Зміст

[ВСТУП 6](#_Toc452709844)

[1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ 8](#_Toc452709845)

[1.1 Аналіз стану ринку мобільних сервісів 9](#_Toc452709846)

[1.2 Аналіз програмних засобів створення мобільних додатків 12](#_Toc452709847)

[1.3 Опис мобільного додатку MyELIT 15](#_Toc452709848)

[2 ПОСТАНОВКА ЗАДАЧІ ТА МЕТОДИ ДОСЛІДЖЕННЯ 17](#_Toc452709849)

[2.1 Мета та задачі 17](#_Toc452709850)

[2.2 Вибір методів розробки 17](#_Toc452709851)

[2.3 Планування робіт 18](#_Toc452709852)

[3 Реалізація проекту MyELIT 32](#_Toc452709853)

[3.1 Структура мобільного додатку MyELIT 32](#_Toc452709854)

[2.2 Створення головного меню. 35](#_Toc452709855)

[2.3 Створення розкладу та історії запитів 37](#_Toc452709856)

[2.4 Створення відслідковування розкладу 39](#_Toc452709857)

[2.5 Створення новин 42](#_Toc452709858)

[2.6 Функція зв’язку з деканатом 44](#_Toc452709859)

[2.7 Локалізація додатку 45](#_Toc452709860)

[4 РЕАЛІЗАЦІЯ ІНТЕРФЕЙСУ 47](#_Toc452709861)

[4.1 UI інтерфейс мобільного додатку “MyELIT” 47](#_Toc452709862)

[4.2 UX інтерфейс мобільного додатку “MyELIT” 49](#_Toc452709863)

[ВИСНОВОК 52](#_Toc452709864)

[СПИСОК ЛІТЕРАТУРИ 54](#_Toc452709865)

[ДОДАТОК А. Технічне завдання 56](#_Toc452709866)

[ДОДАТОК Б 64](#_Toc452709882)

[ДОДАТОК В 66](#_Toc452709883)

[ДОДАТОК Г 67](#_Toc452709884)

# 

# ВСТУП

Наука і техніка як напрямки людської діяльності існують віддавна. Однак, якщо в минулих сторіччях проблеми цих областей цікавили лише вузьке коло причетних до них інтелектуалів, а відношення до техніки було лише прикладним, то наш час висунув обидва ці явища в центр суспільної уваги, залучило до них погляди мільйонів людей.

Стрімкий розвиток техніки обумовив її всеосяжний вплив на сучасний світ. Популярність портативної електроніки та мобільних гаджетів у наш час важко заперечити. За новинами сучасного ринку мобільних телефонів дуже тяжко слідкувати, тому що швидкість створення нових технологій та розробок досягла рекордних показників.

У таких умовах дуже популярними стають мобільні додатки та програмне забезпечення і саме популярність тієї чи іншої мобільної платформи чи операційної системи визначає майбутнє розробників. Сьогодні сфера створення мобільних додатків знаходиться на високому рівні розвитку і лише динаміка розвитку популярності розумних гаджетів допоможе перейти їй на новий рівень.

Прикладні програми для систем мобільного зв’язку поділяються на:

* персональні додатки, що забезпечують доступ до інформаційних сервісів;
* системи моніторингу та системи віддаленого управління, браузери, дозволяють завантажувати сторінки у мережі Інтернет;
* мобільні ігри, мережеві багатокористувацькі гри;
* бізнес-додатки, клієнти інформаційних систем масштабу підприємства;
* комерційні веб-додатки (Інтернет-банкінг та інші види віддаленого доступу).

Розробка мобільних додатків – це довготривалий та складний процес у якому спеціалізовані додатки розробляються для невеликих портативних пристроїв, таких як сучасні смартфони або мобільні телефони. Мобільні додатки входять у комплект нового мобільного телефону або ж є адаптованими для швидкісного завантаження за допомогою спеціальних платформ розповсюдження ПО.

Метою дипломного проекту є створення мобільного додатку кафедри комп’ютерних наук для платформи Windows Phone 8.1. Додаток як об’єкт проекту повинен задовольняти всім поставленим функціональним і нефункціональним вимогам. Предметом проекту є задоволення всіх цих вимог, тобто розроблення необхідних сервісів, таких як: розклад, новини тощо.

# АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

Мобільний додаток створено спеціально для кафедри комп’ютерних наук факультету електроніки та інформаційних технологій Сумського державного університету. Розділи додатку розташовані у вірному, інтуїтивному порядку, тобто спочатку користувач знайомиться з кафедрою та спеціальностями, знайомиться з інститутом аспірантури та докторантури.

Додано розділ з розкладом СумДУ. Розклад додатку дозволяє отримати необхідну інформацію щодо пар університету швидше, з меншим використанням інтернет-трафіку, за рахунок вже деякої завантаженої інформації, та в зручному вигляді. Останніні результати розкладу тимчасово зберігаються, що дозволяє їх переглянути без інтернет з’єднання. Ця функція буде корисною як студентам так і викладачам.

Також додано розділ з новинами університету, що дозволяє бути в курсі останніх подій СумДУ.

## Аналіз стану ринку мобільних сервісів

У наш час мобільні пристрої сильно проникли у повсякденне життя кожної людини. Кожна друга людина на Землі користується смартфоном, а кожна третя використовує його, як кишеньковий помічник.

Смартфони – це окрема категорія телефонів, які мають власну оперативну пам'ять і власний процесор. Більшість смартфонів працюють під операційними системами Windows Phone, Android, IOS.

Для того щоб смартфоном було комфортніше користуватись, розробники створюють спеціальні мобільні додатки. Вони можуть виконувати безліч функцій починаючі з різноманітних музикальних плеєрів до серйозного корпоративного програмного забезпечення. У бізнесі – мобільний додаток дозволяє налагодити зв’язок між клієнтами. А це надає можливість постійного інформування клієнтів що до нових пропозицій, тощо. Клієнт також має змогу ознайомитись з вашим бізнесом сидячі у себе вдома зі своїм смартфоном.

Користувач смартфону має можливість завантажити будь-який мобільний додаток у спеціальному магазині Windows Phone Store (рис. 1.1), AppStore або Google Play відповідно до використовуваної операційної системи, але для цього тільки потрібно створити спеціальний аккаунт, який створює вільний доступ до перегляду та завантаження контенту.

Загалом визначають чотири основні типи мобільних додатків:

* Корпоративні. Їх мета – спрощення роботи компанії, швидка передача даних між працівниками або отримання корпоративної інформації. Цільовою аудиторією таких мобільних додатків є, насамперед, працівники вашої компанії, а також реальні та потенційні клієнти та партнери. Розробка мобільних додатків такого типу простіше, для їх дизайну використовуються кольори та елементи корпоративного стилю компанії.

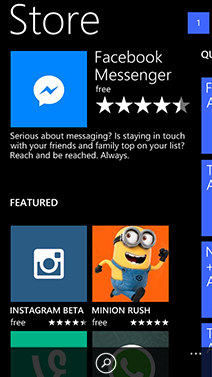


Рисунок 1.1. – Windows Phone Store

* Контентні. Це – мобільні додатки основна мета яких - надавати різного роду інформацію. Текстову або у відео або аудіоформаті. В основному, така розробка мобільних додатків здійснюється для засобів масової інформації, радіо, телеканалів чи порталів.
* Сервісні . Їх мета випливає з назви – надавати певні сервісні послуги, тобто виконувати завдання в реальному часі, які ставить користувач. Розробка мобільних додатків такого типу - складна, адже вони повинні працювати, як годинник: починаючи від калькулятора або будильника і закінчуючи програмами для роботи з великими обсягами тексту або графіки.
* Ігрові. Основне завдання таких мобільних додатків – розважати, але це не означає, що вона єдина. Розробка мобільних додатків такого типу ускладнюється необхідністю вдало розмістити контекстну або пряму рекламу.

На сьогоднішній час ринок мобільних послуг зростає великими темпами, але щоб такий прогрес був і на далі треба виділити основні передумови для його зростання, а саме: використання соціальних мереж, зростання рівня використання планшетних пристроїв, розвиток мобільного банкінгу, електронної комерції і платіжних систем, геопозиціювання, впровадження LTE або 3G тощо.

Перешкодами зростання ринку мобільних додатків є: неінформованість користувачів, труднощі сплати, низький рівень ІТ готовності населення та інші.

Додатки, які вже стали популярними і займають топові місця в “Windows Phone Store” мають явну перевагу над новими додатками, але крім грошей розробникам цікаво знати детальну інформацію про завантаження, коли, де, на якому пристрої та якість обслуговування додатком користувача, для цього існує спеціальна система аналізу для збору та обробки такої інформації, перед тим як назвати такі сервіси, треба розуміти, які базові

функції вона повинна мати, а саме:

• Визначення джерел трафіку (хоча б відслідковування рекламних систем).

• Воронки.

• Повноцінний когортний аналіз.

Когортний аналіз – це виділення певної групи користувачів (когорти) та аналіз її поведінки в часі. Користувачів в когорті об'єднує вчинення дії на певному проміжку часу.

Ідеальний інструмент окрім базових функцій повинен мати наступний функціонал:

• Можливість А/Б тестування в мобільному додатку.

• Можливість відправки таргетувальні пуш-повідомлення.

• Повноцінне API з можливістю вивантажувати сирі данні.

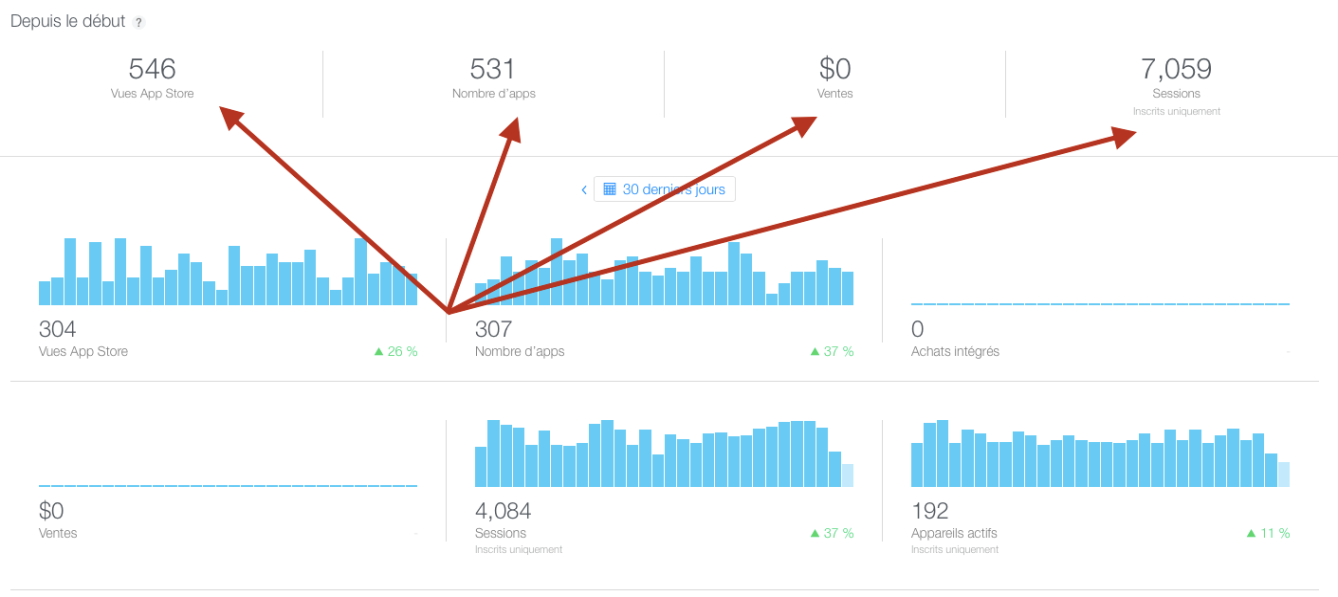
****

Рисунок 1.2. – Детальна система аналізу

На жаль, жоден з існуючих на ринку інструментів не вміє робити все це відразу (хоча деякі близькі, але дуже дорогі), що створює ряд складнощів в аналітиці мобільних додатків, тим паче сервіс для збору аналітики існує і самі сучасніші із них це – Яндекс метрика, Kissmetrics, Kontagent тощо.

## Аналіз програмних засобів створення мобільних додатків

Для розробки додатків для Windows Phone компанія створила свій пакет інструментів для розробки – Windows Phone Software Development Kit (SDK). Корпорація “Microsoft” до останнього тримала SDK для “Windows Phone 8” під замком, але тепер його можна вільно завантажити на офіційному сайті компанії. Відсутність доступу до SDK призвело розробників в деяке здивування, адже в “Windows Phone” завжди була проблема з вибором якісних додатків. В рамках Build Conference компанія “Microsoft” зняла покрови з SDK, і розробники нарешті можуть приступити до створення нових ігор і додатків для “Windows Phone 8.1”.

Для розробки додатку було використане середовище розробки Microsoft Visual Studio 2013 Update 5, яке розповсюджується та підтримується компанією Microsoft. Пакет програм Visual Studio є пропрієтарним продуктом, проте існує безкоштовна версія Community.

Для створення макету сторінок використовувалась декларативна мова розмітки XAML. З точки зору моделі програмування .NET Framework ця мова спрощує створення користувацького інтерфейсу для програми .NET Framework. Можна створити видимі елементи інтерфейсу користувача в декларативній розмітці XAML, а потім відокремити визначення користувача інтерфейсу від логіки часу виконання, використовуючи файли коду програмної частини, приєднані до розмітки за допомогою визначень поділюваних класів. Мова XAML безпосередньо представляє створення екземплярів об'єктів в конкретному наборі резервних типів, визначених у збірках. У цьому полягає її відмінність від більшості інших мов розмітки, які, як правило, являють собою інтерпретовані мови без прямого зв'язку з системою резервних типів. Мова XAML забезпечує робочий процес, що дозволяє декільком учасникам розробляти користувацький інтерфейс і логіку програми, використовуючи потенційно різні засоби.

