

Державний університет «Одеська політехніка»

Інститут комп'ютерних систем

Кафедра інформаційних систем

КУРСОВА РОБОТА

з дисципліни «Програмування мобільних пристроїв»

Тема «Мобільний застосунок для перегляду інформації про футбол»

Студента 3 курсу групи AI_183

спеціальності 122 – «Комп'ютерні науки»

Кеосака А.І.

Керівник: Смик С.Ю.

Національна шкала _____

Кількість балів: _____

Оцінка: ECTS _____

Члени комісії

_____	_____
(підпис)	(прізвище та ініціали)
_____	_____
(підпис)	(прізвище та ініціали)
_____	_____
(підпис)	(прізвище та ініціали)

м. Одеса – 2021 рік

Державний університет «Одеська політехніка»

Інститут комп'ютерних систем

Кафедра інформаційних систем

ЗАВДАННЯ
НА КУРСОВУ РОБОТУ

Кеосак Артем Ігорович

AI-183

1. Тема роботи: Мобільний застосунок для перегляду інформації про футбол

2. Термін здачі студентом закінченої роботи

31.05.2021

3. Зміст розрахунково-пояснювальної записки: вступ, основна частина, висновок, перелік посилань, додаток а

Завдання видано:

04.03.2021

Завдання прийнято до виконання:

04.03.2021

АНОТАЦІЯ

В курсовій роботі розглядається процес створення мобільного застосунку для перегляду інформації про футбол. Цей сервіс був створений на мові програмування Java для платформи Android. Основою застосунку виступає API сервіс «football-data». Він надає можливість отримувати усі потрібні данні через інтернет у реальному часі без потреби зберігати їх на самому пристрої. Сайт сервісу розташований за адресою:

<https://www.football-data.org/>.

Результати роботи розміщено на github-репозиторії за адресою:
<https://github.com/tenn5/kurs>.

THE ANNOTATION

In the course robot, you can see the process of opening the mobile screen for viewing information about football. Tsei service buvs on mobile Java programs for Android platforms. The basis of the lock is the API service "football-data". We have the ability to remove all the data required via the Internet from the real hour without the need to save it on the very attachment. Website for the service of retouching for the address: <https://www.football-data.org/>.

The results of the robot are posted on the github repositories at the address:
<https://github.com/tenn5/kurs>.

Зміст	
ВСТУП	5
1.1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ.....	5
1.2 БАЧЕННЯ СИСТЕМИ	14
1.3 ФОРМУВАННЯ КОРИСТУВАЦЬКИХ ІСТОРІЙ.....	14
ОСНОВНА ЧАСТИНА.....	17
2.1 ПРОЕКТУВАННЯ МОБІЛЬНОГО ЗАСТОСУНКУ	17
2.2 ПРОГРАМНА РЕАЛІЗАЦІЯ	21
2.3 ІНСТРУКЦІЯ ДЛЯ КОРИСТУВАЧА	23
ВИСНОВОК	31
ПЕРЕЛІК ПОСИЛАНЬ.....	31
ДОДАТОК А	33

					ІС КР 122 АІ-183 ПЗ			
З м	А р	№.	П і д п	Д а				
							Л і т .	Л и с т і в
							4	43
							ОНПУ, каф. ІС, гр. АІ183	

ВСТУП

1.1 Аналіз предметної області

Мобільний застосунок або додаток — програмне забезпечення, призначене для роботи на смартфонах, планшетах та інших мобільних пристроях. Багато мобільних застосунків встановлені на самому пристрої або можуть бути завантажені на нього з онлайн магазинів мобільних застосунків, таких як App Store, Google Play, Windows Phone Store та інших, безкоштовно або за плату.

Android – операційна система і платформа для мобільних телефонів та планшетних комп'ютерів, створена компанією Google на базі ядра Linux. Підтримується альянсом Open Handset Alliance (ОНА).

Хоча Android базується на ядрі Linux, він стоїть дещо осторонь Linux-спільноти та Linux-інфраструктури. Базовим елементом цієї операційної системи є реалізація Dalvik віртуальної машини Java, і все програмне забезпечення і застосування спираються на цю реалізацію Java.

У 84 % смартфонів, проданих у 3-му кварталі 2014 року, було встановлено операційну систему Android.

У березні 2017 року ОС Android стала найпопулярнішою ОС, з якої виходили в інтернет. Так, 37,93% користувачів заходили в інтернет із Android'a, а з Windows — 37,91% користувачів. В Азії показники ще вищі — 52,2% і 29,2% відповідно.

Переваги:

Деякі користувачі відзначають, що Android проявляє себе краще одного зі своїх конкурентів, Apple iOS, в ряді особливостей, таких як веб-серфінг, інтеграція з сервісами Google і інших. Також Android, на відміну від iOS, є відкритою платформою, що дозволяє реалізувати на ній більше функцій.

					ІС КР 122 АІ-183 ПЗ	Л и
						5
Змін	Л и	№.	П і д п	Д а		

Незважаючи на початкову заборону на установку програм з «неперевіраних джерел» (наприклад, з карти пам'яті), це обмеження відключається штатними засобами в налаштуваннях пристрою, що дозволяє встановлювати програми на телефони та планшети без інтернет-підключення (наприклад, користувачам, які не мають Wi-Fi-точки доступу і не бажають витратити гроші на мобільний інтернет, який зазвичай коштує дорого), а також дозволяє будь-кому безкоштовно писати програми для Android і тестувати на своєму пристрої.

Android доступний для різних апаратних платформ, таких як ARM, MIPS, x86.

Існують альтернативні Google Play магазини додатків: Amazon Appstore, Opera Mobile Store, GetUpps!, F-Droid.

У версії 4.3 з'явилась підтримка багатокористувацького режиму.

Критика:

Платформа базується на Java (спеціальна реалізація Dalvik), тому переваги і можливості операційної системи Linux на цій платформі практично не використовуються. Наприклад, не використовується жоден з популярних графічних тулкітів і бібліотек (наприклад Qt або GTK), що робить малоімовірною появу значної кількості застосунків, портованих з повноцінного десктопного варіанту Linux на цю платформу через відсутність поза вибором X-сервера і поширених графічних бібліотек.

З'явилася інформація про те, що Google на свій розсуд видалятиме застосунки на телефонах користувачів, якщо порушуються умови їх використання.

Конкуренти Android виступили з критикою платформи, звинувачуючи її в надмірній фрагментації, що створює перешкоди розробникам. Google спростувала всі звинувачення, заявивши, що ніяких подібних проблем немає.

					ІС КР 122 АІ-183 ПЗ	Л и
						6
Змін	Л и	№.	П і д п	Д а		

Щоб користувачі телефонів мали доступ до Google Play та інших сервісів від Google, виробники цих телефонів мають укласти контракт з Google на використання відповідного пропрієтарного програмного забезпечення.

Програми для Android є програмами в нестандартному байт-кодi для віртуальної машини Dalvik.

Google пропонує для вільного завантаження інструментарій для розробки (Software Development Kit), який призначений для x86-машин під операційними системами Linux, Mac OS X (10.4.8 або вище), Windows XP, Windows Vista та Windows 7. Для розробки потрібен Java Development Kit 5 або новіший.

Розробку застосунків для Android можна вести мовою Java (не нижче Java 1.5). Офіційним середовищем розробки є Android Studio, представлене компанією Google в 2013. Крім цього існує плагін для Eclipse — «Android Development Tools» (ADT), призначений для Eclipse версій 3.3-3.7. Для IntelliJ IDEA також існує плагін, який полегшує розробку Android-застосунків. Для середовища розробки NetBeans розроблено плагін, який починаючи з версії Netbeans 7.0 перестав бути експериментальним. Крім того, існує Motodev Studio for Android, що являє собою комплексне середовище розробки, засноване на базі Eclipse і дозволяє працювати безпосередньо з Google SDK.

Крім того, у 2009 році на застосунок до ADT був опублікований Android Native Development Kit (NDK), пакет інструментаріїв і бібліотек дозволяє вести розробку застосунків мовою C/C++. NDK рекомендується використовувати для розробки ділянок коду, критичних до швидкості.

Доступні бібліотеки:

Bionic (Бібліотека стандартних функцій, несумісна з libc);

libc (стандартна системна бібліотека мови C);

мультимедійні бібліотеки (на базі PacketVideo OpenCORE; підтримують такі формати, як MPEG-4, H.264, MP3, AAC, AMR, JPEG та

					ІС КР 122 АІ-183 ПЗ	Л и
						7
Змін	Л и	№.	П і д п	Д а		

PNG);

OpenGL ES 1.0 (рушій двовимірної графіки);

OpenGL ES 1.0 (рушій тривимірної графіки);

Surface Manager (забезпечує для застосунків доступ до 2D/3D);

WebKit (готовий рушій для Web-браузера; обробляє HTML, JavaScript);

FreeType (рушій обробки шрифтів);

SQLite (проста система керування базами даних, доступна для всіх застосувань);

SSL (протокол, що забезпечує безпечну передачу даних мережею).

