукупність модулів вибірки груп, аудиторій, викладачів і предметів. Веб-інтерфейс розкладу дозволяє вносити в базу інформацію про пари, надаючи можливість адміністратору внести всі необхідні атрибути пари. Автоматично забезпечується внесення в базу для цієї пари прізвища викладача, який її проводить. Якщо така інформація відсутня, то адміністратор може її внести вручну. Під час формування розкладу автоматично проводиться перевірка на збіг прізвищ викладачів з внесеною інформацією для інших пар та на наявність вільних аудиторій. Для забезпечення неможливості внесення адміністратором некоректних даних запропоновано використовувати перевірку введених даних на етапі внесення ще до відправки даних на сервер. Під час внесення даних адміністратор одразу бачить, як вони будуть

представлені в остаточному вигляді. Передбачена можливість редагування внесених даних. У веб-інтерфейсі розкладу доступні функції копіювання або перенесення вже внесеної інформації про пари, що істотно пришвидшує роботу. Для перегляду розкладу створено веб-інтерфейс представлення розкладу, в якому користувачеві надається інформація про розклад занять у стандартній формі. Користувач може переглянути весь розклад занять або його частину відповідно до заданих критеріїв пошуку. Доступні критерії пошуку: за парами, днями, групами та потоками, навчальними предметами, викладачами та аудиторіями.

Висновки

На основі дослідження запропоновано модель інформаційної системи розкладу занять вищого навчального закладу з доступом через http-протокол. Інформаційна система впроваджена на факультеті електроніки Львівського національного університету імені Івана Франка. Роботу користувацької частини інформаційної системи «Розклад» можна переглянути в мережі Інтернет за адресою http://www.electronics.wups.lviv.ua/rozk/. База даних створена на СКБД MySQL [9], веб-інтерфейс розроблений з використанням мови програмування PHP.

1. Пасічник В.В. Організація баз даних та знань / В.В.Пасічник, В.А.Резниченко. – К.: Видавнича група BHV, 2006. – 384 с. 2. Олійник А.В. Інформаційні системи і технології у фінансових установах / А.В. Олійник, В.М. Шацька: Навч. посібник. – Львів: Новий Світ-2000, 2006. – 436 с. 3. Столярчук О.В. Веб інтерфейси доступу до баз даних / О.В. Столярчук, Р.Я. Шувар, А.М. Продивус // Електроніка та інформаційні технології. – Львів: Львів. нац. ун-т імені Івана Франка, 2011. – Вип. 1. – С. 160–171. 4. Бевз С.В. Розробка автоматичної системи формування розкладу магістратури / С.В. Бевз, к. т. н., доц.; В.В. Войтко, к. т. н., доц.; С.М. Бурбело; А.М. Шоботенко // Інформаційні технології та комп'ютерна техніка: Наукові праці ВНТУ. – 2009. – № 4. 5. Коффман Э. Г. Теория расписаний и вычислительные машины / Э.Г. Коффман. – М.: Наука, 1984. – 336 с. 6. Пайкерс В. Г. Методика составления расписания в образовательном учреждении / В. Г. Пайкерс. –3-е изд. испр. и доп. – М.: АРКТИ, 2001. 7. Пелещишин А.М. Методи та алгоритми моделювання Web-систем // Вісник Держ. ун-ту "Львівська політехніка". – 2000. – № 406. – С. 199–211. 8. Райордан Р. Основы реляционных баз данных; [пер. с англ.]. – М.: Издательско-торговый дом «Русская Редакция», 2001. – 384 с. 9. Ларри Ульман / MySQL; [пер. з англ. А. Слинкин]. – М.: ДМК Пресс СПБ Питер, 2004. – 352 с.