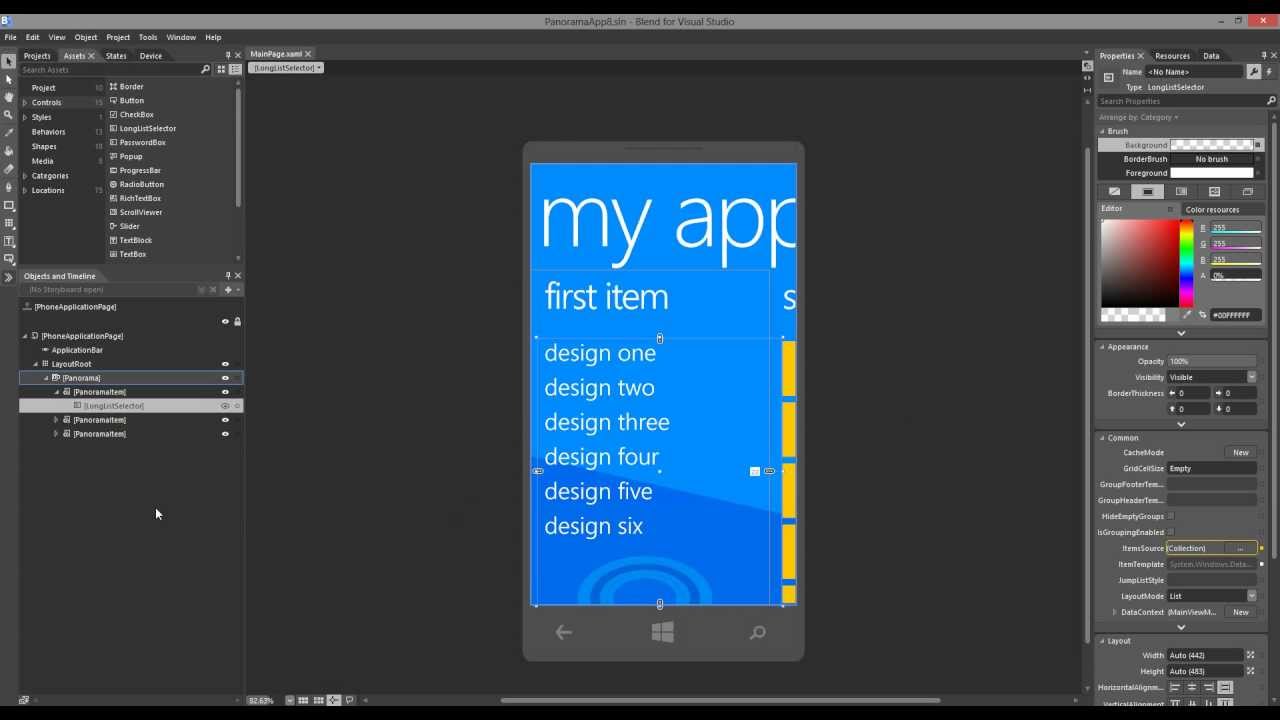


Рисунок 1.3. – Скріншот Expression Blend 2013

Для проектування інтерфейсу був використаний Expression Blend, призначений для розробки дизайну додатків, який входить до складу пакету Microsoft Expression Studio. Цей WYSIWYG - редактор XAML-сторінок дозволяє реалізувати найсміливіші дизайнерські ідеї.

Для створення логіки мобільного додатку була використана об'єктно-орієнтована мова програмування C#. Головною особливістю мови є її орієнтованість на платформу .NET. C# приховує від розробника багато незначних технічних деталей, включаючи операції пов’язані з упаковкою і розпаковкою типів, ініціалізації змінних, збиранням сміття. Це дозволяє C# програмісту приділяти більше уваги змістовій частині коду.

Однією з найбільш значних проблем проектування є складність. Чим більша і складніша програмна система, тим важливіше розбити її на невеликі, чітко окреслені частини. Щоб впоратися зі складністю, необхідно абстрагуватися від деталей. У цьому сенсі ООП являє собою дуже зручну методику проектування.

Класи дозволяють проводити конструювання з корисних компонентів, що володіють простими інструментами, що дозволяє абстрагуватися від деталей реалізації.

Дані та операції над ними утворюють певну сутність, і вони не розносяться по всій програмі, як нерідко буває у випадку процедурного програмування, а описуються разом. Локалізація коду і даних покращує наочність і зручність супроводу програмного забезпечення.

Інкапсуляція дозволяє привнести властивість модульності, що полегшує розпаралелювання виконання завдання між кількома виконавцями і оновлення версій окремих компонентів.

ООП дає можливість створювати системи, що розгортаються. Це одна з основних переваг ООП, і саме вона відрізняє даний підхід від традиційних методів програмування. Розширюваність означає, що існуючу систему можна змусити працювати з новими компонентами, причому без внесення до неї будь-яких змін. Компоненти можуть бути додані на етапі виконання програми.

Часто багаторазового використання програмного забезпечення не вдається домогтися через те, що існуючі компоненти вже не відповідають новим вимогам. ООП допомагає цього досягти без порушення роботи вже наявних компонентів, що дозволяє витягти максимум з багаторазового використання компонентів.

* Скорочується час на розробку, який може бути віддано іншим завданням.
* Компоненти багаторазового використання звичайно містять набагато менше помилок, ніж знову розроблені, адже вони вже не раз піддавалися перевірці.
* Коли якийсь компонент використовується відразу декількома клієнтами, поліпшення, що вносяться до його код, одночасно надають позитивний вплив і на безліч працюючих з ним програм.
* Якщо програма спирається на стандартні компоненти, її структура і призначений для користувача інтерфейс стають більш уніфікованими, що полегшує її розуміння і спрощує використання.

Використання об’єктно-орієнтованого підходу дозволяє значно зменшити кількість помилок за рахунок інкапсуляції та прискорити налагодження програми за рахунок приналежності методів до свого класу. Набагато прискорюється модифікація програми.

## Опис мобільного додатку кафедри комп’ютерних наук СумДУ

Мобільний додаток створений спеціально для додатку кафедри комп’ютерних наук СумДУ факультету електроніки та інформаційних технологій Сумського державного університету. Розділи додатку розташовані у вірному, інтуїтивному порядку, тобто спочатку абітурієнт знайомиться з факультетом, кафедрами та спеціальностями, знайомиться з інститутом аспірантури та докторантури.

Додано розділ з розкладом СумДУ. Розклад додатку дозволяє отримати необхідну інформацію щодо пар університету швидше, з меншим використанням інтернет-трафіку, за рахунок вже деякої завантаженої інформації, та в зручному вигляді. Останніні результати розкладу тимчасово зберігаються, що дозволяє їх переглянути без інтернет з’єднання. Ця функція буде корисною як студентам так і викладачам.

Також додано розділ з новинами університету, що дозволяє бути в курсі останніх подій СумДУ.

# ПОСТАНОВКА ЗАДАЧІ ТА МЕТОДИ ДОСЛІДЖЕННЯ

## Мета та задачі

Метою виконання даної роботи є розробка готового до використання мобільного додатку для Windows Phone. Для реалізації поставленого завдання були виділені наступні задачі дослідження:

* аналіз аналогів;
* розроблення моделі програмного продукту ;
* розробка додатку;
  + розробка інтерфейсу;
  + розробка модулю новин університету;
  + розробка модулю розкладу;
  + розробка інших модулів;
* відладка та тестування;
* оформлення супроводжуючої документації.

## Вибір методів розробки

Для вирішення поставленої задачі будуть використовуватися методи аналізу, моделювання та розробки.

Під час аналізу були проаналізовані існуючі програмні аналоги: функціонал, дизайн інтерфейсу та основні технології, які використовувалися під час їх створення.

Розроблюваний програмний модулю буде програмуватися в середовищі програмування Microsoft Visual Studio 2013. Для розробки дизайну використаний Expression Blend 2013.

## Планування робіт

2.3.1 Ідентифікація мети ІТ-проекту. Мета проекту: розробити мобільний додаток для мобільної платформи Windows Phone 8.1, для того щоб полегшити доступ до корисної інформації, максимально ознайомити користувача з важливою інформацією, щодо кафедри комп’ютерних наук, також ознайомити з навчальним закладом СумДУ, з персоналом кафедри. Для створення проекту є всі необхідні дані та інструментарій для розробки програмних продуктів.

Мобільний додаток має можливість переглядати потрібну документацію через спеціальні розділи. Для того, щоб користувач не мав проблем з використанням мобільного додатка, всі розділи, кнопки керування, створені таким чином, щоб їх було легко використовувати навідь самим звичайним користувачам. Кожна кнопка має коментарі.

Подібний проект є унікальним тому, що не кожен навчальний заклад має подібний мобільний додаток. Досліджуючі IT ринок в Україні можна зробити висновок, що подібні мобільні додатки існують але ти, які направлені на наукову діальяність - дуже мало. Прикладом є мобільний проект КПІ у м. Київ де розробники створили мобільний додаток з невеликими функціями на операційніий системі Android., але проект фактично «загинув» тому, що розробники не оновлювали контент. «MyELIT» це – новий рівень мобільних проектів, проект постійно оновлюється, проект є якісним та корисним. І подальшому проект буде розвиватись завдякі не тільки розробникам, а й факультету ЕлІТ та СумДУ загалом.

2.3.2 Управління вимогами. мобільний додаток MyELIT повинен відповідати поставленим критеріям, які були висунуто у якості функціональних та не функціональних вимог:

* розмір додатку не повинен перевищувати 35 мегабайт;
* мобільний додаток повинен бути безкоштовним;
* мобільний додаток повинен бути без реклами, яка б псувала або заважали комфортній роботі;
* інтерфейс повинен бути інтуїтивно-зрозумілим, повинен підходити для кожного користувача не залежно від полу, року та здібностей;
* усі тексти повинні бути читабельними, кольори - не навантажувати очі як при денній так і вечірній годині;
* надати користувачу доступ до інформації у зручному та правильному контексті до всіх розділів;
* надати користувачу детальний опис дистанційного, заочного та денного форм навчання в СумДУ, ознайомити користувачів з методом навчання на дистанційній формі;
* надавати технічну підтримку усім тим хто її потребує, у мобільному додатку створити усі благо приємні умови для контакту з технічною підтримкою;
* надати можливість перегляду розкладу СумДУ.
* відображення новин університету;
* надання доступу до корисної інформації;
* для моделювання використовувати методологію UML та IDEF0;
* для програмування використовувати Microsoft Visual Studio;
* використовувати мову c#;
* надати можливість користування додатком під сучасною операційною системою Windows Phone 8.1 (RT).

2.3.3 Планування змісту структури робіт ІТ-проекту (WBS). WBS — це ієрархічна структура, побудована з метою логічного розподілу усіх робіт з виконання проекту і подана у графічному вигляді. Це сукупність декількох рівнів, кожний з яких формується в результаті розподілу роботи попереднього рівня на її складові. Елементом найнижчого рівня є група робіт, або так званий робочий пакет (work package)[10].

Схема WBS представлена в додатку Б.(Рис.Б.1)

2.3.4 Планування структури організації, для впровадження  
готового проекту (OBS). Організаційна структура проекту (ОBS) стосується тільки внутрішньої організаційної структури проекту і не зачіпає відносин проектних груп чи учасників з батьківськими організаціями. Будується ОBS аналогічно робочій структурі, а саме:

* на першому рівні відображається організаційна структура як єдиний елемент;
* на другому і нижчих рівнях триває поділ структури на основні організаційні елементи.

Цей процес повторюється до найнижчого рівня — базових робочих груп (змішаних цільових або функціональних), а при реалізації малих проектів — до окремих виконавців. Обсяг робіт для цих найнижчих організаційних рівнів являє собою найнижчі елементи WBS, кожний з яких можна планувати і контролювати як окремі одиниці. Таке саме правило чинне для створення OBS. Кількість рівнів залежить від розміру проекту[11].

Схема OBS представлена в додатку Б.(Рис.Б.2)

2.3.5 Побудова матриці відповідальності (виконавців пакетів робіт). Матриця відповідальності — це графічне визначення того, хто що робить по проекту, або закріплення відповідальності членів проектної команди за виконання окремих елементів проекту і відносини із залученими партнерами. [12]

Матриця відповідальності наведена у додатку (табл. В) та будується на підставі WBS (рис. Б.1) та OBS (рис. Б.2).

2.3.6 Розробка PDM-мережі (розгорнутий вигляд мережевих діаграм Ганта). PDM-мережі складаються з двох типів елементів: робіт, які розташовані у вузлах, та стрілок, які вказують логічні взаємозв’язки між роботами проекту.