В порівнянні зі звичайними застосунками Linux, застосунки Android підкоряються додатковим правилам:

Content Providers — обмін даними між застосунками;

Resource Manager — доступ до таких ресурсів, як файли XML, PNG, JPEG;

Notification Manager — доступ до рядка стану;

Activity Manager — управління активними застосунками.

Для Android був розроблений формат інсталяційних пакетів .apk.

Прикладний програмний інтерфейс (інтерфейс програмування застосунків, інтерфейс прикладного програмування) (англ. Application Programming Interface, API) — набір визначень підпрограм, протоколів взаємодії та засобів для створення програмного забезпечення. Спрощено — це набір чітко визначених методів для взаємодії різних компонентів. API надає розробнику засоби для швидкої розробки програмного забезпечення. API може бути для веб-базованих систем, операційних систем, баз даних, апаратного забезпечення, програмних бібліотек.

Одним з найпоширеніших призначень API є надання набору широко використовуваних функцій, наприклад для малювання вікна чи іконок на екрані. Програмісти використовують переваги API у функціональності,

					ІС КР 122 АІ-183 ПЗ	Л и
						8
Змін	Л и	№.	П і д п	Д а		

таким чином їм не доводиться розробляти все з нуля. API є абстрактним поняттям — програмне забезпечення, що пропонує деякий API, часто називають реалізацією (англ. implementation) даного API. У багатьох випадках API є частиною набору розробки програмного забезпечення, водночас, набір розробки може включати як API, так і інші інструменти/апаратне забезпечення, отже ці два терміни не є взаємозамінювані.

Високо рівневі API часто програють у гнучкості. Виконання деяких функцій нижчого рівня стає набагато складнішим, або навіть неможливим.

Прикладний програмний інтерфейс в об'єктно орієнтованих мовах

В об'єктно орієнтованих мовах прикладний програмний інтерфейс зазвичай включає в себе опис набору визначень класу, з набором форм поведінки, пов'язаних з цими класами. Це абстрактне поняття пов'язане з реальними функціями, які надані або надаватимуться, класами, які реалізуються в методах класу.

Прикладний програмний інтерфейс в цьому випадку можна розглядати як сукупність всіх методів, які публічно доступні в класах (зазвичай званий інтерфейс класу). Це означає, що прикладний програмний інтерфейс вказує методи, за допомогою яких взаємодіє з об'єктами, отриманими з визначень класів і обробляє їх.

У більш загальному плані можна визначити Прикладний Програмний Інтерфейс як сукупність усіх видів об'єктів, які можна вивести з визначення класу, і пов'язаних з ними можливих варіантів поведінки.

Наприклад: клас, що представляє Stack, може просто виставити публічно два методи Push() (для додавання нового елемента в стек) і Pop() (для вилучення останнього пункту, ідеально розташований на вершині стека).

У цьому випадку Прикладний Програмний Інтерфейс може бути інтерпретованим як два методи pop() і push(), або, більш широко,

					ІС КР 122 АІ-183 ПЗ	Л и
						9
Змін	Л и	№.	П і д п	Д а		

використовується варіант, коли можна використовувати елемент типу Stack, який реалізує поведінку стека, надаючи йому можливість для додавання / видалення елементів з вершини. Друга інтерпретація видається більш доречною в дусі об'єктноорієнтованого підходу.

Якість документації, пов'язаної з Прикладним Програмним Інтерфейсом, є часто ключовим фактором, що визначає його успішність з точки зору простоти використання.

Бібліотеки і платформи прикладних програмних інтерфейсів

ППІ, як правило, пов'язаний із бібліотеками програмного забезпечення: ППІ описує і вказує очікувану поведінку в той час, як бібліотека є фактичною реалізацією даного набору правил. Один ППІ може мати декілька реалізацій (або жодної, будучи абстрактним) у вигляді різних бібліотек, які мають такий же інтерфейс.

Прикладний програмний інтерфейс також може бути пов'язаним з платформами програмування: платформа може бути заснована на кількох бібліотеках реалізує декілька інтерфейсів ППІ, але на відміну від звичайного використання ППІ, доступ до поведінки вбудований в платформу опосередкований шляхом розширення його змісту новими класами і вставлений в саму платформу. Крім того, загальний потік управління програми може бути під контролем абонента.

Прикладний програмний інтерфейс та протоколи

Прикладний програмний інтерфейс може бути також реалізацією протоколу.

Коли ППІ реалізує протокол, він може бути заснованим на проксі-методах віддалених викликів, що засновані на протоколі зв'язку. Роль ППІ може полягати саме в тому, щоб приховати деталі транспортного протоколу. Наприклад: RMI є ППІ, який реалізує протокол або JRMP ІОР як RMI-ІОР.

Протоколи, як правило, розподіляються між різними технологіями і зазвичай дозволяють різним технологіям обмінюватися інформацією, діючи

					ІС КР 122 АІ-183 ПЗ	Л и
						10
Змін	Л и	№.	П і д п	Д а		

як абстракція між двома світами. ППІ, як правило, є специфічним для конкретної технології: звідси, інтерфейси даної мови не можуть бути використані на інших мовах, якщо виклики функції не будуть перетворені з конкретної адаптації бібліотеки.

Прикладний програмний інтерфейс спільного використання з допомогою віртуальної машини

Деякі мови, серед яких такі, що працюють на віртуальних машинах (наприклад: мови, сумісні з NET CLI середовища CLR і JVM сумісних мов у віртуальній машині Java) можуть ділитися програмними інтерфейсами.

У цьому випадку віртуальна машина дозволяє мові взаємодії завдяки спільному знаменнику віртуальної машини, що абстрагується від конкретної мови, використовувати проміжний байт-код і його мову.

Футбол асоціації, більш відомий як футбол (від англ. football) — це командний вид спорту, який грається між двома командами по одинадцять гравців зі сферичним м'ячем.

У футбол грають на прямокутному полі з воротами на кожному кінці. Мета гри полягає в тому, щоб забити м'яч у ворота протилежної команди. Гравцям не дозволено торкатися м'яча руками, поки він перебуває в грі, якщо вони не воротарі (і тільки тоді, коли він перебуває в їхньому штрафному майданчику), або під час укидання м'яча. Інші гравці здебільшого використовують свої ноги, щоб завдати вдару або передати м'яч, але можуть також використовувати голову і тулуб. Команда, яка забиває більше голів до кінця матчу — виграє його. Якщо жодна команда не забила м'яч, або рахунок однаковий, то оголошується нічия, або гра переходить у додатковий час, або пенальті, залежно від формату змагань.

Грають на футбольному полі завдовжки 90—120 метрів і завширшки 45—90 метрів. На двох протилежних кінцях стоять ворота (завширшки 7,32 м і заввишки 2,44 м), куди потрібно завести м'яча. М'яч, обвід якого повинен

					ІС КР 122 АІ-183 ПЗ	Л и
Змін	Л и	№.	П і д п	Д а		11

бути 68-71 см, а маса — від 396 до 453 грамів. У момент початку гри тиск усередині кулі має бути від 0,6 до 1,1 атмосфери (600—1100 г/см²).

Кількість запасних гравців визначається регламентом змагання і знаходиться у межах від 3 до 7.

Рішення судді є остаточними і під час гри не переглядаються. Він сам може змінити рішення, якщо ще не відновив гру.

Для подачі скарг на дії суддів існує регламент змагання і спеціальні комітети, що розглядають дані питання.

Футбольний матч складається з двох рівних таймів по 45 хвилин із 15-хвилинною перервою між ними. Після перерви команди міняються воротами.

За домовленістю тривалість тайму може бути змінена. Проте домовленостей потрібно досягти до початку матчу, і вони не повинні суперечити правилам змагання. Перерва між таймами не перевищує 15 хвилин й указується у регламенті змагань. Регламент змагань може потребувати додаткового часу для вирішення нічиїх.

За 9 футбольними правилами, м'яч вважається таким, що «вийшов з гри», якщо він цілком виходить за межі поля, або гра зупинена суддею.

Голом називається ситуація, коли м'яч цілком перетнув лінію воріт, розташовану між стійками і під перекладиною, й при цьому команда, що забила гол, не порушила правил.

Порушення караються штрафним або вільним ударом, попередженням або видаленням гравця.

