Список використаних джерел:

1. Elastic Revealing Insights from Data (Formerly Elasticsearch) [Електронний ресурс]: [Веб-сайт]. – Електронні дані. – U.S., 2015. – Режим доступу: https://www.elastic. co/ (дата звернення 21.04.2015) – Назва з екрана.

Рушанян Г.М.

студент,
Національний технічний університет України
«Київський політехнічний інститут»

РОЗРОБКА ІНФОРМАЦІЙНОЇ СИСТЕМИ-ДОДАТКА «РОЗКЛАД»

Серед багатьох можливих напрямів розробок підсистем, орієнтованих на мобільні пристрої, найбільш пріоритетним є створення та впровадження підсистеми «Розклад», функціоналом якої буде переважно перегляд розкладу занять навчальної групи, розкладу занять та консультацій викладачів, сповіщення студентів про перенесення занять на інший час, інтеграція з функціоналом системи «Електронний кампус НТУУ «КПІ»« в задачах перегляду методичного забезпечення кредитного модулю, результатів поточного контролю знань, умінь та навичок студентів.

Дані дослідження існуючих підсистеми

Більшість студентів вже давно зіткнулись з проблемою відсутності зручного електронного розкладу занять з підтримкою мобільних платформ.

Дослідження «ринку» показало наявність вже існуючих додатків розкладу занять, але у них знайдені певні недоліки:

1. Веб-ресус «rozklad.kpi.ua» – офіційний сервіс з функціоналом «розклад», котрим користується більшість студентів НТУУ «КПІ». На жаль, цей ресурс не має підтримки мобільних платформ, саме тому не задовільняє потреби користувачів та не є універсальним.

2. Мобільний додаток «KPI weeks» – на перший погляд повністю реалізована ідея, але основна проблема даного сервісу – це неможливість отримувати оновлений адміністрацією або викладачами розклад занять. Як результат – додаток має дуже обмежений функціонал.

За співробітництвом зі структурним підрозділом НТУУ «КПІ» – Конструкторським бюро інформаційних систем (КБ ІС) отримані сервіси, що базуються на реальних даних та забезпечують можливість розробки такого функціоналу.

Цільовими платформами для додатку обрані Android та IOS, так як більшість користувачів використовує саме ці платформи. В майбутньому також запланована підтримка Windows Mobile.

Додатково новий мобільний додаток повинен задовольняти наступним вимогам:

- мати раціональну архітектуру – за рахунок зниження навантаження на запити до серверу як наслідок використання локальної БД,
- реалізувати захист інформації та інформаційних потоків – за рахунок розподілу функціоналу на частину функціоналу загального користування та частину функціоналу, до якого можливий доступ тільки після аутентифікованого входу в систему,
- забезпечити зручність інтерфейсу – за рахунок врахування специфічних особливостей кожної мобільної платформи.

Опис мобільного додатку «Розклад занять»

На практиці для розробки подібних мобільних систем застосовується трирівнева клієнт-серверна архітектура [1]. Архітектура підсистеми «Розклад занять» наведена на рис. 1.