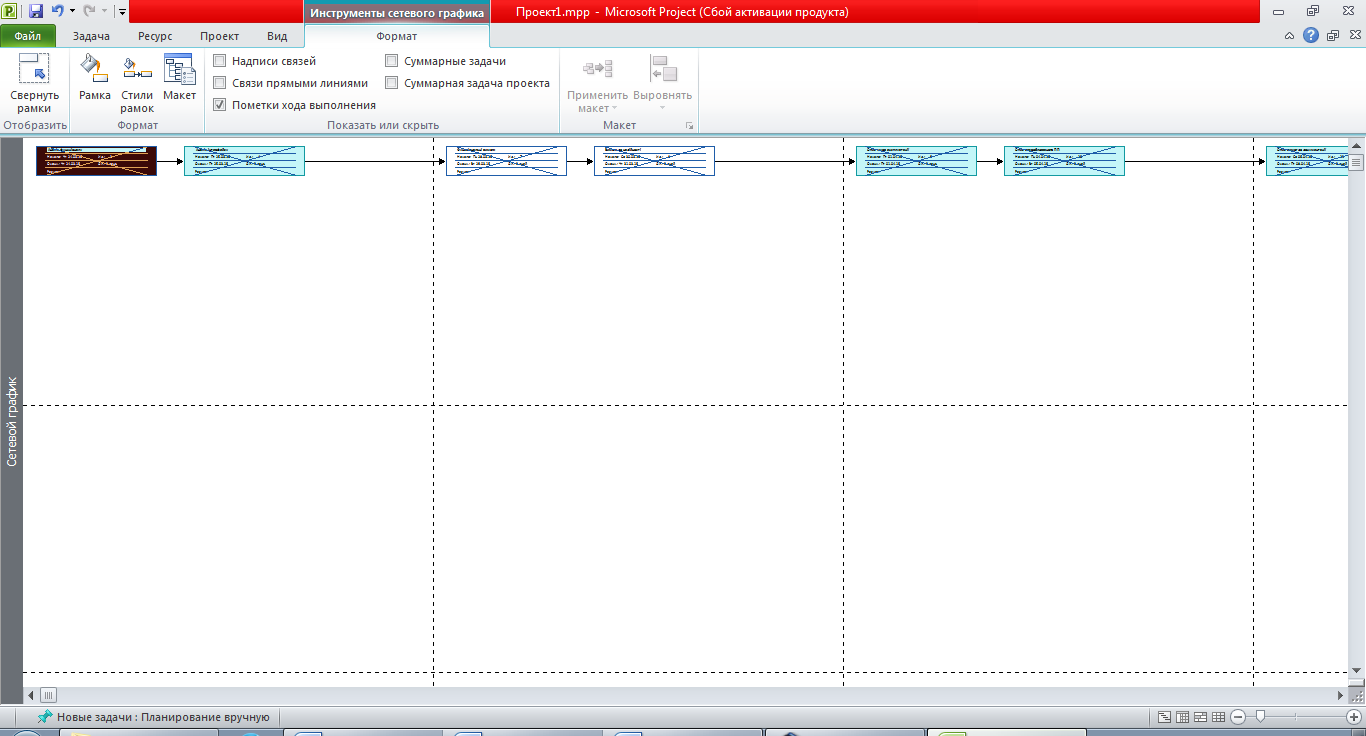


Рисунок 2.1 – PDM-мережа всіх робіт

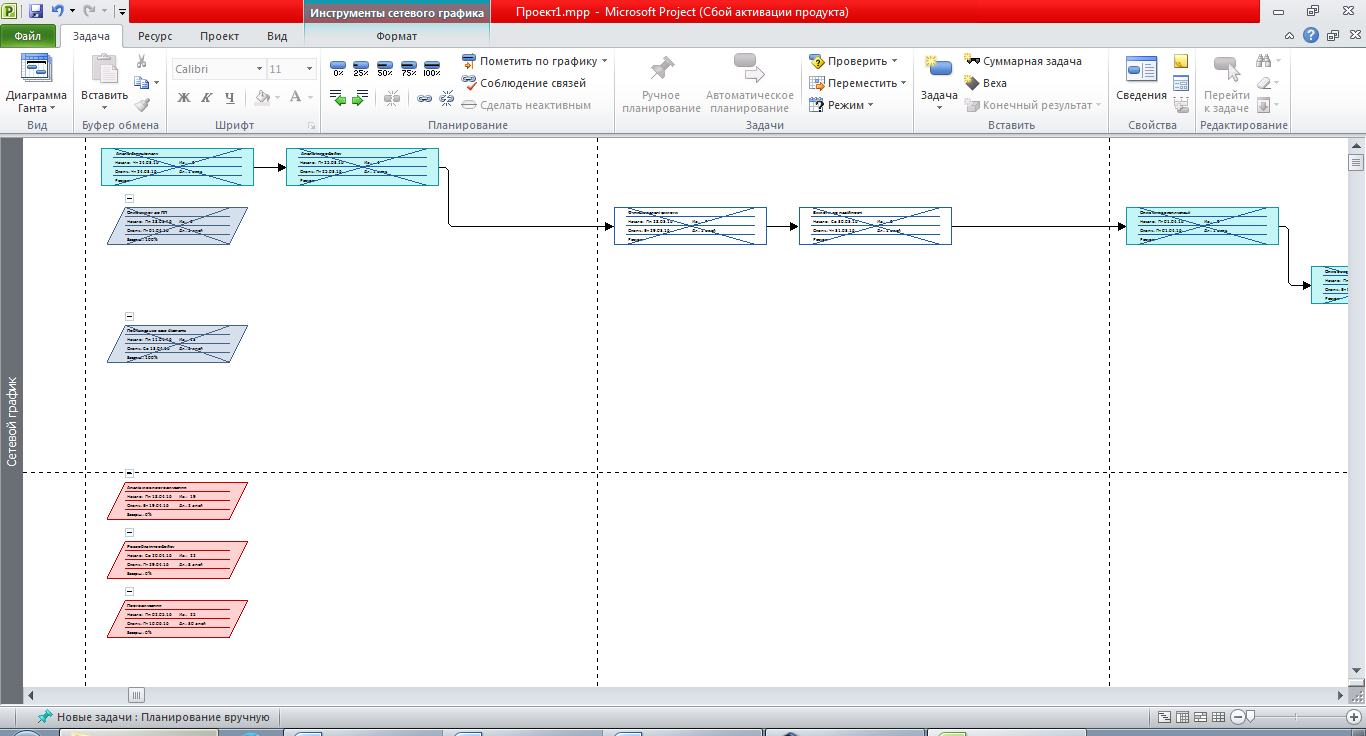


Рисунок 2.2 – PDM-мережа загальних робіт

2.3.7 Побудова календарного графіку виконання ІТ-проекту (включаючи побудову часткових мережевих моделей у вигляді діаграм Ганта). Діаграма Ганта ( англ. Gantt chart, також стрічкова діаграма, графік Ганта) – це популярний тип діаграм, який використовується для ілюстрації плану, графіка робіт за будь-яким проектом. Є одним з методів планування та управління проектами. [13]

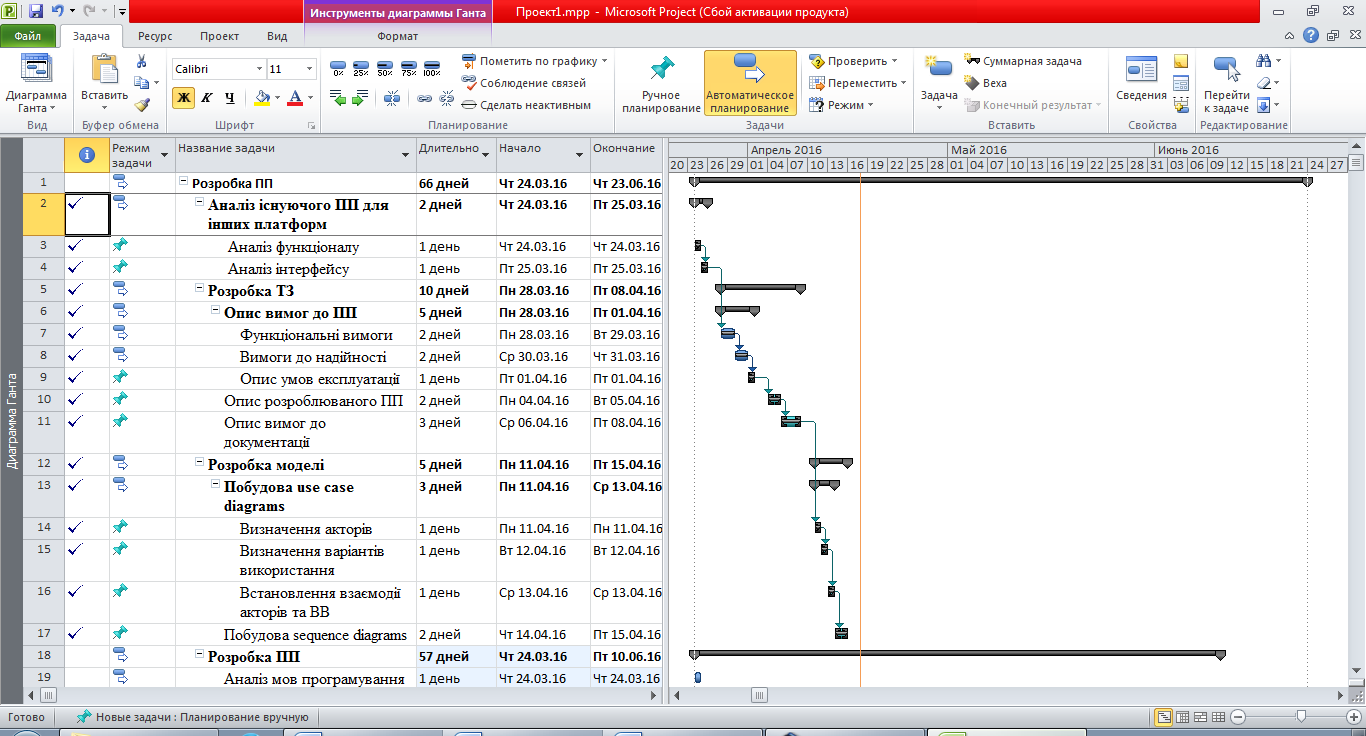


Рисунок 2.3 – Діаграма Ганта в MS Project

Таблиця 2.2 – Детальний перелік робіт

|  |  |
| --- | --- |
| **№** | **Розробка ПП** |
| 1 | Аналіз існуючого ПП для інших платформ |
| 1.1 | Аналіз функціоналу |
| 1.2 | Аналіз інтерфейсу |
| 2 | Розробка ТЗ |
| 2.1 | Опис вимог до ПП |
| 2.1.1 | Функціональні вимоги |
| 2.1.2 | Вимоги до надійності |
| 2.1.3 | Опис умов експлуатації |
| 2.2 | Опис розроблюваного ПП |
| 2.3 | Опис вимог до документації |
| 3 | Розробка моделі |
| 3.1 | Побудова use case diagrams |
| 3.1.1 | Визначення акторів |
| 3.1.2 | Визначення варіантів використання |
| 3.1.3 | Встановлення взаємодії акторів та ВВ |
| 3.2 | Побудова sequence diagrams |
| 4 | Розробка ПП |
| 4.1 | Аналіз мов програмування |
| 4.1.1 | C# |
| 4.1.2 | XAML |
| 4.2 | Розробка інтерфейсу |
| 4.2.1 | Створення pages |
| 4.2.2 | Додавання елементів інтерфейсу |
| 4.3 | Програмування |
| 4.3.1 | Розробка розкладу |

Продовження таблиці 2.2

|  |  |
| --- | --- |
| 4.3.4 | Розробка інших модулів |
| 5 | Тестування |
| 5.1 | Аналіз результатів роботи ПП |
| 5.2 | Перевірка та виправлення помилок |
| 5.2.1 | Виявлення помилок |
| 5.2.2 | Виправлення помилок |
| 6 | Розробка необхідної документації |
| 6.1 | Розробка керівництво користувача |

2.3.8 Розрахунок бюджету проекту. Бюджет проекту повинен враховувати:

* собівартість виконання робіт;
* резерв на відомі ризики (виявлення на етапі планування ризиків;
* базову вартість;
* резерв на ризики, що не були виявлені на етапі планування.

Першочергові витрати – це витрати на закупівлю робочого місця, а саме програмне та апаратне забезпечення для розробки. Ці витрати мають одноразовий характер.

Загальна вартість апаратного забезпечення становить 8000 грн.

Вартість ліцензійного Visual Studio 2013 – 2000$.

Включити у витрати можна також вартість електроенергії.

Згідно актуальної ціни за електроенергію в Україні на роботу апаратного забезпечення буде затрачено близько 50 грн, враховуючи зовнішнє освітлення.

Такий фактор як орендна плата за приміщення для роботи не є оплачуваним, адже розробник працює у себе вдома.

Через не закладені ризики, які можуть вплинути на розробку проекту, бюджет може змінитися.

2.3.9 Планування можливих відхилень (графіки поставок, укладання контрактів тощо). Фактори, які можуть призвести до відхилень поділяються на декілька групи:

* часові (недотримання календарного плану);
* технічні (відмова техніки);
* фінансові (перевищення бюджетних розтрат проекту).

Якщо дані фактори проявляться у подальшій розробці проекту, то вони будуть подолані наступним чином:

* часові – за рахунок найняття нового персоналу та використання часового резерву;
* технічні – за рахунок залучення резервних коштів на покупку потрібного технічного забезпечення;
* фінансові – за рахунок залучення нових інвестицій.

2.3.10 Проведення оцінки ефективності проектних дій (NPV - чиста поточна вартість, IRR – ставка внутрішньої прибутковості). Для того щоб визначити ефективність проектних дій необхідно розрахувати [14]:

* чисту поточну вартість (NPV);
* ставку внутрішньої прибутковості (IRR).

Для початку розрахуємо дисконтний грошовий потік за формулою:

(2.1)

потік платежів після провадження системи: за перший рік прибуток становитиме , за другий рік , за третій ;

– ставка дисконтування, яка дорівнює – 0.2;

Тепер розрахуємо чисту поточну вартість:

(2.2)

розмір інвестиції, дорівнює –

Тепер розрахуємо для ставки дисконтування :

;

;

Тепер розрахуємо ставку внутрішньої прибутковості :

(2.3)

Оскільки , то проект можна вважати прийнятним.

2.3.11 Управління ризиками ІТ-проекту. Класифікація ризиків проекту зазвичай проводиться на етапі їх аналізу за такими критеріями як імовірність виникнення та ступінь впливу в рамках на проект. Але досить доцільним є створення початкової класифікації ризиків ІТ-проектів, яка слугуватиме вхідною інформацією для процесу їх ідентифікації. В даному напрямку вже проведено немало аналітичної роботи вітчизняними та зарубіжними спеціалістами. Серед всіх опублікованих варіантів класифікації ризиків як в загальному плані так і зокрема в ІТ-індустрії є досить багато спільного. Перш за все всі ризики прийнято розділяти за джерелом виникнення на внутрішні та зовнішні. Внутрішні ризики залежать безпосередньо від внутрішнього середовища проекту та мають відносно високий рівень керованості. Зовнішні ризики виникають незалежно від проектної діяльності, можуть враховуватися учасниками проекту та бути частково керованими. Зовнішні ризики можна якісно розділити на дві підгрупи – передбачувані та непередбачувані.

Таблиця 2.3 – Ризики проекту

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Ризик** | **Ефект** | **Вирішення** |
| 1. | Несправність обладнання. Під час роботи над проектом може відмовити технічне забезпечення (ноутбук або його компоненти). | Витрата часу | Залучення (резервування) обладнання або грошей, збереження проміжних результатів проекту |
| 2. | Відмова програмного забезпечення. | Витрата часу | Збереження проміжних результатів проекту |
| 3. | Проблеми з електропостачанням. | Витрата часу та якості | Забезпечення електрогенераторами |
| 4. | Нестача часу для реалізації всього запланованого функціоналу. | Невиконання ТЗ | Резервування часу |
| 5. | Нечітка постановка ТЗ. | Невиконання ТЗ | Детальний опис ТЗ, обговорення «проблемних» компонентів. |
| 6. | Зміна ТЗ | Витрата часу та якості | Укладання договору щодо штрафів при зміні ТЗ. |
| 7. | Хвороба розробника | Витрата часу та якості | Залучення (резервування) додаткового розробника |
| 8. | Інші (непроектні) проблеми персоналу, залученого до проекту | Витрата часу та якості | Залучення (резервування) додаткового персоналу |
| 9. | Критичні баги, які унеможливлюють подальшу роботу над проектом | Витрата часу | Детальне планування робіт проекту |
| 10. | Державна нестабільність (початок революції, державного перевороту ,війни тощо) | Неможливість персоналу приймати подальшу участь у проекті | Залучення (резервування) обладнання та персоналу на територіально віддалених регіонах. |
| 11. | Природні катастрофи (урагани, повені тощо) | Неможливість персоналу приймати подальшу участь у проекті | Залучення (резервування) обладнання та персоналу на територіально віддалених регіонах. |

**Розрахунок індексів ризиків**

Ймовірність виникнення:

1. Слабоймовірні
2. Малоймовірні
3. Ймовірні
4. Досить ймовірні
5. Майже можливі

Величина втрат:

1. Мінімальні
2. Низькі
3. Середні
4. Високі
5. Максимальні

Ступінь впливу:

1. Ігноровані (R=(1:4))
2. Незначні (R=(5:8))
3. Помірні (R=(9:10))
4. Суттєві (R=(12:16))
5. Критичні (R=(20:25))

Рівень ризику:

1. Прийнятні (R=(1:4))
2. Виправдані (R=(5:10))
3. Недопустимі (R=(12:25))

Оцінка індексу ризику:

де R — індекс ризику (бали); — вірогідність виникнення ризиків згідно класифікації (бали); — величина втрат відповідно до класифікації ризику (бали).