Пенальті — вид штрафного удару у футболі, що пробивається з відстані 12 ярдів (приблизно 11 метрів від воріт). Тільки воротар команди, що захищається, може знаходитись між м'ячем і воротами під час пробиття цього удару. Пенальті пробивається під час звичайної гри. Схожі удари також пробиваються під час післяматчевих пенальті для визначення команди, що проходить у наступний раунд змагань у випадку, якщо матч закінчується внічию. Хоча процес пробиття цих ударів схожий з пробиттям пенальті, вони

					ІС КР 122 АІ-183 ПЗ	Л и
						12
Змін	Л и	№.	П і д п	Д а		

офіційно не вважаються пенальті; процес пробиття цих ударів відбувається згідно з іншими правилами.

Вкидання м'яча є одним із способів відновлення гри. Гол, забитий безпосередньо після вкидання, не зараховується. Вкидання призначається у випадку, коли м'яч повністю перетне бокову лінію поля, з місця, де м'яч перетнув лінію на користь команди-суперниці гравця, що останнім торкнувся м'яча.

Удар від воріт призначається, коли м'яч, останнього разу торкнувшись гравця нападаючої команди, повністю перетнув лінію воріт, і гол не був забитий. Якщо ж м'яч пішов за лінію воріт від гравця команди, що оборонялася, призначається кутовий.

Кутовий удар є одним з найнебезпечніших стандартних положень. Команди часто відпрацьовують тактику у захисті й нападі при пробитті кутових ударів. Оскільки всі відстані заздалегідь відомі, то грамотно пробитий кутовий удар стає прекрасним шансом забити гол. Найчастіше м'яч з кутового навішують у штрафний майданчик, де високі футболісти або прагнуть одразу пробити по воротах, або скинути м'яч під удар партнерові. Рідше подають м'яч низом, але через велике скупчення гравців при кутових опанувати м'ячем буде складніше.

Футбольні змагання проводяться на рівні клубів, аматорських і професійних, а також на рівні національних збірних країн світу.

Найважливішим футбольним змаганням є Чемпіонат світу з футболу, який розігрується під егідою ФІФА кожних чотири роки, і в якому беруть участь національні збірні понад 200-от країн світу. Це змагання складається з кваліфікаційного етапу, на якому відбираються 32 найкращі збірні, й фіналу. Окрім Кубка світу серед чоловіків ФІФА проводить також Кубок світу серед жінок, а також низку молодіжних і юнацьких змагань з віковими обмеженнями для учасників.

					ІС КР 122 АІ-183 ПЗ	Л и
						13
Змін	Л и	№.	П і д п	Д а		

ФІФА відповідає також за проведення Олімпійського футбольного турніру — змагання з футболу в рамках Олімпіад.

У проміжку між фіналами Кубка світу, проводяться континентальні чемпіонати, зокрема в Європі, чемпіонат Європи з футболу.

На клубному рівні найважливішими футбольними змаганнями є національні чемпіонати, в Україні — Чемпіонат України з футболу, та національні кубки, в Україні — Кубок України з футболу. Національні футбольні змагання організовані за системою ліг і дивізіонів й охоплюють гравців усіх рівнів — від аматорів до професіоналів.

Окрім національних клубних змагань існують міжнародні клубні турніри. Найпрестижнішими міжнародними турнірами на клубному рівні є європейська Ліга чемпіонів УЄФА та латиноамериканський Кубок Лібертадорес. Іншим міжнародним кубковим змаганням у Європі до 2009 року був Кубок УЄФА, який з сезону 2009/2010 було реорганізовано в Лігу Європи.

1.2 Бачення системи

Передбачається створення мобільного застосунку для перегляду інформації про футбол. Була визначена основна користувацька роль:

- «користувач» - має можливість переглядати турнірну таблицю з топ 5 чемпіонатів (Англія, Німеччина, Іспанія, Франція, Італія). Також доступна можливість ознайомитись з результатами всіх матчів, які вже пройшли. Ще при натисканні на ім'я команди - буде відкрито вікно з інформацією про дану команду.

1.3 Формування користувацьких історій

Вимоги до мобільного застосунку пред'являються у вигляді користувацьких історій:

					ІС КР 122 АІ-183 ПЗ	Л и
						14
Змін	Л и	№.	П і д п	Д а		

1) Як «користувач» я можу вибрати інформацію про який чемпіонат мені цікаво побачити.

Детальний опис:

– Коли Ви відкриваєте програму буде автоматично завантажено інформацію про чемпіонат Англії.

– Для перемикання між чемпіонатами потрібно відкрити меню через кнопку в лівому верхньому кутку або через свайп вправо.

– Далі буде надана можливість відкрити інформацію про чемпіонат, що цікавить, за допомогою натискання на ім'я одного з 5 доступних чемпіонатів (Англія, Німеччина, Іспанія, Франція, Італія).

Пріоритет: 1.

2) Як «користувач» я можу подивитися результати ігор команд чемпіонату, що цікавить

Детальний опис:

– Після відкриття цікавить чемпіонату свайпом вліво або натисканням на слово «Match», яке знаходиться над турнірною таблицею, буде відображено результати останнього туру.

– Для перегляду результату іншого туру потрібно натиснути на напис «1 tour» (число відображає результат якого туру зараз показується, тому число може відрізнитися), що знаходиться по центру у верхній частині вікна.

– Результат кожного туру буде складатися з 3 компонентів, а саме: по краях дві кнопки, на яких буде написано ім'я двох команд грають один з одним, а по центру результат матчу.

– Якщо матч не буде ще зіграний - поле результату буде білим і в ньому відобразитися напис «-:-»

– Якщо матч буде зіграний - поле результату буде зеленим, а напис буде відображати результат матчу.

					ІС КР 122 АІ-183 ПЗ	Л и
						15
Змін	Л и	№.	П і д п	Д а		

– Якщо матч буде зіграний - поле імені команд перефарбуватися з білого в відповідний колір (червоний - команда програла, жовтий - нічия, а зелений - виграла).

Пріоритет: 1.

3) Як «користувач» я можу переглянути інформацію про команду

Детальний опис:

– При натисканні на ім'я команди в турнірній таблиці або в результатах матчів буде показана інформація про дану команду: ім'я, емблема, головний стадіон, рік створення, кнопка «Go to website» і склад команди.

– При натисканні на кнопку «Go to website» буде відкритий в браузері офіційний сайт даної команди.

Пріоритет: 2.

					ІС КР 122 АІ-183 ПЗ	Л и
						16
Змін	Л и	№.	П і д п	Д а		

ОСНОВНА ЧАСТИНА

2.1 Проектування мобільного застосунку

Після формування користувацьких історій було спроектовано зовнішній вигляд і функціонал мобільного застосунку. Задля цього було обрано онлайн-сервіс Figma. Задля спрощення увесь функціонал було розбито на 3 вікна: турнірна таблиця, результати матчів, інформація про команду.

Перше вікно буде складатися з 4 блоків:

- 1) Значок кнопки «Меню» і назва чемпіонату
- 2) Смужка для переключення з вікна «Турнірна таблиця» на вікно «Результат матчів» і навпаки
- 3) Назва кожної колонки з даними
- 4) Дані про усі команди даного чемпіонату

Результат відображено на рисунку 2.1.1

					ІС КР 122 АІ-183 ПЗ	Л и
						17
Змін	Л и	№.	П і д п	Д а		


England						
STANDINGS				MATCH		
	Team	Form	W	D	L	G P
1	 Chelsea	LWDWW	17	10	7	34 61

Рис. 2.1.1 – Турнірна таблиця

Друге вікно складається теж з 4 блоків, але перші з двох блоків ідентичні. В третьому блоку розміщено випадаючий список, який дозволяє обирати результати якого туру зараз показуються, а в останньому данні про результати ігор. Результат відображено на рисунку 2.1.2