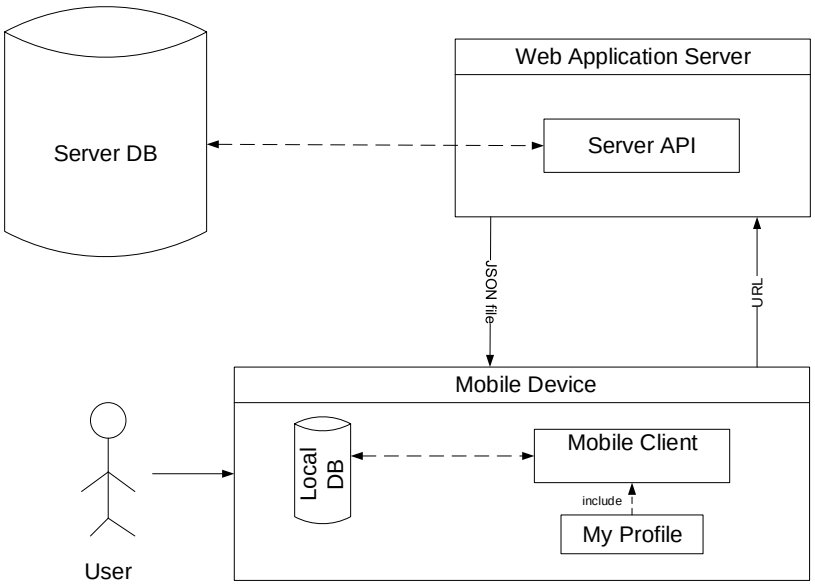

Рис. 1. Архітектура «клієнт-серверного» мобільного додатку

В даному випадку, з бази даних сервера системи «Електронний кампус» дані передаються за допомогою сервера додатків (Web Application Server). Сервер додатків в свою чергу через сервер API (Server API) передає інформацію мобільному пристрою (Mobile Device) у вигляді JSON-об’єкта. Далі, клієнт приймає цей об’єкт та перетворює його у дані потрібної структури і зберігає їх в локальній БД мобільному пристрою. В результаті користувач мобільного пристрою бачить всю необхідну інформацію на дисплеї з використанням запропонованого далі інтерфейса.

В даному випадку, клієнтом є мобільний додаток на платформах Android/IOS. Клієнт відправляє посилання до API функції, яке формується у вигляді текстового посилання до певного методу певного контролера. В цей метод передаються необхідні дані для обробки запиту.

Самі API функції розташовуються на стороні сервера додатків, вони містять логіку запитів до бази даних. При цьому використовується з’єднання

API з ядром, що надає до API набори моделей даних для роботи з даними БД. Безпосередньо з БД робота не проводиться, а звернення здійснюються до моделі Entity Framework.

Крім взаємодії з сервером, мобільний додаток зберігає дані у свою локальну базу даних, а саме – Core Data (IOS)/SQLite (Android). Усі дані, що записані до цієї БД, містяться на телефоні [2].

Таких підхід зумовлений тим, що робота мобільного додатку повина здійснюватись також в умовах відсутності зв'язку з сервером, при цьому буде використовуватись лише частина програмного забезпечення для відображення останніх данних, що були завантажені. Шлях реалізації цього підходу простий – в разі наявності зв'язку з сервером усі дані завантажуються до додатку і одразу дублюються у локальну БД мобільного пристрою. Таким чином створюється кеш, який оновлюється і дає можливість додатку використовувати інформацію навіть в оффлайн режимі [3].

Обрана технологія дозволяє прискорити та покращити роботу з базою даних, а також задовольняє першій вимозі.

В майбутніх версіях буде створена синхронізація пристроїв з iCloud та Google cloud, що дасть змогу синхронізувати інформацію між пристроями та створювати резервні копії.

Список використаних джерел:

1. Alex Berson «CLIENT/SERVER ARCHITECTURE. 2nd edition» // McGraw-Hill. 1996. – 569 с.

2. Jonathon Manning «Swift Development with Cocoa» // O'Reilly. 2014. – 474 с.

3. Jeff McWherter «Professional Mobile Application Development» // Wrox. 2012 – 390 с.

Стасюк В.О.

студент,
Національний технічний університет України
«Київський політехнічний інститут»

ПОДАЧА МАСТИЛЬНО-ОХОЛОДЖУВАЛЬНИХ РІДИН ПІД ВИСОКИМ ТИСКОМ ПРИ ОБРОБЛЕННІ ВАЖКООБРОБЛЮВАНИХ МАТЕРІАЛІВ

В сучасній промисловості дуже часто доводиться обробляти титан, жароміцні супер-сплави, нержавіючі сталі та сталі з низьким вмістом вуглецю. При обробленні таких матеріалів виникають проблеми, зв'язані з їхньою поганою оброблюваністю, або з проблематичним утворенням стружки. Для вирішення цих проблем компанією Sandvik Coromant був розроблений метод оброблення, з подачею мастильно-охолоджувальної рідини (МОР) під високим тиском [1, с. 23].