Таблиця 2.4 – Розрахунок ступеню впливу ризиків проекту

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **№** | **Ризик** | **Ймовірність** | **Втрати** | **Ступінь впливу (R)** |
| 1. | Несправність обладнання. Під час роботи над проектом може відмовити технічне забезпечення (ноутбук або його компоненти). | 1 | 4 | 4 |
| 2. | Відмова програмного забезпечення. | 2 | 4 | 8 |
| 3. | Проблеми з електропостачанням. | 2 | 4 | 8 |
| 4. | Нестача часу для реалізації всього запланованого функціоналу. | 3 | 4 | 12 |
| 5. | Нечітка постановка ТЗ. | 2 | 3 | 6 |
| 6. | Зміна ТЗ | 1 | 4 | 4 |
| 7. | Хвороба розробника | 2 | 2 | 4 |
| 8. | Інші (непроектні) проблеми персоналу, залученого до проекту | 3 | 3 | 9 |
| 9. | Критичні баги, які унеможливлюють подальшу роботу над проектом | 3 | 3 | 9 |
| 10. | Державна нестабільність (початок революції, державного перевороту ,війни тощо) | 1 | 5 | 5 |
| 11. | Природні катастрофи (урагани, повені тощо) | 1 | 5 | 5 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Ймовірність | 5 | 5 | 10 | 15 | 20 | 25 |
| 4 | 4 | 8 | 12 | 16 | 20 |
| 3 | 3 | 6 | 9 | 12 | 15 |
| 2 | 2 | 4 | 6 | 8 | 10 |
| 1 | 1 | 2 | 3 | 4 | 5 |
|  | 1 | 2 | 3 | 4 | 5 |
|  | Втрати | | | | | |

Рисунок 2.4 – Матриця «Ймовірність – Втрати»

Таблиця 2.5 – Класифікація за рівнем ризику

|  |  |
| --- | --- |
| **R** | **Опис ідентифікованого ризику** |
| **Ігноровані (R=(1:4))** | |
| 4 | Несправність обладнання |
| 4 | Хвороба розробника |
| 4 | Зміна ТЗ |
| **Виправдані (R=(5:16))** | |
| 5 | Державна нестабільність |
| 5 | Природні катастрофи |
| 6 | Нечітка постановка ТЗ |
| 8 | Відмова програмного забезпечення |
| 8 | Проблеми з електропостачанням |
| 9 | Інші (непроектні) проблеми персоналу, залученого до проекту |
| 9 | Критичні баги, які унеможливлюють подальшу роботу над проектом |
| 12 | Нестача часу для реалізації всього запланованого функціоналу |

Продовження таблиці 2.5

|  |  |
| --- | --- |
| **Недопустимі (R=(17:25))** | |
| – | – |

Таблиця 2.6 – Класифікація за ступенем впливу

|  |  |
| --- | --- |
| **R** | **Опис ідентифікованого ризику** |
| **Прийнятні (R=(1:4))** | |
| 4 | Несправність обладнання |
| 4 | Хвороба розробника |
| 4 | Зміна ТЗ |
| **Незначні (R=(5:8))** | |
| 5 | Державна нестабільність |
| 5 | Природні катастрофи |
| 6 | Нечітка постановка ТЗ |
| 8 | Відмова програмного забезпечення |
| 8 | Проблеми з електропостачанням |
| **Помірні (R=(9:10))** | |
| 9 | Інші (непроектні) проблеми персоналу, залученого до проекту |
| 9 | Критичні баги, які унеможливлюють подальшу роботу над проектом |
| **Суттєві (R=(12:16))** | |
| 12 | Нестача часу для реалізації всього запланованого функціоналу |
| **Критичні (R=(20:25))** | |
| – | – |

# Реалізація проекту MyELIT

## Структура мобільного додатку MyELIT

Перші версії додатку MyELIT були створені у 2014 році під операційні системи Android та Apple iOS (рис. 2.1). Проте версії для Windows Phone не існувало. Тому було прийняте рішення про створення додатку для цієї платформи.

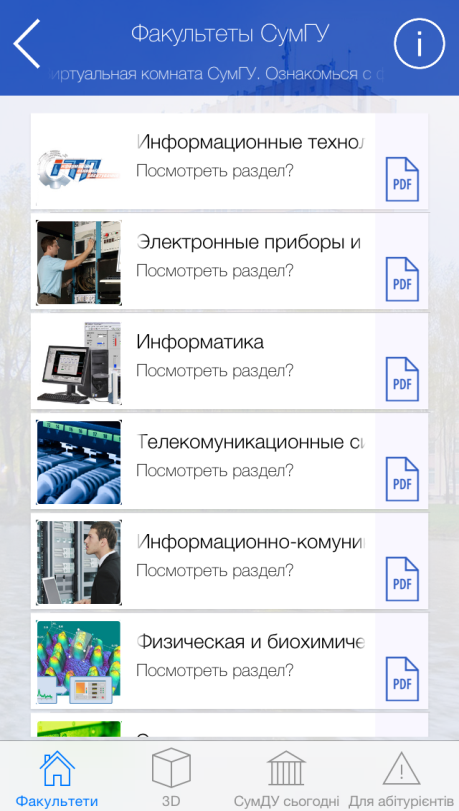


Рисунок 3.1 – Версія 1.0 мобільного додатку MyELIT  
для платформи Apple IOS

Версію додатку для платформи Windows Phone створено з використанням нативних інструментів розробки. Структура додатку складається з власної архітектури відповідно сайту факультету ЕлІТ та кількох додаткових розділів.

Мобільний додаток складається з шести інформаційних розділів, шести сервісних та одного розважального (рис. 3.2), які розміщені на основному екрані. Майже в кожному розділі існують підрозділи.

Структура додатку:

* розклад
* мій розклад
* історія запитів розкладу
* налаштування розкладу
* новини факультету
* новини університету
* радіо
* зв’язок з деканатом
* подзвонити
* написати на пошту
* факультет
* загальна інформація
* структура факультету
* деканат
* абітурієнтам
* спеціальності факультету
* як отримати елітну освіту?
* мультимедійний еліт
* правила прийому
* студентам
* денна форма навчання
* кредитно-модульна система
* індивідуальна підготовка
* регламент дисциплін
* навчання поза університетом
* студентське життя
* профспілкова діяльність
* дозвілля
* наукова діяльність
* аспірантура і докторантура
* наукові спеціальності
* звітність
* захист дисертацій
* журнали та конференції
* журнал нано-та електронної фізики (jnep)
* конференції NAP
* конференція AIST
* конференція факультету ЕлІТ
* міжнародна діяльність
* подвійні дипломи
* європейські стипендіальні програми
* eacea
* практична діяльність
* співпраця з іт-компаніями
* наукові та практичні напрацювання
* працевлаштування

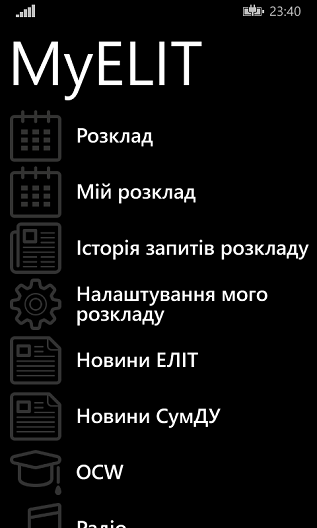


Рисунок 3.2. – Головне меню додатку

## 2.2 Створення головного меню.

Головне меню – це доступ до контенту, який знаходиться у мобільному додатку. Головне меню додатку складається як вже це раніше було сказано з тринадцяти основних розділів, але це лише структура, але як розроблена головне меню?

Спочатку був створений XAML-макет сторінки головного меню. Головним елементом є ListView, розділи є ListView елементами. Кожна сторінка (page) Windows Phone додатку має асоційований з нею .cs файл з кодом на C#, для створення бізнес логіки додатку. Отже для створення пунктів головного меню створюємо ObserverableCollection, наповнюємо колекцію необхідними даними та «зв’язуємо» з створеним раніше ListView у XAML. Далі створюємо обробник події при натисканні на будь-який елемент ListView. При спрацьовуванні він визначає, який з пунктів меню був обраний та викликає необхідну нову сторінку.

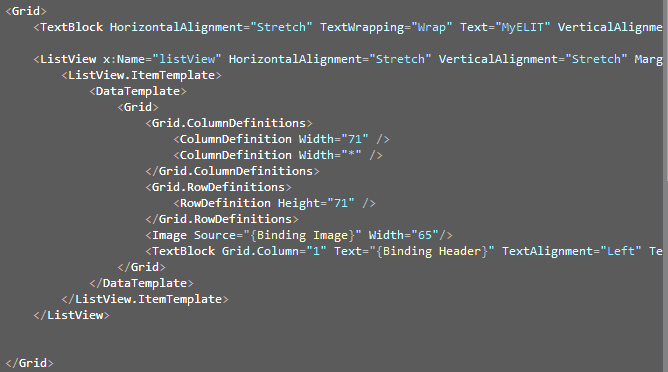


Рисунок 3.3 – XAML-макет сторінки головного меню

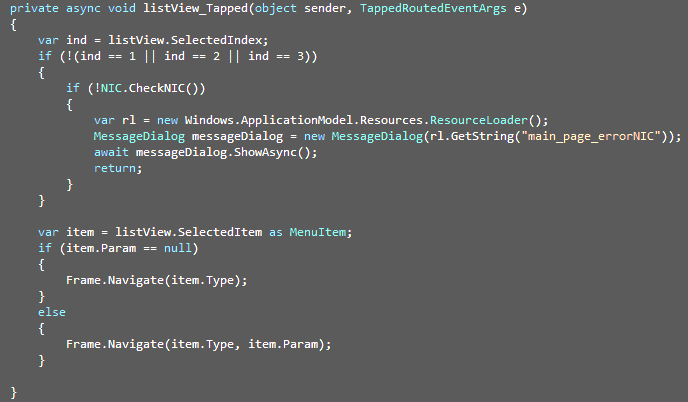


Рисунок 3.4 – Створення пунктів меню та оброблювача події

## 2.3 Створення розкладу та історії запитів

Однією з функціональних вимог проекту є створення розкладу СумДУ. Сервер розкладу має API, за допомогою якого можна отримати інформацію про пари у форматі JSON.

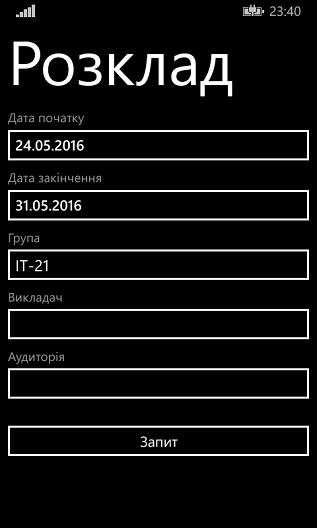


Рисунок 3.5 – Скріншот розкладу

XAML-розмітка цієї сторінки має два DatePicker, для вибору дат початку і закінчення розкладу та три ComboBox для вибору групи або/та викладача або/та аудиторії. Списки викладачів, груп та аудиторій напередодні завантажуються з сервера, створюється масиви з необхідними даними та записується до файлів. Отже елементи ComboBox-ів зчитуються з файлів, що дозволяє скоротити використання Інтернет-трафіку та збільшити швидкодію, що в свою чергу є корисним для смартфонів.

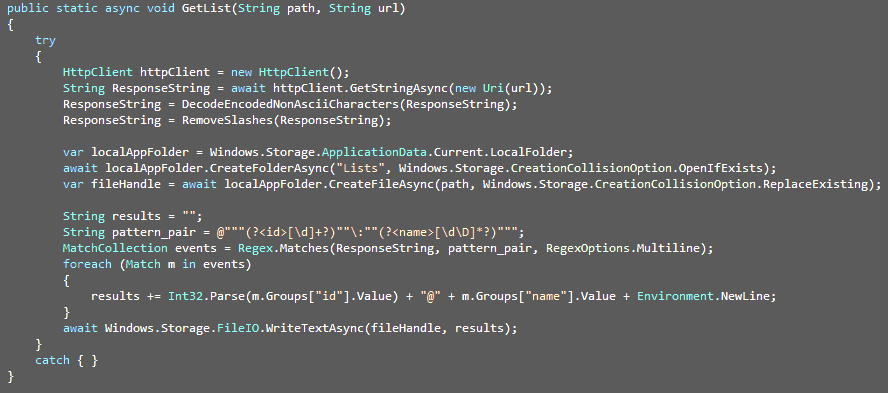


Рисунок 3.6 – Код для завантаження списків

Після обирання користувачем цікавих йому параметрів розкладу та натиснення кнопки «Запит» обробник події завантажує з сервера файл у форматі JSON, а синтаксичний аналізатор (парсер) перетворює дані у двомірний масив днів, де кожен день містить пари. Цей масив починає асинхронно зберігатися у файл та виступає у якості агрументу функції для переходу на нову сторінку, де й буде відображений.

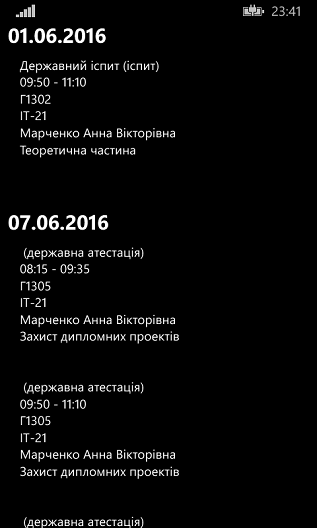


Рисунок 3.7 – Приклад відображення розкладу

Результат кожного запиту зберігається, що дозволяє переглянути розклад без Інтернет-з’єднання. Доступ до історії запитів можна отримати з головного меню обравши пункт з одноіменною назвою.

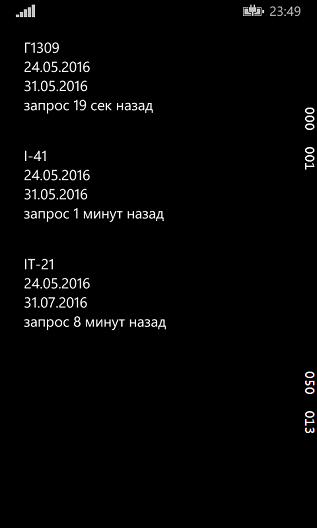


Рисунок 3.8– Приклад відображення історії запитів

## 2.4 Створення відслідковування розкладу

Додаток MyELIT має можливість автоматично відстежувати групу або викладача та відображати його розклад у «живій» плитці на стартовому екрані. Для реалізації задачі перш за все необхідно створити фонове завдання в телефоні (Background Task), оскільки оновлення «живої» плитки має відбуватися без участі користувача чи запущеного додатку. Для цього додаємо у рішення (solution) новий проект Background Task. Створюємо фоновому завданню обробник події при запуску.