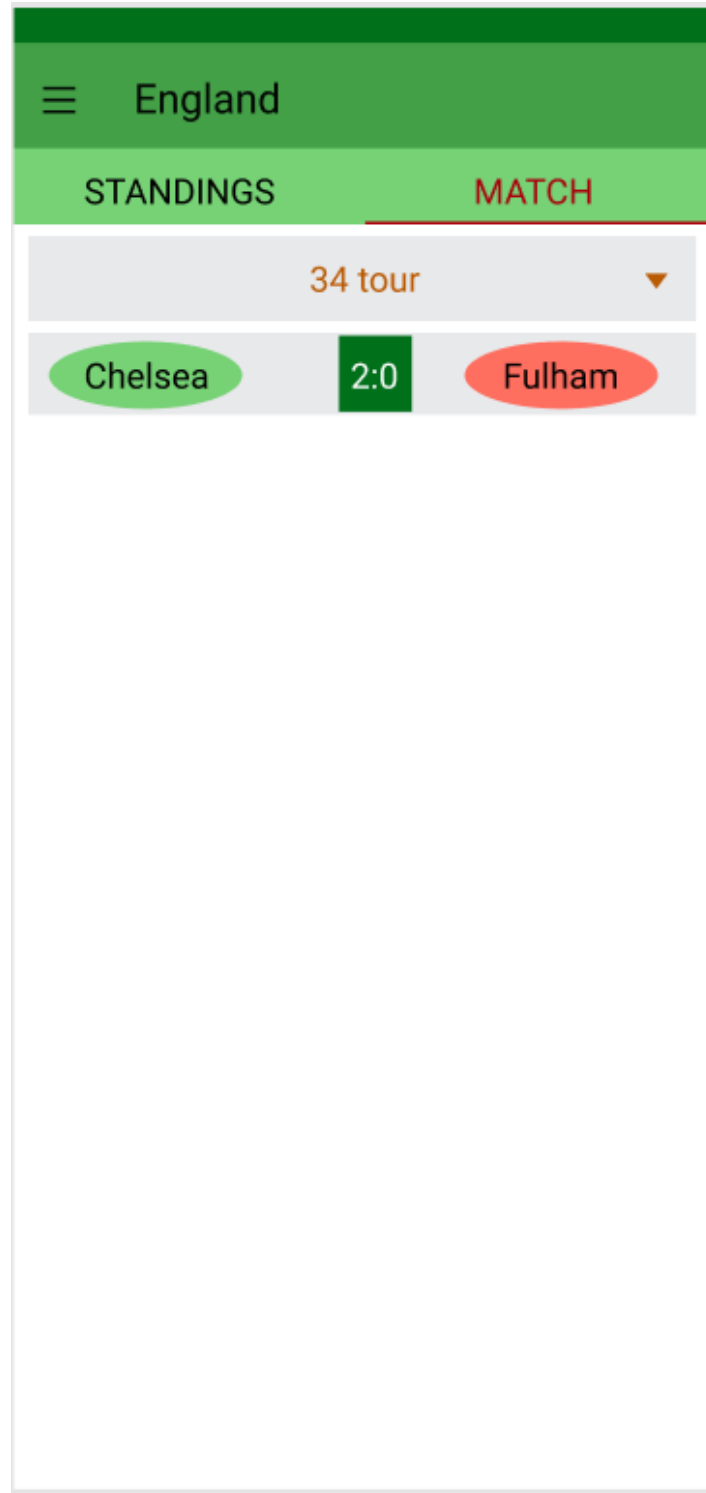


Рис. 2.1.2 – Результат матчів

Останнє вікно можна поділити на 4 умовні блока, а саме:

- 1) Назва команди
- 2) Блок у якому розташовано усю інформацію про цю команду:
 - Емблема команди
 - Їх стадіон
 - Рік створення
 - Кнопка, при натисненні на яку, посилає на офіційний сайт клубу
- 3) Надпис «Squad»
- 4) Інформація про гравців

Результат відображено на рисунку 2.1.3



Рис. 2.1.3 – Результат матчів

2.2 Програмна реалізація

Першим у програмі через маніфест запускається тема SplashScreen:

```
android:theme="@style/Theme.Football2">
```

```
<activity
```

```
    android:name=".SplashScreen"
```

```
    android:theme="@style/SplashScreen">
```

```
    <intent-filter>
```

```
        <action android:name="android.intent.action.MAIN" />
```

```
        <category android:name="android.intent.category.LAUNCHER" />
```

```
    </intent-filter>
```

```
</activity>
```

яка дозволяє користувачу до запуску системи бачити не просто білий екран, а рисунок заставки. Як тільки система буде підготовлена – клас «SplashScreen» запустить клас «MainActivity». Цей клас налаштує кнопку «Меню»:

```
Drawable drawable = ResourcesCompat.getDrawable(getResources(),  
R.drawable.ic_baseline_menu_24, null);
```

```
drawable = DrawableCompat.wrap(drawable);
```

```
DrawableCompat.setTint(drawable, Color.BLACK);
```

```
getSupportActionBar().setDisplayHomeAsUpEnabled(true);
```

та в залежності від обраного пункту меню передасть відповідні данні у клас «Container»:

```
container = Container.getInstance(Country.ENGLAND);
```

```
toolbar.setTitle(R.string.menu_england);
```

```
getSupportFragmentManager().beginTransaction().replace(R.id.nav_host_f  
ragment, container, null).commit();
```

Надалі клас налаштовує адаптер в залежності від чемпіонату та передає у нього два класу «Standings» та «Matches»:

					IC KP 122 AI-183 ПЗ	Л и
						21
Змін	Л и	№.	П і д п	Д а		

```

AdapterContainer adapter = new AdapterContainer(view.getContext());
adapter.addFragment(new Standings(Country.ENGLAND), "Standings");
adapter.addFragment(new Matches(Country.ENGLAND), "Match");

```

Ці класи відповідають турнірній таблиці та результатам матчів відповідно. Обидва класи запускають свої вікна:

```

getUrlData.execute(constRequest.getStandingsEnglandUrl());

```

та в залежності від чемпіонату передають данні до класу «GetData», який відповідає за усі данні (в залежності від запиту отримує данні та передає їх у класи, які в свою чергу формують будь-які переліки та відображають їх).

При натисканні на назву команди відкривається вікно з інформацією про цю команду. Це виникає через виклик класу «InfoTeam» який відкриває своє вікно та знов таки передає данні до класу «GetData»:

```

TextView nameTeam = findViewById(R.id.name_team);
TextView venue = findViewById(R.id.venue);
TextView founded = findViewById(R.id.founded);
Button website_btn = findViewById(R.id.website_btn);
RecyclerView rv = findViewById(R.id.squad_rv);
ImageView imageView = findViewById(R.id.icon_team);
DataActivity data = new DataActivity(this, rv, TypeData.INFO);
DataInfoTeam dataInfoTeam = new DataInfoTeam(nameTeam, venue,
founded, website_btn, imageView);

GetData getUrlData = new GetData(data, dataInfoTeam);
GetRequest constRequest = new GetRequest();
getUrlData.execute(constRequest.getTeamById()
+
getIntent().getStringExtra("id"));

```

2.3 Інструкція для користувача

При запуску мобільного застосунку відкривається заставка (рис 2.3.1)

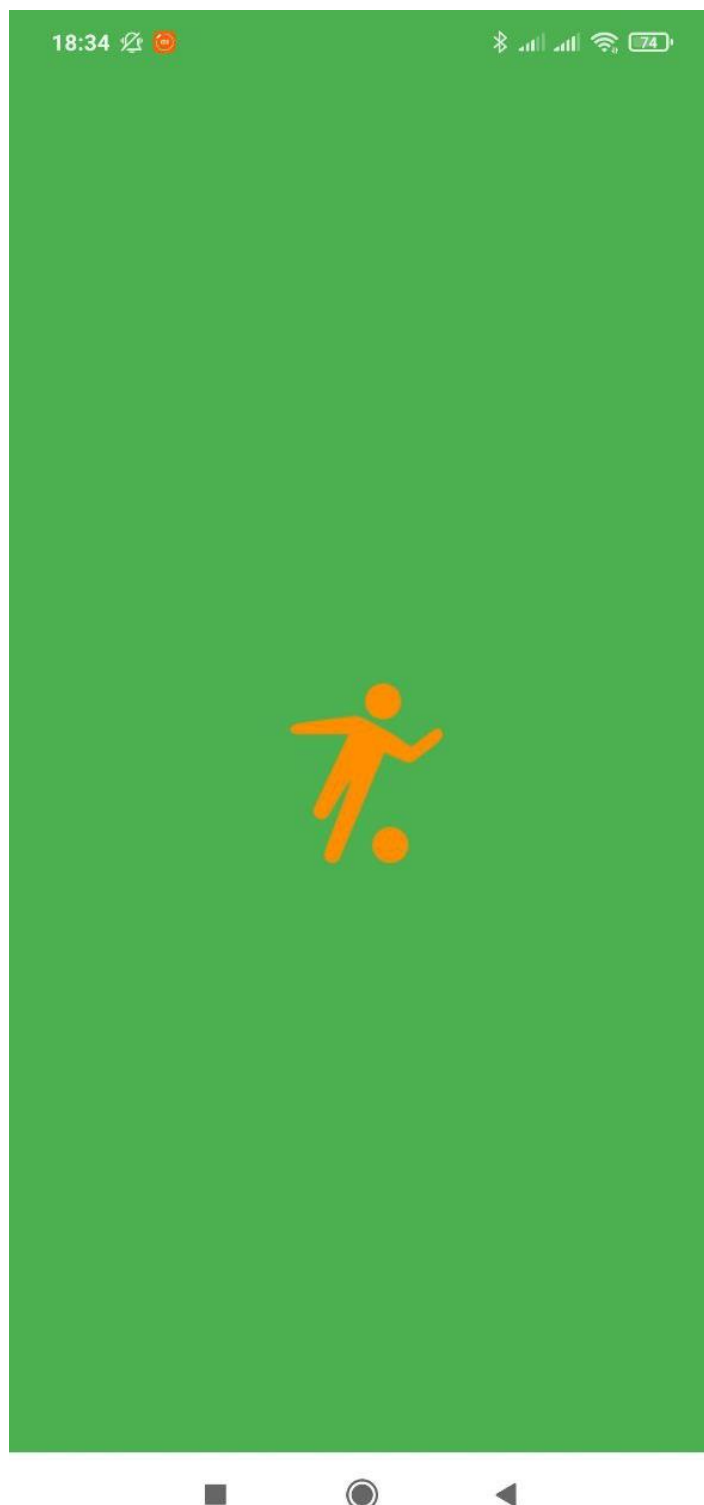


Рис. 2.3.1 – Заставка

					ІС КР 122 АІ-183 ПЗ	Л и
Змін	Л и	№.	П і д п	Д а		23

Після налаштування пристрою відкривається турнірна таблиця Англії, в якій буде відображено інформацію про позицію команди, її емблема, результати останніх 5 матчів, кількість виграних, програних та нічийних матчів, кількість ігор та кількість отриманих очок(рис. 2.3.2).

	Team	Form	W	D	L	G	P
1	Manchester City	WLWLW	27	5	6	38	86
2	Manchester United	WLLDW	21	11	6	38	74
3	Liverpool	WWWWW	20	9	9	38	69
4	Chelsea	WWLWL	19	10	9	38	67
5	Leicester City	DLWLL	20	6	12	38	66
6	West Ham United	WLDWW	19	8	11	38	65
7	Tottenham Hotspur	WLWLW	18	8	12	38	62
8	Arsenal	WWWWW	18	7	13	38	61
9	Leeds United	LWWWW	18	5	15	38	59
10	Everton	WDLWL	17	8	13	38	59
11	Aston Villa	LDLWW	16	7	15	38	55

Рис. 2.3.2 – Відображення турнірної таблиці

Далі після свайпу вправо або натисненні кнопки «Match» буде показано результати матчів останнього туру: які команди грали, результати матчів та поле з іменем команди окрашене у відповідний колір (червоне – команда програла, жовта – нічия, а зелена – перемогла) (рис. 2.3.3).



Рис. 2.3.3 – Відображення результату матчу

Для отримання результатів матчу іншого туру потрібно натиснути на надпис «38 tour» (рис. 2.3.4) (число може відрізнятися, так як воно відображає номер обраного туру) та обрати потрібний тур з усіх запропонованих.

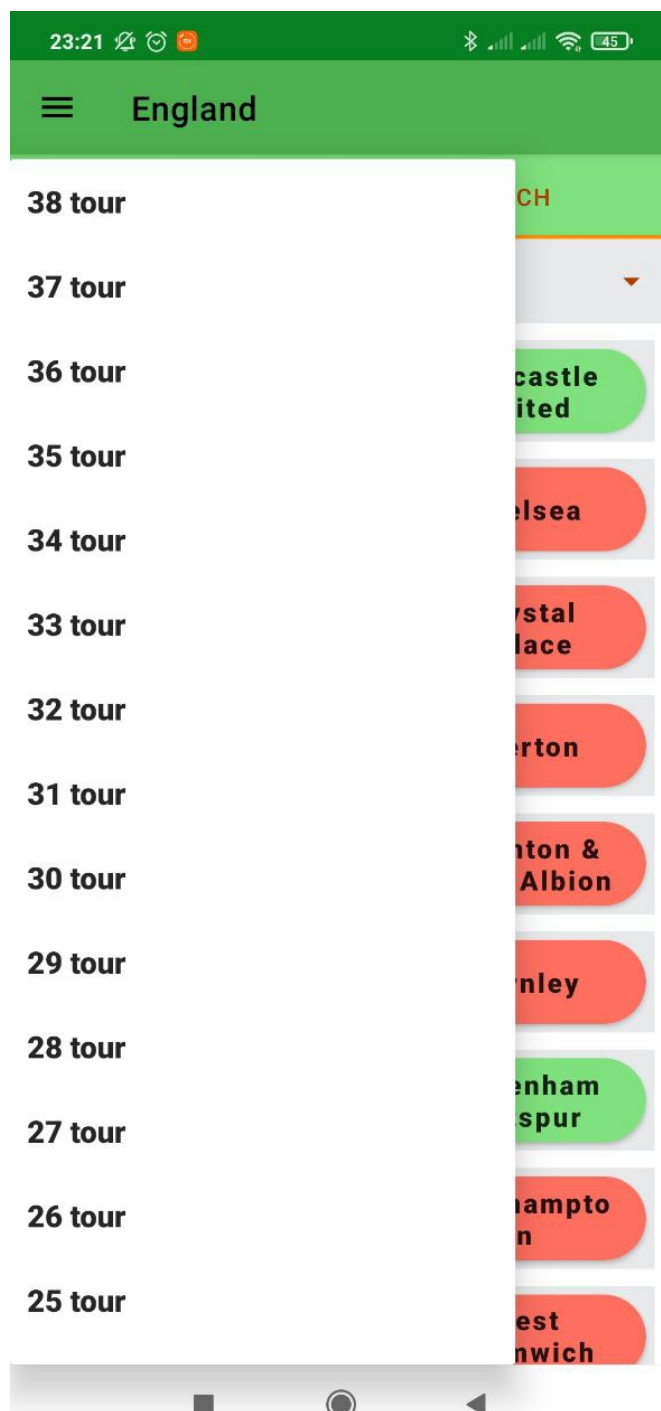


Рис. 2.3.4 – Відображення інформації про команду

Обравши тур, що цікавить (в даному випадку 25) буде заново відображено інформацію про результати матчів, але вже нового обраного туру (рис. 2.3.5).

England		
STANDINGS MATCH		
25 tour		
Wolverhampton	1:0	Leeds United
Southampton	1:1	Chelsea
Burnley	0:0	West Bromwich
Liverpool	0:2	Everton
Fulham	1:0	Sheffield United
West Ham United	2:1	Tottenham Hotspur
Aston Villa	1:2	Leicester City
Arsenal	0:1	Manchester City
Manchester United	3:1	Newcastle United

Рис. 2.3.5 – Відображення результату матчу іншого туру

Свайпом вліво знов поверне до турнірній таблиці. За для обрання іншого туру потрібно відкрити меню турів. Для цього свайпом вліво на турнірній таблиці або натисканням на кнопку у верхньому лівому куті буде відкрито меню (рис. 2.3.6).