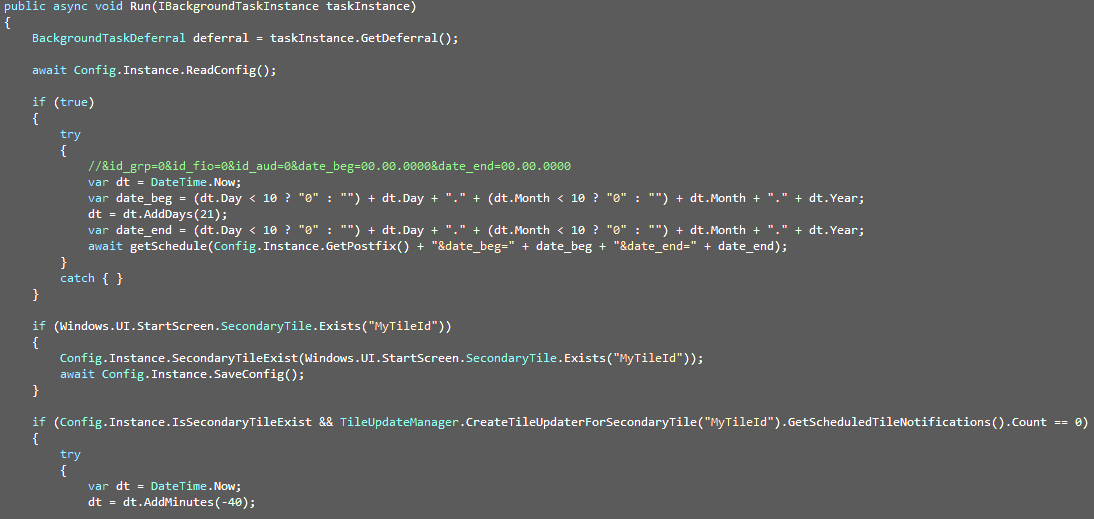


Рисунок 3.9– Обробник події при запуску фонового завдання

У обробнику реалізуємо наступні задачі:

* завантаження останнього розкладу з сервера;
* оновлення «живої» плитки;
* видалення непотрібних файлів рокладу.

Background Task перш за все виконує перевірку наявності Інтернет-з’єднання та дати і часу останнього оновлення файлів розкладу. Якщо умови задовольняють вимоги виконується завантаження нового розкладу.

Після цього фонове завдання намагається оновити «живу» плитку. Плитці можна задати затримку в оновленні, що дозволяє відобразити актуальну пару саме в необхідний заданий момент, а не лише під час роботи фонового завдання. Згідно політики Microsoft фонове завдання не працює постійно і може бути запущене не частіше одного разу за п’ятнадцять хвилин або за умови виконання якоїсь умови (зміна часового поясу, користувач почав використовувати смартфон, користувач закінчив використання смартфону тощо). Background Task отримує об’єкт «живої» плитки за її ідентифікатором та перевіряє наявність запланованого оновлення у черзі. Якщо черга порожня відбувається пошук найближчої пари за спеціальним алгоритмом. Ця пара додається до черги оновлення. Крім того під час пошуку алгоритм видаляє файли, які є неактуальними, тобто містять застарілий розклад.

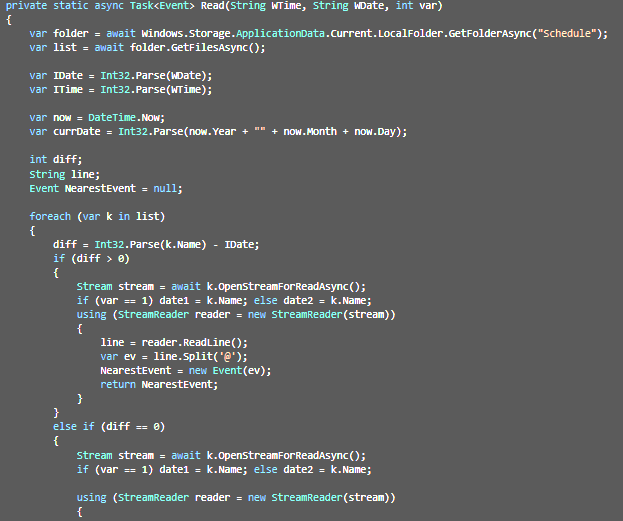


Рисунок 3.10–Код для пошуку найближчої пари

Оскільки фонове завдання та основна програма не «спілкуються» безпосередньо інформація щодо відслідковуваної групи або викладача зберігається у файлі конфігурацій. Сконфігурувати розклад розклад можна за допомогою відповідного пункту головного меню.

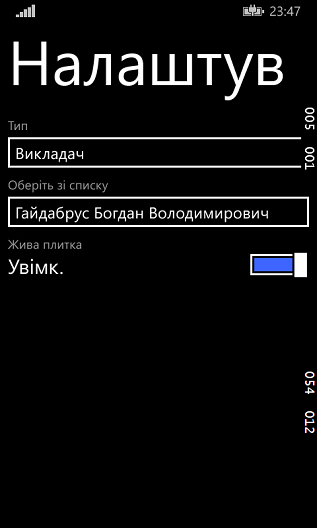


Рисунок 3.11–Приклад налаштування розкладу

Відображення розкладу на «живій» плитці є опціональним, свій розклад можна також переглядати, натиснувши на «Мій розклад» у головному меню.

## 2.5 Створення новин

Спеціально для відображення новин у додатку було створено два синтаксичні аналізатори (парсери), для отримання новин. Новини університету можна отримати використовуючи RSS. Синтаксично проаналізувавши файл отримуємо масив з новинами, який передаємо у якості аргументу у фунцію виклику сторінки з новинами. Новини факультету аналізуються безпосередньо з html-сторінки сайту факультету ЕлІТ.



Рисунок 3.12– Фругмент коду парсеру новин університету

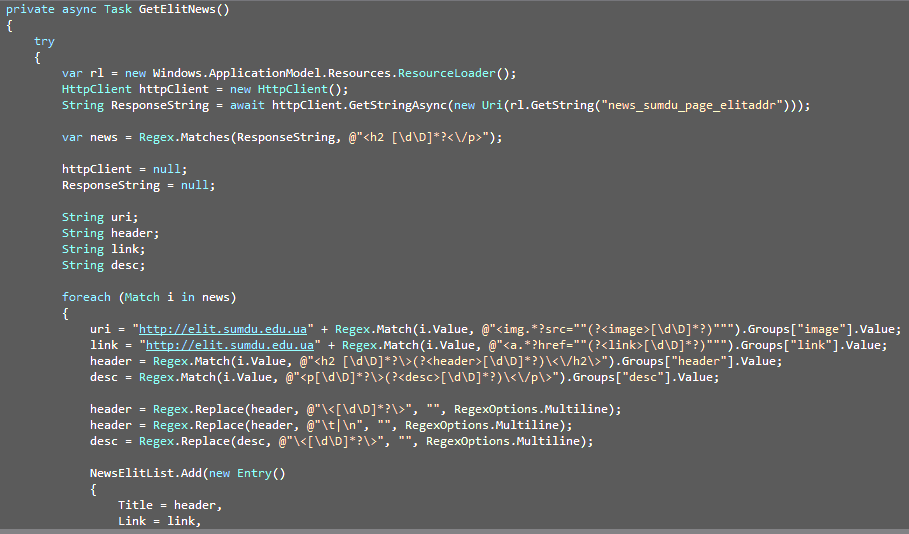


Рисунок 3.13– Фругмент коду парсеру новин факультету ЕлІТ



Рисунок 3.14–Приклад вигляду новин

## 2.6 Функція зв’язку з деканатом

Тепер не треба шукати телефонні номери та почтові адреси і гадати, як зателефонувати або написати листа до приймальної комісії або деканату якщо виникли питання, ці проблеми вмить вирішує мобільний додаток MyELIT. За допомогою спеціальних функцій абітурієнт має можливість не тільки зателефонувати а й написати на пошту. Це дуже комфортно коли біля тебе немає комп’ютера та швидкісного інтернету, завдяки спеціальній бібліотеці MFMailComposeViewController виконується відправка листа (рис. 2.31), а за допомогою методу UIApplication можна створити телефоний дзвінок на будь який номер (рис. 2.32), але як це працює?

Розглянемо спочатку відправку листів, працює вона дуже просто, до нашого класу треба підключити бібліотеку MFMailComposeViewController, яка доречи входить до програмного середовища, далі написати такий код:

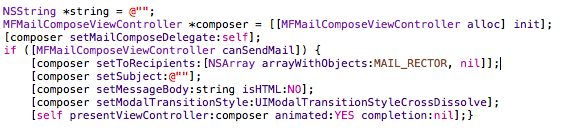


Рисунок 3.15 – Структура відправки листів через бібліотеку

Розглянемо поточний код. MFMailComposeViewController додається до пам’яті смартфону, далі створюється перевірка, якщо контролер відповідає тоді користувач має можливість відправити листа, за допомогою функції setToRecipients можна добавити один, а то й більше електроних адрес на, які будемо відправляти листи, setSubject назва листа, в нашому випадку там нічого немає, тобто ми даємо змогу написати свій заголовок користувачу.

Завершує структуру коду setMessageBody ця - функція дає змогу використовувати HTML обкладинку для створення форми відправлення але вона нам не потрібна, setModalTransitionStyle функція вказує, що після відправлення або відміни вікно з набором тексту повинно зникнути, якщо ж цього не зробити – користувач просто не зможе повернутись до мобільного додатку, та на останок [self presentViewController:composer animated:YES completion:nil] говорить о тому, що вікно закриється з використанням анімацій.

## 2.7 Локалізація додатку

Додаток локалізовано на три мови: українську, російську та англійську. Мова обирається в залежності від налаштувань системи. Для локалізації додатку було створено три файли ресурсів з локалізованими змінними, які можна «прив’язавати» як за допомогою XAML так і за допомогою C#.

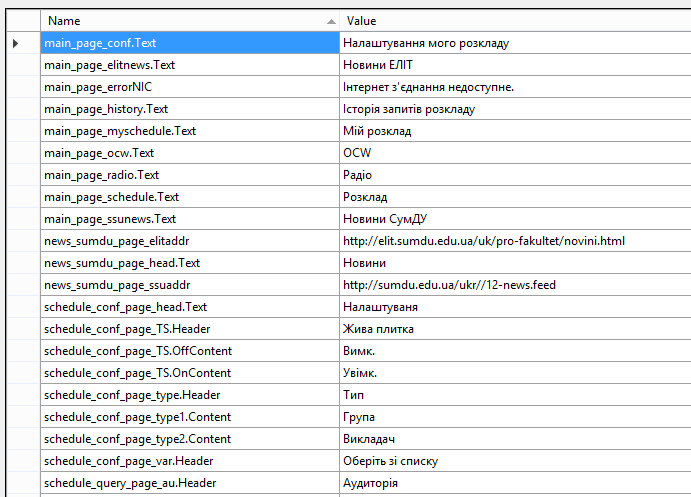


Рисунок 3.16 – Приклад файлу ресурсу

# РЕАЛІЗАЦІЯ ІНТЕРФЕЙСУ

## 4.1 UI інтерфейс мобільного додатку “MyELIT”

У третьому розділі дипломного проекту було розглянуто програмну концепцію створення мобільного додатку, тепер розглянемо основну оболонку, саме те, що користувач бачить має можливість натиснути або потягнути. Як правило оболонка майбутнього мобільного додатку створюється завжди першою, тому що розробникам потрібно бачити, як буде виглядати додаток, які елементи повинні бути щоб їх у подальшому реалізувати програмно.

UI (скорочено) або (User Interface) – сукупність засобів для обробки та відображення інформації, максимально пристосованих для зручності користувача, у графічних системах інтерфейс користувача реалізовується багато віконним режимом, змінами кольору, розміру, видимості (прозорість, напівпрозорість, невидимість) вікон, їхнім розташуванням, сортуванням елементів вікон, гнучкими налаштовуваннями як самих вікон, так і окремих їхніх елементів (файли, папки, ярлики, шрифти тощо), доступністю багатокористувацьких налаштувань. Користувацький інтерфейс створюють через спеціальні програмні продукти такі як: “CorelDRAW” та Expression Blend тощо. Дизайн мобільного додатку “MyELIT” було створено за допомогою програмного продукту “ CorelDRAW Graphics Suite X8” (рис. 3.1). Створення такого дизайну як правило починається з технічного завдання для дизайнера, якщо ж технічне завдання вже існує – можна починати працювати.

Дизайн MyELIT для Widows Phone 8.1 має на меті представити додаток як нативний. Саме тому він наслідував UI та UX платформи. Це дозволило спростити розробку дизайну.

Були створені іконки додатку, які містять емблему факультету ЕлІТ (рис. 4.2). Іконки були створені з використанням CorelDRAW Graphics Suite X8.



Рисунок 4.2 – Іконка додатку MyELIT на стартовому екрані

Залишився останній момент – вибір шрифту. Враховуючи різні можливості людей бачити текст, треба підібрати правильний та доступний для всіх умов комфорту читання шрифт. Перша версія додатку мала не комфортний для читання шрифт – це була “Helvetica-Neue-Thin” (рис. 4.3), шрифт був тонким тому оку було дуже складно з орієнтуватись на словах.

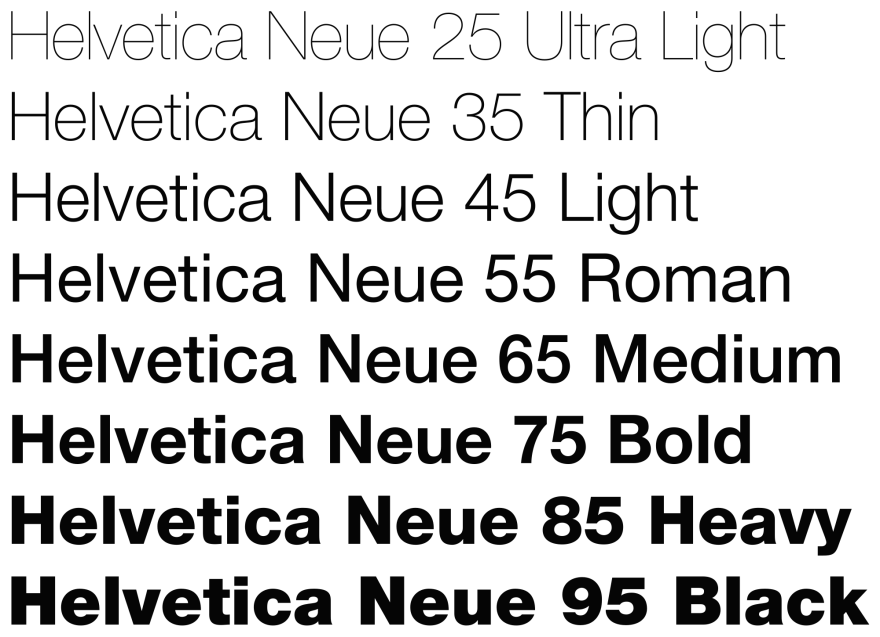


Рисунок 4.3 – Helvetica-Neue-Thin

Windows Phone версія додатку виправила цю проблему, за основу був взятий новий читабельний шрифт “Roboto” (рис. 3.7)



Рисунок 4.4 – Шрифт “Roboto”

Цей шрифт використовуються у різних варіаціях, наприклад перша варіація шрифту – “Roboto-Medium” використовується у заголовках кожного екрану і не більше (рис. 3.8).