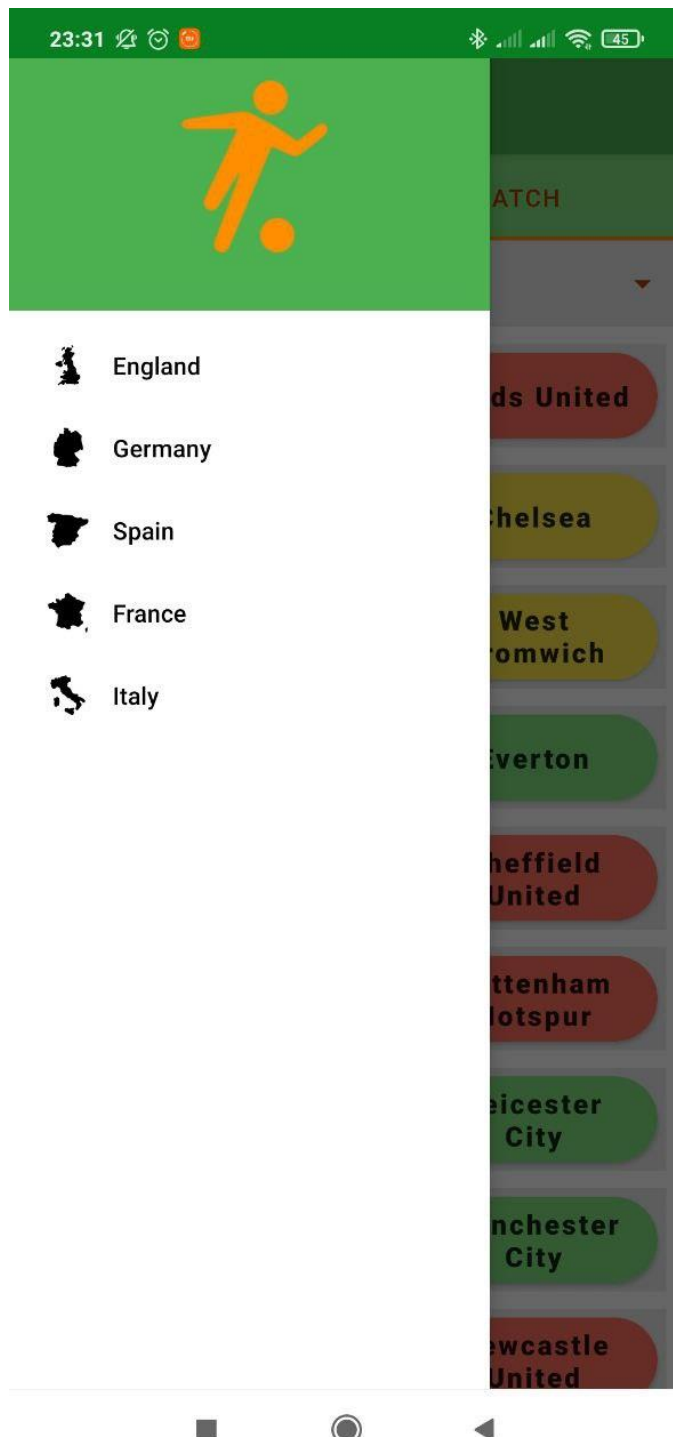


Рис. 2.3.6 – Зміна обраного чемпіонату

Для отримання інформації про клуб потрібно натиснути на її ім'я у будь-якому місті. Буде відображено назва команди, емблема, назва стадіону, рік створення, кнопка для переходу на офіційний сайт клубу (рис. 2.3.7), а також повний перелік складу команди: ім'я, позиція або посада (тренер, асистент тренера), громадянство та рік народження (рис. 2.3.8).

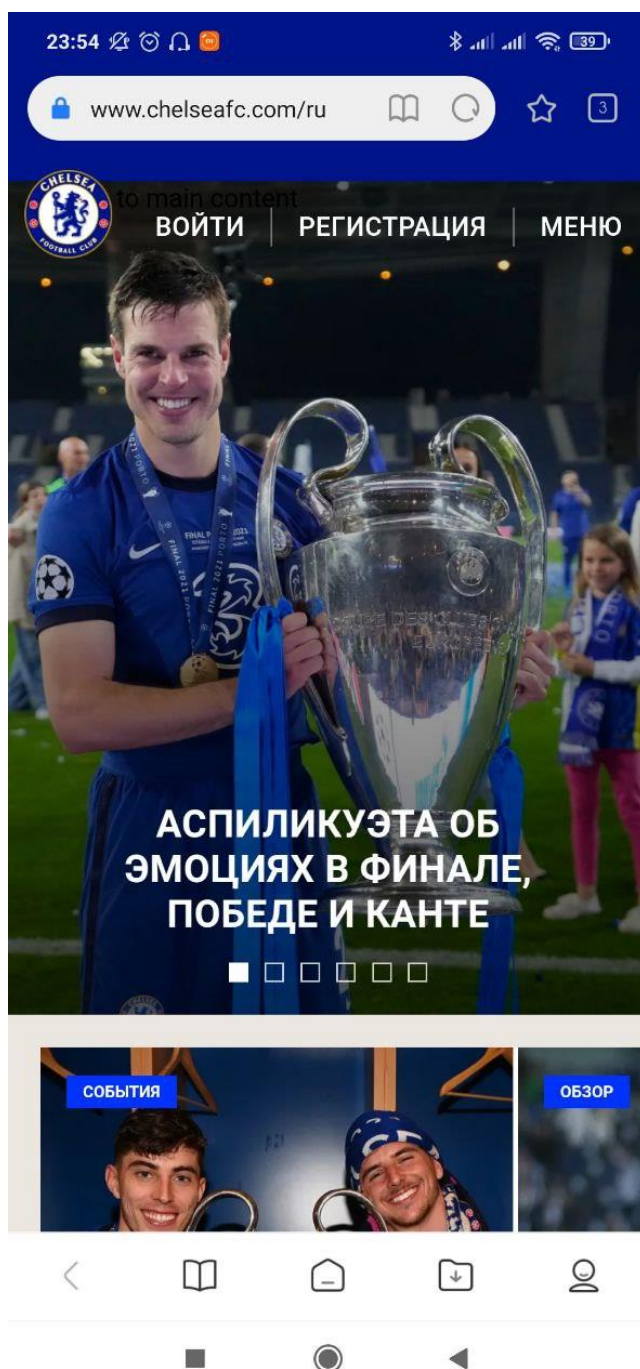


Рис. 2.3.7 – Офіційний сайт клубу

					ІС КР 122 АІ-183 ПЗ	Л и
Змін	Л и	№.	П і д п	Д а		29

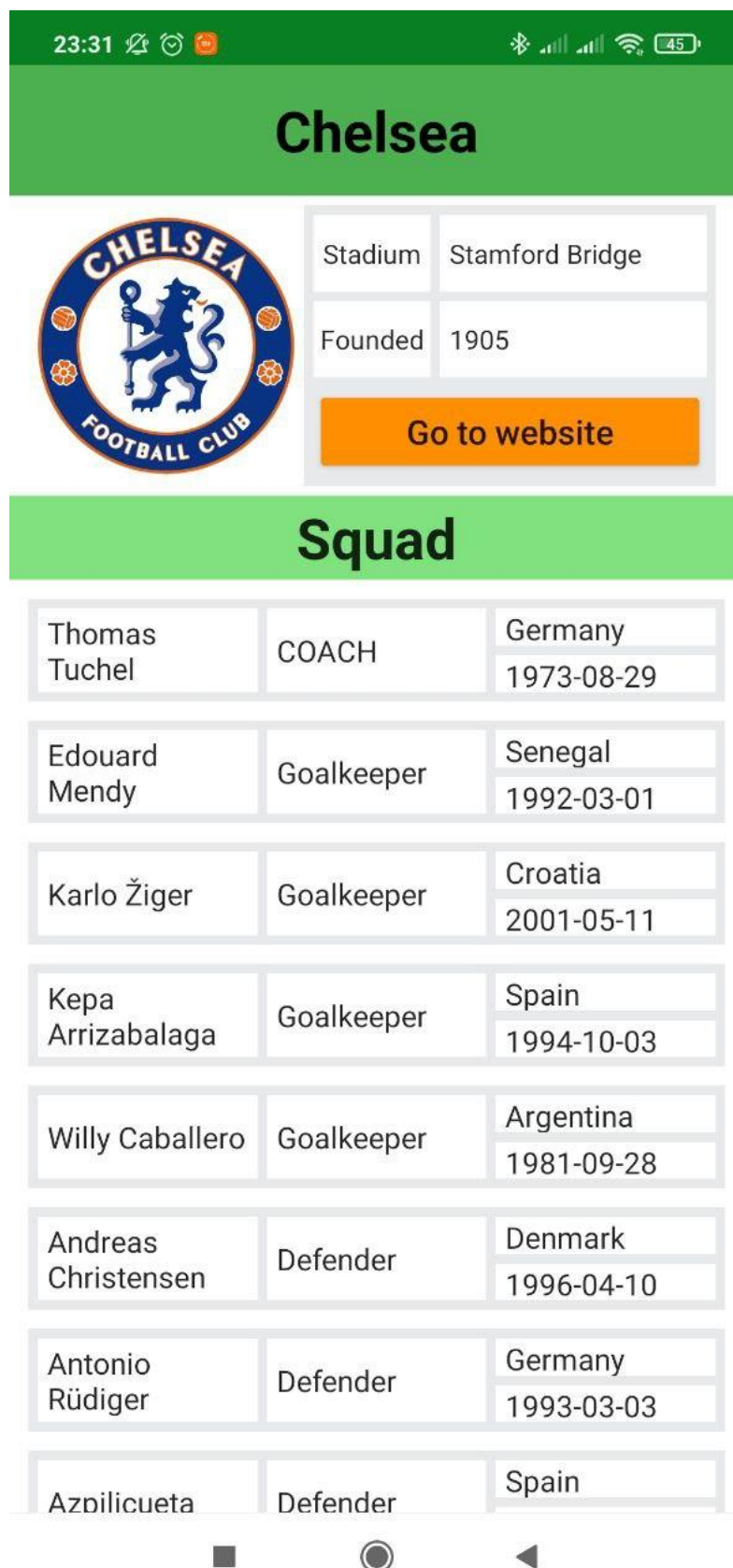


Рис. 2.3.8 – Відображення інформації про команду

ВИСНОВОК

У ході виконання курсової роботи були закріпленні, на практиці, теоретичні знання з курсу програмування мобільних пристроїв та були отриманні практичні навички щодо проектування, розробки програми для мобільної розробки. Були отримані навички роботи з активіті, адаптерами, списками, фрагментами та багато іншим для платформи Android. Були отриманні навички роботи з API та отримання даних з JSON БД. Також було визначено як отримувати та відображати зображення через url. Отриманні практичні навички у роботі з Java.

					ІС КР 122 АІ-183 ПЗ	Л и
						31
Змін	Л и	№.	П і д п	Д а		

ПЕРЕЛІК ПОСИЛАНЬ

1. football-data.org [Електронний ресурс] – Режим доступу:
<https://www.football-data.org/index>
2. Developers [Електронний ресурс] – Режим доступу:
<https://developer.android.com/guide?hl=ru>
3. Figma [Електронний ресурс] – Режим доступу:
<https://www.figma.com/>
4. COLOR TOOL [Електронний ресурс] – Режим доступу:
<https://material.io/resources/color/#!/?view.left=0&view.right=0&secondary.color=0288D1&primary.color=4CAF50>

					ІС КР 122 АІ-183 ПЗ	Л и
						32
Змін	Л и	№.	П і д п	Д а		

ДОДАТОК А

Клас «Container»:

@Override

```
public View onCreateView(LayoutInflater inflater, ViewGroup container,
                          Bundle savedInstanceState) {
    return inflater.inflate(R.layout.container, container, false);
}
```

@Override

```
public void onViewCreated(@NonNull View view, @Nullable Bundle savedInstanceState) {
    super.onViewCreated(view, savedInstanceState);
    ViewPager viewPager = view.findViewById(R.id.container);
    TabLayout tabLayout = view.findViewById(R.id.tab_layout);
    Country country = Country.valueOf(getArguments().getString("country", ""));

    AdapterContainer adapter = new AdapterContainer(view.getContext());
    switch (country){
        case ENGLAND:
            adapter.addFragment(new Standings(Country.ENGLAND), "Standings");
            adapter.addFragment(new Matches(Country.ENGLAND), "Match");
            break;
        case GERMANY:
            adapter.addFragment(new Standings(Country.GERMANY), "Standings");
            adapter.addFragment(new Matches(Country.GERMANY), "Match");
            break;
        case SPAIN:
            adapter.addFragment(new Standings(Country.SPAIN), "Standings");
            adapter.addFragment(new Matches(Country.SPAIN), "Match");
            break;
        case FRANCE:
            adapter.addFragment(new Standings(Country.FRANCE), "Standings");
            adapter.addFragment(new Matches(Country.FRANCE), "Match");
            break;
        default:
            adapter.addFragment(new Standings(Country.ITALY), "Standings");
            adapter.addFragment(new Matches(Country.ITALY), "Match");
            break;
    }
}
```

					ІС КР 122 АІ-183 ПЗ	Л и
						33
Змін	Л и	№.	П і д п	Д а		

```

viewPager.setAdapter(adapter);

tabLayout.setupWithViewPager(viewPager);
}

public static Container getInstance(Country country){
    Bundle bundle = new Bundle();
    bundle.putString("country", String.valueOf(country));
    Container container = new Container();
    container.setArguments(bundle);
    return container;
}

```

Клас «Matches»:

```

private Country country;
private Spinner spinner;
private RecyclerView rv;

public Matches(Country country) {
    this.country = country;
}

public View onCreateView(@NonNull LayoutInflater inflater,
                        ViewGroup container, Bundle savedInstanceState) {

    return inflater.inflate(R.layout.matches, container, false);
}

@Override
public void onViewCreated(@NonNull View view, @Nullable Bundle savedInstanceState) {
    super.onViewCreated(view, savedInstanceState);
    spinner = view.findViewById(R.id.spinner);
    rv = view.findViewById(R.id.matches_rv);

    DataActivity data = new DataActivity(getContext(), rv, TypeData.SPINNER);
    DataMatch dataMatch = new DataMatch(spinner, country);

    GetUrlData getUrlData = new GetUrlData(data, dataMatch);

```

					IC KP 122 AI-183 ПЗ	Л и
						34
Змін	Л и	№.	П і д п	Д а		

```

GetRequest constRequest = new GetRequest();

switch (country){
    case ENGLAND:
        getUrlData.execute(constRequest.getAllMatchesEngland());
        break;
    case GERMANY:
        getUrlData.execute(constRequest.getAllMatchesGermany());
        break;
    case SPAIN:
        getUrlData.execute(constRequest.getAllMatchesSpain());
        break;
    case FRANCE:
        getUrlData.execute(constRequest.getAllMatchesFrance());
        break;
    default:
        getUrlData.execute(constRequest.getAllMatchesItaly());
        break;
}
}

```

Клас «Standings»:

```

private Country country;

public Standings(Country country) {
    this.country = country;
}

public View onCreateView(@NonNull LayoutInflater inflater,
                        ViewGroup container, Bundle savedInstanceState) {

    return inflater.inflate(R.layout standings, container, false);
}

@Override
public void onViewCreated(@NonNull View view, @Nullable Bundle savedInstanceState) {
    super.onViewCreated(view, savedInstanceState);
}

```

```

        DataActivity data = new DataActivity(getContext(), view.findViewById(R.id.standings),
        TypeData.STANDINGS);

        GetUrlData getUrlData = new GetUrlData(data);
        GetRequest constRequest = new GetRequest();

        switch (country){
            case ENGLAND:
                getUrlData.execute(constRequest.getStandingsEnglandUrl());
                break;
            case GERMANY:
                getUrlData.execute(constRequest.getStandingsGermanyUrl());
                break;
            case SPAIN:
                getUrlData.execute(constRequest.getStandingsSpainUrl());
                break;
            case FRANCE:
                getUrlData.execute(constRequest.getStandingsFranceUrl());
                break;
            default:
                getUrlData.execute(constRequest.getStandingsItalyUrl());
                break;
        }
    }
}

```

Клас «MainActivity»:

```

private Toolbar toolbar;
private DrawerLayout drawerLayout;

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    toolbar = findViewById(R.id.toolbar);
    drawerLayout = findViewById(R.id.drawer_layout);

    getSupportFragmentManager().beginTransaction().replace(R.id.nav_host_fragment,
    Container.getInstance(Country.ENGLAND), null).commit();
    toolbar.setTitle(R.string.menu_england);
}

```

```

Drawable drawable = ResourcesCompat.getDrawable(getResources(), R.drawable.ic_baseline_menu_24, null);
drawable = DrawableCompat.wrap(drawable);
DrawableCompat.setTint(drawable, Color.BLACK);

setSupportActionBar(toolbar);
getSupportActionBar().setHomeAsUpIndicator(drawable);
getSupportActionBar().setDisplayHomeAsUpEnabled(true);
new ActionBarDrawerToggle(this, drawerLayout,
    toolbar, R.string.navigation_drawer_open, R.string.navigation_drawer_close);

NavigationView navigationView = findViewById(R.id.nav_view);
navigationView.setNavigationItemSelectedListener(this::navigationItemSelected);
}

private boolean navigationItemSelected(MenuItem item) {
    Container container;
    switch (item.getItemId()){
        case R.id.nav_england:
            container = Container.getInstance(Country.ENGLAND);
            toolbar.setTitle(R.string.menu_england);
            break;
        case R.id.nav_germany:
            container = Container.getInstance(Country.GERMANY);
            toolbar.setTitle(R.string.menu_germany);
            break;
        case R.id.nav_spain:
            container = Container.getInstance(Country.SPAIN);
            toolbar.setTitle(R.string.menu_spain);
            break;
        case R.id.nav_france:
            container = Container.getInstance(Country.FRANCE);
            toolbar.setTitle(R.string.menu_france);
            break;
        default:
            container = Container.getInstance(Country.ITALY);
            toolbar.setTitle(R.string.menu_italy);
            break;
    }
    getSupportFragmentManager().beginTransaction().replace(R.id.nav_host_fragment, container, null).commit();
    return true;
}