Друга варіація - “Roboto-Regular”, використовується у додатковому тексті, наприклад: текст опису розділу у головному меню (рис. 3.8) або опис завантажування файлів, який знаходиться при кожній можливості завантаження PDF документу. Отже практично на всіх цих елементах тримається дизайн мобільного додатку але як завжди є виключення і це – радіо та система ECTS.

## 4.2 UX інтерфейс мобільного додатку “MyELIT”

Досвід користувача (user experience, скорочено UX) – це відчуття людини при взаємодії з системою. Системою може виступати мобільний додаток, веб-сайт, програмне забезпечення, тощо.

Ті, хто працює над створенням UX (так звані UX дизайнери) проводять дослідження і оцінюють взаємодію користувачів з системою, беручи до уваги такі речі, як простота використання, сприйняття вартості системи, корисності, ефективності тощо.

Живі плитки – фірмова особливість інтерфейсу Windows Phone і Windows, один з ключових елементів дизайну Microsoft. Живі плитки – це красиві ярлики програм, на які додаток може виводити якийсь контент і повідомлення і деяким чином взаємодіяти з користувачем. Така взаємодія є гарним прикладом реалізації UX. Якщо додаток не має живі плитки, він зазвичай залишається в загальному списку додатків і буде запускатися значно рідше. Додаток MyELIT має живу плитку (рис. 4.1), на якій може відображати актуальний розклад СумДУ. Це дозволяє додатку залишатися на стартовому екрані та спонукає користувача частіше запускати MyELIT.

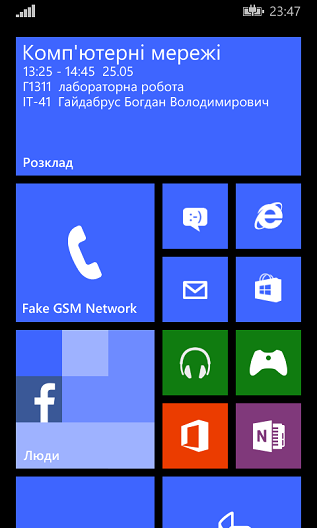


Рисунок 4.1 – Жива плитка MyELIT на стартовому екрані

Таким чином за весь період створення мобільного додатку було використано більше двох варіацій створення мобільного дизайну. Була проведена велика робота по визначенню шрифтів та кольорів, які б не навантажували очі та користувачу було б комфортніше читати або переглядати текстову інформацію. UX може існувати і працювати дуже ефективно з досить простим для користувача інтерфейсом. Наприклад, ви можете зустріти додаток з приголомшливим дизайном, яке складно використовувати (хороший UI, поганий UX). Також бувають додатки, які виглядають погано, але виявляються дуже зручними у використанні (поганий UI, хороший UX). Якість UX визначають спеціальні заходи тестування мобільного додатку, іншими словами це людина – тестер, саме вона перевіряє на зрозумілість для різних користувачів інтерфейс та перевіряє функціонал додатку, а після перевірки віддає звітність що до нюансів, які потрібно виправити для того, щоб пустити продукт на ринок для загального користування.

# ВИСНОВОК

Дипломна робота орієнтована не тільки на абітурієнтів, які бажають навчатись на факультеті ЕлІТ, а і для студентів, які вже навчаються або закінчили університет але мріють вчитись на аспірантурі або докторантурі. У рамках дипломної роботи було розглянуто другу оновлену версію мобільного додатку “MyELIT”, його функції та можливості.

Було проаналізовано існуючі додатки MyELIT для платформ IOS та Android. Додатку для популярної платформи Windows Phone не існувало. Тому було прийнято рішення про заповнення цієї «прогалини». Було вивчено потребу користувачів у мобільному додатку Window Phone 8.1, спроектовано унікальну архітектуру додатку, яка відповідає концепціям Windows Phone.

Проект реалізовано з використанням мови програмування С#, XAML, що забезпечує достатню швидкодію додатку. Були використані методологія ООП, патерни проектування та особливості мов для запезпечення якості коду і легкої подальшої підтримки проекту.

Додаток наслідував увесь розроблений функціонал попередників: інформаційну складову, радіо, зв’язок з деканатом. MyELIT мав інформаційно-розважальний характер.

Версія для Windows Phone, у порівнянні з існуючими додатками, отримала новий унікальний додатковий функціонал, а саме: відображення новин факультету та університету, реалізовано розклад СумДУ, його відслідковування та відображення на живій плитці на стартовому екрані. MyELIT набув сервісно-інформаційного характеру.

Було ретельно описано основні функції додатку, архітектуру головного меню, UI та UX дизайн. У ході роботи можна було побачити, як змінювався дизайн проекту за часи його розробки починаючі від шрифтів та закінчуючи графікою. Також не забули про детальний аналіз ринку мобільних технологій, оцінили всі його переваги та недоліки, ознайомились з системою аналітики, розглянули типи систем аналізу.

Також оновлена версія “MyELIT” має три мові локалі (Українська, Російська, та Англійська), доречі остання була створена спеціально для іноземних студентів.

# СПИСОК ЛІТЕРАТУРИ

1. Microsoft Developer Network [Електронний ресурс] – Режим доступу до ресурсу: https://msdn.microsoft.com.
2. Stack Overflow [Електронний ресурс] – Режим доступу до ресурсу: http://stackoverflow.com/.
3. Habrahabr [Електронний ресурс] – Режим доступу до ресурсу: https://habrahabr.ru/.
4. Github [Електронний ресурс] – Режим доступу до ресурсу: https://github.com/.
5. Нейгел К. C# 5.0 и платформа .NET 4.5 для профессионалов / К. Нейгел, Б. Ивьен. – Москва: Вильямс, 2014. – 1440 с.
6. Шевчук А., Касьянов А. Patterns via C#. Приемы объектно-ориентированного проектирования: ITVDN / А. Шевчук, Д. Охрименко, 2015 – 288с.
7. Иванов Д., Ф. Новиков Моделирование на UML / Д. Иванов, Ф. Новиков – Санкт-Петербург: СПбГУ ИТМО, 2010. - 200 с. - экз.
8. Proft [Електронний ресурс] – Режим доступу до ресурсу: 8. http://proft.me/2013/05/26/uml-2-tipy-diagramm.
9. Mymanager [Електронний ресурс] – Режим доступу до ресурсу: http://mymanager.com.ua/bp/bs/overview/notation\_idef0.php.
10. Батенко Л. П., Загородніх О. А., Ліщинська В. В. Б 28 Управління проектами: Навч. посібник. — К.: КНЕУ, 2003. — 231 с.
11. Ubooks [Електронний ресурс] – Режим доступу до ресурсу: http://ubooks.com.ua/books/000245/inx18.php.
12. Ubooks [Електронний ресурс] – Режим доступу до ресурсу: http://ubooks.com.ua/books/000245/inx14.php.
13. Wikipedia [Електронний ресурс] – Режим доступу до ресурсу: 13. https://uk.wikipedia.org/wiki/Діаграма\_Ганта.
14. Studfiles [Електронний ресурс] – Режим доступу до ресурсу: http://www.studfiles.ru/preview/5534709/page:9/.
15. Джозеф А. C# 5.0. Карманный справочник / А. Джозеф, А. Бен., 2013. – 288 с.
16. Симан М. Dependency Injection in .NET / М. Симан, А. Барышнев, Е. Зазноба., 2014. – 364 с.
17. Millett S. Patterns, Principles, and Practices of Domain-Driven Design / S. Millett, N. Tune., 2015. – 795 с.
18. Натан А. WPF 4: Unleashed / Адам Натан., 2011. – 880 с.
19. Троелсен Э. Pro Expression Blend 4 / Эндрю Троелсен., 2012. – 368 с. – (Вильямс). – (Expert's Voice).
20. Garofalo R. pic Building Enterprise Applications with Windows® Presentation Foundation and the Model View ViewModel Pattern / Raffaele Garofalo., 2011. – 225 с.

# ДОДАТОК А. Технічне завдання

Міністерство освіти і науки України

Сумський державний університет

Кафедра комп’ютерних наук

Секція інформаційних технологій проектування

Технічне завдання

на розроблення програмного продукту

**Модуль перевірки знань студентів**

**Шифр: ПП-МД-МЕ**

**Аркушів 6**

2016

Додаток А.Технічне завдання

ЗМІСТ

[1 ВСТУП 58](#_Toc452633807)

[1.2 Область застосування 59](#_Toc452633808)

[1.3 Об’єкт, в якому використовують програму 59](#_Toc452633809)

[2 ПІДСТАВА ДЛЯ РОЗРОБКИ 59](#_Toc452633810)

[2.1 Документи на основі яких ведеться проектування 59](#_Toc452633811)

[2.2 Шифр теми 60](#_Toc452633812)

[3 ВИМОГИ ДО ПРОГРАМНОГО ВИРОБУ 60](#_Toc452633813)

[3.1 Вимоги до функціональних характеристик 60](#_Toc452633814)

[3.2 Вимоги до надійності 61](#_Toc452633815)

[3.3 Умови експлуатації 61](#_Toc452633816)

[3.4 Вимоги до складу і параметрами технічних засобів 61](#_Toc452633817)

[3.5 Вимоги до інформаційної та програмної сумісності 62](#_Toc452633818)

[4 ВИМОГИ ДО ПРОГРАМНОЇ ДОКУМЕНТАЦІЇ 62](#_Toc452633819)

[5 СТАДІЇ ТА ЕТАПИ РОЗРОБКИ 62](#_Toc452633820)

[6 ПОРЯДОК КОНТРОЛЮ ТА ПРИЙМАННЯ 63](#_Toc452633821)

# 1 ВСТУП

Дана програма використовується викладачами для перевірки набутих знань студентів, шляхом проходження тесту по вибраному предмету.

**1.1** Найменування програми

Додаток MyELIT для Windows Phone. повинен відповідати поставленим критеріям:

* мобільний додаток повинен бути безкоштовним;
* мобільний додаток повинен бути без реклами, яка б псувала або заважали комфортній роботі;
* інтерфейс повинен бути інтуїтивно-зрозумілим, повинен підходити для кожного користувача не залежно від полу, року та здібностей;
* усі тексти повинні бути читабельними, кольори - не навантажувати очі як при денній так і вечірній годині;
* надати можливість перегляду розкладу СумДУ.
* наявність автоматичного відслідковування розкладу;
* відображення новин університету та факультету;
* надання доступу до корисної інформації;
* надати користувачу детальний опис дистанційного, заочного та денного форм навчання в СумДУ, ознайомити користувачів з методом навчання на дистанційній формі;

## 1.2 Область застосування

Полегшення доступу до корисної інформації, максимальне ознайомлення абітурієнта з важливою інформацією, що стосується вступу до факультету ЕлІТ, також з навчальним закладом СумДУ, з персоналом кафедри.

## 1.3 Об’єкт, в якому використовують програму

Додаток «MyELIT» можуть використовувати в студенти, абітурієнти, викладачі.

# 2 ПІДСТАВА ДЛЯ РОЗРОБКИ

Розробка ведеться на основі дипломної роботи.

Найменування організації: група ІТ-21, кафедра Комп’ютерних наук, секція ІТП. Тема проекту: «Мобільний додаток «MyELIT»»

## 2.1 Документи на основі яких ведеться проектування

1. Техническоезадание. Требования к содержаниюи оформлению. ГОСТ19.201-78.   
2. Руководство системного программиста. Требования к содержанию и оформлению. ГОСТ19.503-79

 3. Руководство оператора. Требования к содержаниюи оформлению. ГОСТ19.505-79

4. C# 2005 и платформа .Net 3.0 для профессионалов. Нейгел К., Ивьен Б. и др. –М.: ООО “И.Д. Вильямс", 2008. –1376 с.

5. ЭнрюТроелсен. Язык программирования C# 5.0 и платформа .Net 4.5, 6-е изд: Пер. с англ. –М.: ООО “И.Д. Вильямс", 2013. –1312 с.

## 2.2 Шифр теми

Шифр теми: ПП-МД-МЕ.

# 3 ВИМОГИ ДО ПРОГРАМНОГО ВИРОБУ

При реалізації та використанні додатку повинні бути враховані вимоги до функціональних характеристик, надійності проекту, параметрами технічних засобів, інформаційної і програмної сумісності.

## 3.1 Вимоги до функціональних характеристик

ПП повинен виконувати наступні функції:

* *читання даних з БД;*
* *реєстрація студента в системі, шляхом вибору групи та прізвища студента;*
* *надання доступу до вибору тесту по конкретному предмету;*
* *автоматично генерувати питання до тесту з БД;*
* *виводити результати тесту в окремому вікні після тестування;*
* *заносити результати до БД;*
* *записувати результати тесту в окремий файл.*

## 3.2 Вимоги до надійності

Розроблюваний програмний продукт для тестування студентів не повинен вимагати від користувача спеціальної підготовки і так само має мати захист від помилкових дій користувача, а саме:

* повинна бути розроблена інструкція користувача;
* контроль вибору даних при реєстрації в системі;
* контроль вибору тестів з обраної дисципліни;
* контроль доступу до питань та відповідей з тесту;
* контроль проходження тесту.

## 3.3 Умови експлуатації

Програма може бути перенесена на такі види носіїв даних як: лазерний диск, флеш-накопичувач, жорсткий диск. Не повинно бути впливів води і механічних ударів на носій. Умови експлуатації повинні відповідати санітарним та технічним нормам експлуатації ЕОМ і визначаються відповідно до СанПіН2.2.2545-96 «Гігієнічні вимоги до відео-дисплейних терміналів, персональних обчислювальних машин і організації роботи». Для роботи з ПК допускаються працівники, які мають достатній рівень знань в предметній області.