```

					IC KP 122 AI-183 ПЗ	Л и
						37
Змін	Л и	№.	П і д п	Д а		

```
}
```

Клас «InfoTeam»:

```
@Override
protected void onCreate(@Nullable Bundle savedInstanceState){
    super.onCreate(savedInstanceState);
    setContentView(R.layout.info_team);

    TextView nameTeam = findViewById(R.id.name_team);
    TextView venue = findViewById(R.id.venue);
    TextView founded = findViewById(R.id.founded);
    Button website_btn = findViewById(R.id.website_btn);
    RecyclerView rv = findViewById(R.id.squad_rv);
    ImageView imageView = findViewById(R.id.icon_team);

    DataActivity data = new DataActivity(this, rv, TypeData.INFO);
    DataInfoTeam dataInfoTeam = new DataInfoTeam(nameTeam, venue, founded, website_btn, imageView);

    GetUrlData getUrlData = new GetUrlData(data, dataInfoTeam);
    GetRequest constRequest = new GetRequest();
    getUrlData.execute(constRequest.getTeamById() + getIntent().getStringExtra("id"));
}
```

Клас «GetUrlData»:

```
private DataActivity dataActivity;
private DataMatch dataMatch;
private DataInfoTeam dataInfoTeam;

private final AdapterView.OnItemClickListener itemSelectedListener = new
AdapterView.OnItemClickListener() {
    @Override
    public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
        int matchDay = Integer.parseInt(((String)parent.getItemAtPosition(position)).split(" ")[0]);

        GetUrlData getUrlData = new GetUrlData(new DataActivity(dataActivity.getContext(),
dataActivity.getRecyclerView(), TypeData.MATCH));
        GetRequest constRequest = new GetRequest();

        switch (dataMatch.getCountry()){
```

					IC KP 122 AI-183 ПЗ	Л и
						38
Змін	Л и	№.	П і д п	Д а		

```

        case ENGLAND:
            getUrlData.execute(constRequest.getMatchesEnglandByTour() + matchDay);
            break;
        case GERMANY:
            getUrlData.execute(constRequest.getMatchesGermanyByTour() + matchDay);
            break;
        case SPAIN:
            getUrlData.execute(constRequest.getMatchesSpainByTour() + matchDay);
            break;
        case FRANCE:
            getUrlData.execute(constRequest.getMatchesFranceByTour() + matchDay);
            break;
        default:
            getUrlData.execute(constRequest.getMatchesItalyByTour() + matchDay);
            break;
    }
}

@Override
public void onNothingSelected(AdapterView<?> parent) {
}

};

public GetUrlData(DataActivity dataActivity) {
    this.dataActivity = dataActivity;
}

public GetUrlData(DataActivity dataActivity, DataMatch dataMatch) {
    this.dataActivity = dataActivity;
    this.dataMatch = dataMatch;
}

public GetUrlData(DataActivity dataActivity, DataInfoTeam dataInfoTeam){
    this.dataActivity = dataActivity;
    this.dataInfoTeam = dataInfoTeam;
}

@Override
protected String doInBackground(String... strings) {
    HttpURLConnection connection = null;
    BufferedReader reader = null;

```

```

try {
    URL url = new URL(strings[0]); // открыли URL соединение
    connection = (URLConnection) url.openConnection(); // открыли Http соединение
    connection.setRequestProperty(new GetRequest().getKey(), new GetRequest().getValue());
    connection.connect();

    InputStream stream = connection.getInputStream(); // считали поток
    reader = new BufferedReader(new InputStreamReader(stream));

    StringBuffer buffer = new StringBuffer(); // обычная строка, но лучше подходит для работы с
    BufferedReader

    String line;

    while ((line = reader.readLine()) != null) { // считываем текст построчно
        buffer.append(line).append("\n"); //записываем строки в переменную и добавляем конец
        строки
    }

    return buffer.toString();
} catch (IOException e) {
    e.printStackTrace();
} finally { // закрываем потоки
    if(connection != null) {
        connection.disconnect();
    }
    try {
        if(reader != null) {
            reader.close();
        }
    } catch (IOException e){
        e.printStackTrace();
    }
}

return null;
}

@Override
protected void onPostExecute(String result){

```

					ІС КР 122 АІ-183 ПЗ	Л и
						40
Змін	Л и	№.	П і д п	Д а		


```

super.onPostExecute(result);
if (result == null){
    return;
}
try {
    if (dataActivity.getTypeData() != TypeData.SPINNER) {
        dataActivity.getRecyclerView().setHasFixedSize(true);
        dataActivity.getRecyclerView().setLayoutManager(new
LinearLayoutManager(dataActivity.getContext()));
    }
    switch (dataActivity.getTypeData()) {
        case STANDINGS:
            dataActivity.getRecyclerView().setAdapter(
                new AdapterListStandings(dataActivity.getContext(), createTeams(new
JSONObject(result))));
            break;
        case MATCH:
            dataActivity.getRecyclerView().setAdapter(
                new AdapterListMatch(dataActivity.getContext(), createMatch(new
JSONObject(result))));
            break;
        case INFO:
            dataActivity.getRecyclerView().setAdapter(
                new AdapterListSquad(dataActivity.getContext(), setInfoTeam(new
JSONObject(result))));
            break;
        default:
            setSpinner(new JSONObject(result));
            break;
    }
} catch (JSONException e) {
    e.printStackTrace();
}
}

private List<ListStandings> createTeams(JSONObject jsonObject) throws JSONException {
    List<ListStandings> teams = new ArrayList<>();
    JSONArray jsonArray =
    jsonObject.getJSONArray("standings").getJSONObject(0).getJSONArray("table");
    for (int i = 0; i < jsonArray.length(); i++){

```

					IC KP 122 AI-183 ПЗ	Л и
						41
Змін	Л и	№.	П і д п	Д а		

```

        teams.add(new ListStandings(jsonArray.getJSONObject(i)));
    }
    return teams;
}

private List<ListMatch> createMatch(JSONObject jsonObject) throws JSONException {
    List<ListMatch> teams = new ArrayList<>();
    JSONArray jsonArray = jsonObject.getJSONArray("matches");
    for (int i = 0; i < jsonArray.length(); i++){
        teams.add(new ListMatch(jsonArray.getJSONObject(i)));
    }
    return teams;
}

private void setSpinner(JSONObject jsonObject) throws JSONException {
    JSONArray jsonArray = jsonObject.getJSONArray("matches");
    int                                     numberLastTour
jsonArray.getJSONObject(0).getJSONObject("season").getInt("currentMatchday");

    String[] array = new String[numberLastTour];
    for(int i = 0; i < numberLastTour; i++){
        array[i] = (numberLastTour - i) + " tour";
    }

    ArrayAdapter<String> adapter = new ArrayAdapter<>(dataActivity.getContext(),
        R.layout.spinner_item, array);
    adapter.setDropDownViewResource(R.layout.spinner_dropdown_item);
    dataMatch.getSpinner().setAdapter(adapter);
    dataMatch.getSpinner().setOnItemSelectedListener(itemSelectedListener);
}

private List<ListSquad> setInfoTeam(JSONObject jsonObject) throws JSONException{
    dataInfoTeam.getNameTeam().setText(jsonObject.getString("name")
        .replaceAll(new GetRequest().getRegexNameTeam(), "")
        .replace("1.", "")
        .trim().replaceAll("\\s{2,}", " "));

    Utils.fetchSvg(dataActivity.getContext(),
        jsonObject.getString("crestUrl"),
    dataInfoTeam.getImageView());
}

```

					IC KP 122 AI-183 ПЗ	Л и
						42
Змін	Л и	№.	П і д п	Д а		

```

dataInfoTeam.getVenue().setText(jsonObject.getString("venue"));
dataInfoTeam.getFounded().setText(jsonObject.getString("founded"));
dataInfoTeam.getWebsite().setOnClickListener(v -> {
    try {
        Intent browserIntent = new Intent(Intent.ACTION_VIEW,
Uri.parse(jsonObject.getString("website")));
        dataActivity.getContext().startActivity(browserIntent);
    } catch (JSONException e) {
        e.printStackTrace();
    }
});

List<ListSquad> squad = new ArrayList<>();
JSONArray jsonArray = jsonObject.getJSONArray("squad");
for (int i = 0; i < jsonArray.length(); i++){
    squad.add(new ListSquad(jsonArray.getJSONObject(i)));
}

List<String> compare = Arrays.asList("COACH", "ASSISTANT_COACH", "Goalkeeper",
"Defender", "Midfielder", "Attacker");
Collections.sort(squad, (object1, object2) -> {
    Integer firstId = compare.indexOf(object1.getPosition());
    Integer secondId = compare.indexOf(object2.getPosition());
    if (firstId.compareTo(secondId) == 0) {
        return object1.getName().compareTo(object2.getName());
    } else {
        return firstId.compareTo(secondId);
    }
});
return squad;
}

```