## 3.4 Вимоги до складу і параметрами технічних засобів

* Мобільний телефон з операційною системою Windows Phone 8.1;

## 3.5 Вимоги до інформаційної та програмної сумісності

Для роботи програмного продукту автоматичного формування специфікацій на машині повинні бути встановлені:

* операційна система Windows Phone 8.1;

# 4 ВИМОГИ ДО ПРОГРАМНОЇ ДОКУМЕНТАЦІЇ

Попередній склад програмної документації встановлено у відповідності з ГОСТ19.101-77. Нижче наведений список програмних документів та їх зміст.  
Технічне завдання - даний документ.

Експлуатаційні документи - керівництво користувача до програмного продукту та керівництво системного адміністратора.

# 5 СТАДІЇ ТА ЕТАПИ РОЗРОБКИ

1. до 15.03.2016 р. Аналіз існуючих аналогів.
2. до 20.03.2016 р. Розробка ТЗ.
3. до 29.03.2016 р. Розробка моделі ПП.
4. до 10.05.2016 р. Розробка програмного продукту в Visual Studio.
5. до 28.05.2016 р. Відладка та тестування.
6. до 02.06.2016 р. Розробка супроводжуючої документації
7. до 04.06.2016 р. Здача в експлуатацію

# 6 ПОРЯДОК КОНТРОЛЮ ТА ПРИЙМАННЯ

Контроль коректності функціонування і придатність мобільного додатку MyELIT для Windows Phone до використання в робочих умовах виконується спільно розробниками і споживачами даного програмного комплексу на підставі його тестування. Рішення про приймання в експлуатацію приймається на підставі результатів тестування .

Порядок контролю:

1.Контроль плану роботи (до 15.03.2016 р.)

2.Контроль проекту (до 10.05.2016 р.)

3.Тестування додатку (до 28.05.2016 р.)

4. Здача супроводжуючої документації (до 02.06.2016 р.)

6. Презентація роботи (до 10.06.2016 р.)

# ДОДАТОК Б

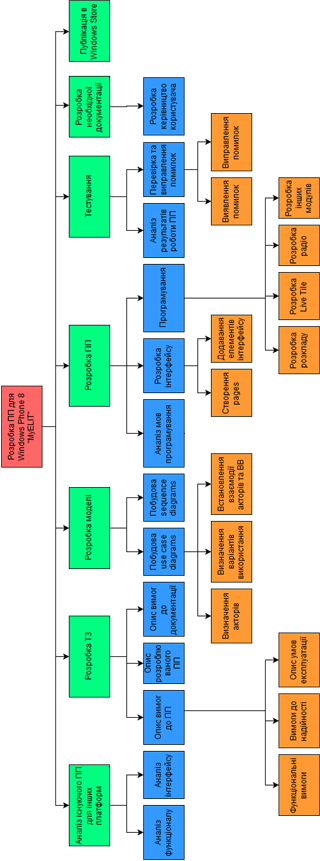


Рисунок Б.1 – Ієрархічна структура робіт (WBS)

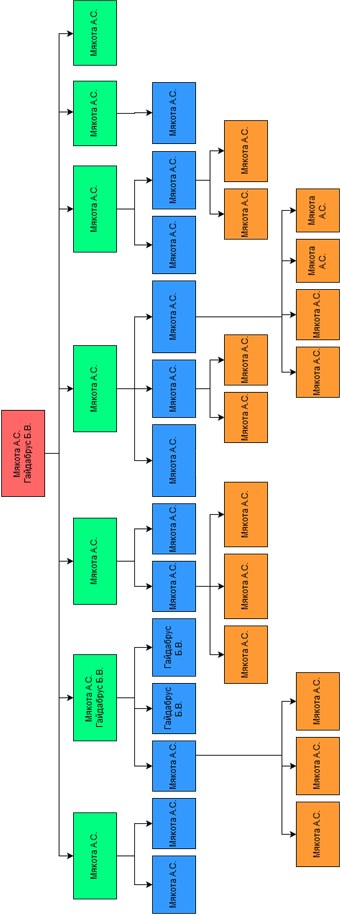


Рисунок Б.2 – Організаційна структура виконавців проекту (OBS)

# ДОДАТОК В

Таблиця В.1 – Матриця відповідальності

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Розробка ПП** | **Гайдабрус Б. В.** | **Мякота А. С.** |
| 1 | Аналіз існуючого ПП для інших платформ |  |  |
| 1.1 | Аналіз функціоналу |  |  |
| 1.2 | Аналіз інтерфейсу |  |  |
| 2 | Розробка ТЗ |  |  |
| 2.1 | Опис вимог до ПП |  |  |
| 2.1.1 | Функціональні вимоги |  |  |
| 2.1.2 | Вимоги до надійності |  |  |
| 2.1.3 | Опис умов експлуатації |  |  |
| 2.2 | Опис розроблюваного ПП |  |  |
| 2.3 | Опис вимог до документації |  |  |
| 3 | Розробка моделі |  |  |
| 3.1 | Побудова use case diagrams |  |  |
| 3.1.1 | Визначення акторів |  |  |
| 3.1.2 | Визначення варіантів використання |  |  |
| 3.1.3 | Встановлення взаємодії акторів та ВВ |  |  |
| 3.2 | Побудова sequence diagrams |  |  |
| 4 | Розробка ПП |  |  |
| 4.1 | Аналіз мов програмування |  |  |
| 4.2 | Розробка інтерфейсу |  |  |
| 4.2.1 | Створення pages |  |  |
| 4.2.2 | Додавання елементів інтерфейсу |  |  |
| 4.3 | Програмування |  |  |
| 4.3.1 | Розробка розкладу |  |  |
| 4.3.2 | Розробка Live Tile |  |  |
| 4.3.3 | Розробка радіо |  |  |
| 4.3.4 | Розробка інших модулів |  |  |
| 5 | Тестування |  |  |
| 5.1 | Аналіз результатів роботи ПП |  |  |
| 5.2 | Перевірка та виправлення помилок |  |  |
| 5.2.1 | Виявлення помилок |  |  |
| 5.2.2 | Виправлення помилок |  |  |
| 6 | Розробка необхідної документації |  |  |
| 6.1 | Розробка керівництво користувача |  |  |
| 7 | Публікація в Windows Store |  |  |

# ДОДАТОК Г

Код сторінки новин:

using System;

using System.Collections.Generic;

using System.Collections.ObjectModel;

using System.IO;

using System.Linq;

using System.Net.Http;

using System.Runtime.InteropServices.WindowsRuntime;

using System.Text.RegularExpressions;

using System.Threading.Tasks;

using Windows.Foundation;

using Windows.Foundation.Collections;

using Windows.UI.Xaml;

using Windows.UI.Xaml.Controls;

using Windows.UI.Xaml.Controls.Primitives;

using Windows.UI.Xaml.Data;

using Windows.UI.Xaml.Input;

using Windows.UI.Xaml.Media;

using Windows.UI.Xaml.Navigation;

using MyELIT;

using Windows.UI.Xaml.Media.Imaging;

using Windows.System;

namespace MyELIT

{

public sealed partial class NewsSumduPage : Page

{

private ObservableCollection<Entry> NewsSSUList { get; set; }

private ObservableCollection<Entry> NewsElitList { get; set; }

public NewsSumduPage()

{

this.InitializeComponent();

this.NavigationCacheMode = NavigationCacheMode.Disabled;

NewsSSUList = new ObservableCollection<Entry>();

NewsElitList = new ObservableCollection<Entry>();

}

protected async override void OnNavigatedTo(NavigationEventArgs e)

{

if ((e.Parameter as String) == "SSU")

{

List.ItemsSource = NewsSSUList;

if (NewsSSUList.Count == 0) await GetSSUNews();

}

if ((e.Parameter as String) == "Elit")

{

List.ItemsSource = NewsElitList;

if (NewsSSUList.Count == 0) await GetElitNews();

}

}

private async Task GetSSUNews()

{

try

{

var rl = new Windows.ApplicationModel.Resources.ResourceLoader();

HttpClient httpClient = new HttpClient();

String ResponseString = await httpClient.GetStringAsync(new Uri(rl.GetString("news\_sumdu\_page\_ssuaddr")));

String p\_title = @"<title>(?<title>[\d\D]\*?)</title>";

String p\_link = @"<link>(?<link>[\d\D]\*?)</link>";

String p\_desc = @"<description>(?<description>[\d\D]\*?)</description>";

var titles = Regex.Matches(ResponseString, p\_title).OfType<Match>().Select(m => m.Groups["title"].Value).ToArray();

var links = Regex.Matches(ResponseString, p\_link).OfType<Match>().Select(m => m.Groups["link"].Value).ToArray();

var descriptions = Regex.Matches(ResponseString, p\_desc).OfType<Match>().Select(m => m.Groups["description"].Value).ToArray();

httpClient = null;

ResponseString = null;

p\_title = null;

p\_link = null;

p\_desc = null;

for (int i = 1; i <= 25; i++)

{

descriptions[i] = Regex.Match(descriptions[i], @"<p[\d\D]\*?>(?<desc>[\d\D]\*?)<\/p>").Groups["desc"].Value;

var uri = Regex.Match(descriptions[i], @"<img.\*?src=""(?<image>[\d\D]\*?)""").Groups["image"].Value;

descriptions[i] = Regex.Replace(descriptions[i], @"\<[\d\D]\*?\>", "", RegexOptions.Multiline);

NewsSSUList.Add(new Entry()

{

Title = titles[i],

Link = links[i],

Description = descriptions[i],

});

try

{

NewsSSUList.Last().ImageUrl = new BitmapImage(new Uri(uri, UriKind.Absolute));

}

catch { }

}

}

catch { }

}

private async Task GetElitNews()

{

try

{

var rl = new Windows.ApplicationModel.Resources.ResourceLoader();

HttpClient httpClient = new HttpClient();

String ResponseString = await httpClient.GetStringAsync(new Uri(rl.GetString("news\_sumdu\_page\_elitaddr")));

var news = Regex.Matches(ResponseString, @"<h2 [\d\D]\*?<\/p>");

httpClient = null;

ResponseString = null;

String uri;

String header;

String link;

String desc;

foreach (Match i in news)

{

uri = "http://elit.sumdu.edu.ua" + Regex.Match(i.Value, @"<img.\*?src=""(?<image>[\d\D]\*?)""").Groups["image"].Value;

link = "http://elit.sumdu.edu.ua" + Regex.Match(i.Value, @"<a.\*?href=""(?<link>[\d\D]\*?)""").Groups["link"].Value;

header = Regex.Match(i.Value, @"<h2 [\d\D]\*?\>(?<header>[\d\D]\*?)\<\/h2\>").Groups["header"].Value;

desc = Regex.Match(i.Value, @"<p[\d\D]\*?\>(?<desc>[\d\D]\*?)\<\/p\>").Groups["desc"].Value;

header = Regex.Replace(header, @"\<[\d\D]\*?\>", "", RegexOptions.Multiline);

header = Regex.Replace(header, @"\t|\n", "", RegexOptions.Multiline);

desc = Regex.Replace(desc, @"\<[\d\D]\*?\>", "", RegexOptions.Multiline);

NewsElitList.Add(new Entry()

{

Title = header,

Link = link,

Description = desc,

});

try

{

NewsElitList.Last().ImageUrl = new BitmapImage(new Uri(uri, UriKind.Absolute));

}

catch { }

}

}

catch { }

}

private class Entry

{

public String Title { get; set; }

public BitmapImage ImageUrl { get; set; }

public String Description { get; set; }

public String Link { get; set; }

}

private async void List\_Tapped(object sender, TappedRoutedEventArgs e)

{

var link = ((sender as ListView).SelectedItem as Entry).Link;

await Launcher.LaunchUriAsync(new Uri(link));

}

}

}

Код головної сторінки:

using Library;

using System;

using System.Collections.Generic;

using System.Collections.ObjectModel;

using System.Globalization;

using System.IO;

using System.Linq;

using System.Net.Http;

using System.Runtime.InteropServices.WindowsRuntime;

using System.Text.RegularExpressions;

using Windows.Foundation;

using Windows.Foundation.Collections;

using Windows.UI.Popups;

using Windows.UI.Xaml;

using Windows.UI.Xaml.Controls;

using Windows.UI.Xaml.Controls.Primitives;

using Windows.UI.Xaml.Data;

using Windows.UI.Xaml.Input;

using Windows.UI.Xaml.Media;

using Windows.UI.Xaml.Media.Imaging;

using Windows.UI.Xaml.Navigation;

namespace MyELIT

{

public sealed partial class MainPage : Page

{

private ObservableCollection<MenuItem> menulist { get; set; }

public MainPage()

{

this.InitializeComponent();

this.NavigationCacheMode = NavigationCacheMode.Required;

menulist = new ObservableCollection<MenuItem>();

var rl = new Windows.ApplicationModel.Resources.ResourceLoader();

menulist.Add(new MenuItem()

{

Image = new BitmapImage(new Uri("ms-appx:///Assets/MainPage/1463919136\_calendar.png")),

Header = rl.GetString("main\_page\_schedule/Text"),

Type = typeof(ScheduleQueryPage)

});

menulist.Add(new MenuItem()

{

Image = new BitmapImage(new Uri("ms-appx:///Assets/MainPage/1463919136\_calendar.png")),

Header = rl.GetString("main\_page\_myschedule/Text"),

Type = typeof(ScheduleViewPage)

});

menulist.Add(new MenuItem()

{

Image = new BitmapImage(new Uri("ms-appx:///Assets/MainPage/1463922079\_news.png")),

Header = rl.GetString("main\_page\_history/Text"),

Type = typeof(ScheduleHistoryPage)

});

menulist.Add(new MenuItem()

{

Image = new BitmapImage(new Uri("ms-appx:///Assets/MainPage/1463921790\_settings.png")),

Header = rl.GetString("main\_page\_conf/Text"),

Type = typeof(ScheduleConfigurationPage),

});

menulist.Add(new MenuItem()

{

Image = new BitmapImage(new Uri("ms-appx:///Assets/MainPage/1463987317\_note.png")),

Header = rl.GetString("main\_page\_elitnews/Text"),

Type = typeof(NewsSumduPage),

Param = "Elit"

});

menulist.Add(new MenuItem()

{

Image = new BitmapImage(new Uri("ms-appx:///Assets/MainPage/1463987317\_note.png")),

Header = rl.GetString("main\_page\_ssunews/Text"),

Type = typeof(NewsSumduPage),

Param = "SSU"

});

menulist.Add(new MenuItem()

{

Image = new BitmapImage(new Uri("ms-appx:///Assets/MainPage/1463987171\_study.png")),

Header = rl.GetString("main\_page\_ocw/Text"),

Type = typeof(OCWPage)

});

menulist.Add(new MenuItem()

{

Image = new BitmapImage(new Uri("ms-appx:///Assets/MainPage/1463987210\_music.png")),

Header = rl.GetString("main\_page\_radio/Text"),

Type = typeof(RadioPage)

});

menulist.Add(new MenuItem()

{

Image = new BitmapImage(new Uri("ms-appx:///Assets/MainPage/1463987210\_music.png")),

Header = "11111111111",

Type = typeof(SiteProFacZagInfPage)

});

listView.ItemsSource = menulist;

}

protected async override void OnNavigatedTo(NavigationEventArgs e)

{

GetList("Lists\\Teachers", "http://schedule.sumdu.edu.ua/index/json?method=getTeachers");

GetList("Lists\\Auditoriums", "http://schedule.sumdu.edu.ua/index/json?method=getAuditoriums");

GetList("Lists\\Groups", "http://schedule.sumdu.edu.ua/index/json?method=getGroups");

try

{

var a = await Windows.Storage.ApplicationData.Current.LocalFolder.GetFileAsync("Config");

}

catch

{

var \_ = Config.Instance.SaveConfig();

}

var b = Config.Instance.ReadConfig();

}

public static async void GetList(String path, String url)

{

try

{

HttpClient httpClient = new HttpClient();

String ResponseString = await httpClient.GetStringAsync(new Uri(url));

ResponseString = DecodeEncodedNonAsciiCharacters(ResponseString);

ResponseString = RemoveSlashes(ResponseString);

var localAppFolder = Windows.Storage.ApplicationData.Current.LocalFolder;

await localAppFolder.CreateFolderAsync("Lists", Windows.Storage.CreationCollisionOption.OpenIfExists);

var fileHandle = await localAppFolder.CreateFileAsync(path, Windows.Storage.CreationCollisionOption.ReplaceExisting);

String results = "";

String pattern\_pair = @"""(?<id>[\d]+?)""\:""(?<name>[\d\D]\*?)""";

MatchCollection events = Regex.Matches(ResponseString, pattern\_pair, RegexOptions.Multiline);

foreach (Match m in events)

{

results += Int32.Parse(m.Groups["id"].Value) + "@" + m.Groups["name"].Value + Environment.NewLine;

}

await Windows.Storage.FileIO.WriteTextAsync(fileHandle, results);

}

catch { }

}

static string DecodeEncodedNonAsciiCharacters(string value)

{

return Regex.Replace(

value,

@"\\u(?<Value>[a-zA-Z0-9]{4})",

m =>

{

return ((char)int.Parse(m.Groups["Value"].Value, NumberStyles.HexNumber)).ToString();

});

}

static string RemoveSlashes(string value)

{

return Regex.Replace(

value,

@"\\",

m =>

{

return "";

});

}

private class MenuItem

{

public BitmapImage Image { get; set; }

public String Header { get; set; }

public Type Type { get; set; }

public System.Object Param { get; set; }

}

private async void listView\_Tapped(object sender, TappedRoutedEventArgs e)

{

var ind = listView.SelectedIndex;

if (!(ind == 1 || ind == 2 || ind == 3))

{

if (!NIC.CheckNIC())

{

var rl = new Windows.ApplicationModel.Resources.ResourceLoader();

MessageDialog messageDialog = new MessageDialog(rl.GetString("main\_page\_errorNIC"));

await messageDialog.ShowAsync();

return;

}

}

var item = listView.SelectedItem as MenuItem;

if (item.Param == null)

{

Frame.Navigate(item.Type);

}

else

{

Frame.Navigate(item.Type, item.Param);

}

}

}

}

Код сторінки розкладу:

using Library;

using System;

using System.Collections.Generic;

using System.Collections.ObjectModel;

using System.Globalization;

using System.IO;

using System.Linq;

using System.Net.Http;

using System.Runtime.InteropServices.WindowsRuntime;

using System.Text.RegularExpressions;

using System.Threading.Tasks;

using Windows.Foundation;

using Windows.Foundation.Collections;

using Windows.Storage;

using Windows.UI.ViewManagement;

using Windows.UI.Xaml;

using Windows.UI.Xaml.Controls;

using Windows.UI.Xaml.Controls.Primitives;

using Windows.UI.Xaml.Data;

using Windows.UI.Xaml.Input;

using Windows.UI.Xaml.Media;

using Windows.UI.Xaml.Navigation;

namespace MyELIT

{

public sealed partial class ScheduleQueryPage : Page

{

private int TeacherId = 0, AuditoriumId = 0, GroupId = 0;

private ObservableCollection<Library.Object> grList { get; set; }

private ObservableCollection<Library.Object> tcList { get; set; }

private ObservableCollection<Library.Object> adList { get; set; }

public ObservableCollection<Library.Day> DayList { get; private set; }

public ScheduleQueryPage()

{

this.InitializeComponent();

this.NavigationCacheMode = NavigationCacheMode.Required;

date2.Date = DateTime.Now.AddDays(7);

grList = new ObservableCollection<Library.Object>();

tcList = new ObservableCollection<Library.Object>();

adList = new ObservableCollection<Library.Object>();

DayList = new ObservableCollection<Library.Day>();

}

protected override void OnNavigatedTo(NavigationEventArgs e)

{

c1.ItemsSource = grList;

c2.ItemsSource = tcList;

c3.ItemsSource = adList;

Read("Lists\\Groups", 1);

Read("Lists\\Teachers", 2);

Read("Lists\\Auditoriums", 3);

}

public async void Read(String path, int var)

{

StorageFolder local = Windows.Storage.ApplicationData.Current.LocalFolder;

Stream stream = await local.OpenStreamForReadAsync(path);

String line;

String[] pair = new String[2];

using (StreamReader reader = new StreamReader(stream))

{

while ((line = reader.ReadLine()) != null)

{

pair = line.Split('@');

if (var == 1) grList.Add(new Library.Object(Int32.Parse(pair[0]), pair[1]));

if (var == 2) tcList.Add(new Library.Object(Int32.Parse(pair[0]), pair[1]));

if (var == 3) adList.Add(new Library.Object(Int32.Parse(pair[0]), pair[1]));

}

}

}

private async void Button\_Click\_3(object sender, RoutedEventArgs e)

{

if (TeacherId != 0 || AuditoriumId !=0 || GroupId != 0)

{

StatusBarProgressIndicator progressbar = StatusBar.GetForCurrentView().ProgressIndicator;

var rl = new Windows.ApplicationModel.Resources.ResourceLoader();

progressbar.Text = rl.GetString("schedule\_query\_page\_loading");

progressbar.ShowAsync();

//&id\_grp=0&id\_fio=0&id\_aud=0&date\_beg=00.00.0000&date\_end=00.00.0000

String id\_grp = GroupId != 0 ? "&id\_grp=" + GroupId : "";

String id\_fio = TeacherId != 0 ? "&id\_fio=" + TeacherId : "";

String id\_aud = AuditoriumId != 0 ? "&id\_aud=" + AuditoriumId : "";

String beg = "" + (date1.Date.Day < 10 ? "0" : "") + date1.Date.Day + "." + (date1.Date.Month < 10 ? "0" : "") + date1.Date.Month + "." + date1.Date.Year;

String end = "" + (date2.Date.Day < 10 ? "0" : "") + date2.Date.Day + "." + (date2.Date.Month < 10 ? "0" : "") + date2.Date.Month + "." + date2.Date.Year;

String date\_beg = "&date\_beg=" + beg;

String date\_end = "&date\_end=" + end;

String postfix = id\_grp + id\_fio + id\_aud + date\_beg + date\_end;

DateTime dt = DateTime.Now;

String folder = "History\\" + GroupId + "@" + TeacherId + "@" + AuditoriumId + "@" + beg + "@" + end + "@" + dt.Year + "." + dt.Month + "." + dt.Day + "." + dt.Hour + "." + dt.Minute + "." + dt.Second + "\\";

await saveSchedule(postfix, folder);

await readSchedule(folder); // History\\ + postfix

Frame.Navigate(typeof(ScheduleViewPage), DayList);

progressbar.HideAsync();

}

}

private async Task saveSchedule(String postfix, String path)

{

try

{

HttpClient httpClient = new HttpClient();

String ResponseString = await httpClient.GetStringAsync(new Uri("http://schedule.sumdu.edu.ua/index/json?method=getSchedules" + postfix));

ResponseString = DecodeEncodedNonAsciiCharacters(ResponseString);

ResponseString = RemoveSlashes(ResponseString);

string p\_DATE\_REG = @"""DATE\_REG"":""(?<DATE\_REG>[\d\D]\*?)""";

string p\_TIME\_PAIR = @"""TIME\_PAIR"":""(?<TIME\_PAIR>[\d\D]\*?)""";

string p\_NAME\_FIO = @"""NAME\_FIO"":""(?<NAME\_FIO>[\d\D]\*?)""";

string p\_NAME\_AUD = @"""NAME\_AUD"":""(?<NAME\_AUD>[\d\D]\*?)""";

string p\_NAME\_GROUP = @"""NAME\_GROUP"":""(?<NAME\_GROUP>[\d\D]\*?)""";

string p\_ABBR\_DISC = @"""ABBR\_DISC"":""(?<ABBR\_DISC>[\d\D]\*?)""";

string p\_NAME\_STUD = @"""NAME\_STUD"":""(?<NAME\_STUD>[\d\D]\*?)""";

string p\_REASON = @"""REASON"":""(?<REASON>[\d\D]\*?)""";

string p\_INFO = @"""INFO"":""(?<INFO>[\d\D]\*?)""";

var DATE\_REG = Regex.Matches(ResponseString, p\_DATE\_REG).OfType<Match>().Select(m => m.Groups["DATE\_REG"].Value).ToArray();

var TIME\_PAIR = Regex.Matches(ResponseString, p\_TIME\_PAIR).OfType<Match>().Select(m => m.Groups["TIME\_PAIR"].Value).ToArray();

var NAME\_FIO = Regex.Matches(ResponseString, p\_NAME\_FIO).OfType<Match>().Select(m => m.Groups["NAME\_FIO"].Value).ToArray();

var NAME\_AUD = Regex.Matches(ResponseString, p\_NAME\_AUD).OfType<Match>().Select(m => m.Groups["NAME\_AUD"].Value).ToArray();

var NAME\_GROUP = Regex.Matches(ResponseString, p\_NAME\_GROUP).OfType<Match>().Select(m => m.Groups["NAME\_GROUP"].Value).ToArray();

var ABBR\_DISC = Regex.Matches(ResponseString, p\_ABBR\_DISC).OfType<Match>().Select(m => m.Groups["ABBR\_DISC"].Value).ToArray();

var NAME\_STUD = Regex.Matches(ResponseString, p\_NAME\_STUD).OfType<Match>().Select(m => m.Groups["NAME\_STUD"].Value).ToArray();

var REASON = Regex.Matches(ResponseString, p\_REASON).OfType<Match>().Select(m => m.Groups["REASON"].Value).ToArray();

var INFO = Regex.Matches(ResponseString, p\_INFO).OfType<Match>().Select(m => m.Groups["INFO"].Value).ToArray();

httpClient = null;

ResponseString = null;

p\_DATE\_REG = null;

p\_TIME\_PAIR = null;

p\_NAME\_FIO = null;

p\_NAME\_AUD = null;

p\_NAME\_GROUP = null;

p\_ABBR\_DISC = null;

p\_NAME\_STUD = null;

p\_REASON = null;

p\_INFO = null;

int t = DATE\_REG.Length;

for (int i = 0; i < t; i++)

{

DATE\_REG[i] = String.Join("", DATE\_REG[i].Split('.').Reverse());

}

SortedDictionary<String, SortedDictionary<String, Event>> SortedSchedule = new SortedDictionary<String, SortedDictionary<String, Event>>();

for (int i = 0; i < t; i++)

{

if (!SortedSchedule.ContainsKey(DATE\_REG[i])) SortedSchedule.Add(DATE\_REG[i], new SortedDictionary<String, Event>());

SortedSchedule[DATE\_REG[i]].Add(TIME\_PAIR[i], new Event(TIME\_PAIR[i], NAME\_FIO[i], NAME\_AUD[i], NAME\_GROUP[i], ABBR\_DISC[i], NAME\_STUD[i], REASON[i], INFO[i]));

}

DATE\_REG = null;

TIME\_PAIR = null;

NAME\_FIO = null;

NAME\_AUD = null;

NAME\_GROUP = null;

ABBR\_DISC = null;

NAME\_STUD = null;

REASON = null;

INFO = null;

String results;

var dt = DateTime.Now;

var localAppFolder = Windows.Storage.ApplicationData.Current.LocalFolder;

foreach (String i in SortedSchedule.Keys)

{

results = "";

foreach (var j in SortedSchedule[i].Values)

{

results +=

j.START\_TIME + "@" +

j.END\_TIME + "@" +

j.NAME\_FIO + "@" +

j.NAME\_AUD + "@" +

j.NAME\_GROUP + "@" +

j.ABBR\_DISC + "@" +

j.NAME\_STUD + "@" +

j.REASON + "@" +

j.INFO +

Environment.NewLine;

}

var fileHandle = await localAppFolder.CreateFileAsync(path + i, Windows.Storage.CreationCollisionOption.ReplaceExisting);

await Windows.Storage.FileIO.WriteTextAsync(fileHandle, results);

}

SortedSchedule = null;

}

catch { }

}

private async Task readSchedule(String path\_to\_dir)

{

DayList.Clear();

try

{

var folder = await Windows.Storage.ApplicationData.Current.LocalFolder.GetFolderAsync(path\_to\_dir);

var list = await folder.GetFilesAsync();

String line;

Stream stream;

ObservableCollection<Event> buff = new ObservableCollection<Event>();

foreach (var k in list)

{

stream = await k.OpenStreamForReadAsync();

using (StreamReader reader = new StreamReader(stream))

{

while ((line = reader.ReadLine()) != null)

{

var ev = line.Split('@');

buff.Add(new Event(ev));

}

}

DayList.Add(new Day(k.Name, new ObservableCollection<Event>(buff)));

buff.Clear();

}

}

catch { }

}

string DecodeEncodedNonAsciiCharacters(string value)

{

return Regex.Replace(

value,

@"\\u(?<Value>[a-zA-Z0-9]{4})",

m =>

{

return ((char)int.Parse(m.Groups["Value"].Value, NumberStyles.HexNumber)).ToString();

});

}

string RemoveSlashes(string value)

{

return Regex.Replace(

value,

@"\\",

m =>

{

return "";

});

}

private void c1\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

if (null != c1.SelectedItem)

{

var a = (Library.Object) c1.SelectedItem;

GroupId = a.Id;

}

}

private void c2\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

if (null != c2.SelectedItem)

{

var a = (Library.Object)c2.SelectedItem;

TeacherId = a.Id;

}

}

private void c3\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

if (null != c3.SelectedItem)

{

var a = (Library.Object)c3.SelectedItem;

AuditoriumId = a.Id;

}

}

}

}