

Міністерство освіти і науки України
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»
Кафедра інформатики факультету інформатики

ОСОБЛИВОСТІ РОЗРОБКИ ПІД SMART TV

Текстова частина до курсової роботи
за спеціальністю „Інженерія програмного забезпечення” 121

Керівник курсової роботи
к.т.н., ст. вик. Шабінська М.О.
(прізвище та ініціали)

(підпис)
“ ____ ” _____ 2020 р.

Виконав студент _____
Фомін В.О.
(прізвище та ініціали)
“ ____ ” _____ 2020 р.

Київ 2020

Міністерство освіти і науки України
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»

Кафедра інформатики факультету інформатики

ЗАТВЕРДЖУЮ

Зав.кафедри інформатики,

проф., д.ф.-м.н.

_____ С. С. Гороховський

(підпис)

„_____” _____ 2020 р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ

на курсову роботу

студенту Фоміну В.О. факультету Інформатики Третього курсу

ТЕМА Особливості розробки на Smart TV

Вихідні дані:

-

-

Зміст ТЧ до курсової роботи:

Індивідуальне завдання

Вступ

1 Що таке Smart TV

2 Перелік основних платформ Smart TV

3 Огляд основних рис Smart TV та стан платформ на 2020 рік

4 Перелік основних проблем розробки на Smart TV та їх рішення

5 Розробка власного застосунку та висвітлення проблем з якими довелося зіткнутись

Висновки

Список літератури

Додатки (за необхідністю)

Дата видачі „_____” _____ 2019 р. Керівник _____

(підпис)

Завдання отримав _____

(підпис)

Тема: ОСОБЛИВОСТІ РОЗРОБКИ ПІД SMART TV

Календарний план виконання роботи:

№	Назва етапу курсового проекту (роботи)	Термін виконання
1.	Отримання завдання на курсову роботу	листопад
2.	Огляд літератури та матеріалів щодо існуючих платформ Smart TV, загальних особливостей, обмежень та способів розробки під ці платформи	грудень-січень
3.	Створення практичної частини роботи – застосування який транслює музику з інтернет радіо	січень-кінець березня
4.	Підведення підсумків та написання текстової частини	січень-початок квітня
6.	Надання роботи керівнику для перевірки	кінець квітня
8.	Коригування роботи за результатами перевірки	кінець квітня-початок травня
9	Подання роботи на кафедру для перевірки на плагіат	щонайменше за два тижні до захисту
10.	Захист курсової роботи	середина травня

Студент Фомін Володимир Олегович

Керівник Шабінська Марія Олегівна

“ ”

Зміст

АНОТАЦІЯ.....	4
ВСТУП	5
1. ЩО TAKE SMART TV.....	6
2. ПЕРЕЛІК ОСНОВНИХ ПЛАТФОРМ SMART TV.....	6
3. ОГЛЯД ОСНОВНИХ РИС SMART TV ТА СТАН ПЛАТФОРМ НА 2020 РІК	8
4. ПЕРЕЛІК ОСНОВНИХ ПРОБЛЕМ РОЗРОБКИ НА SMART TV ТА ЇХ РІШЕННЯ	11
5. РОЗРОБКА ВЛАСНОГО ЗАСТОСУНКУ ТА ВИСВІТЛЕННЯ ПРОБЛЕМ З ЯКИМИ ДОВЕЛОСЬ ЗІТКНУТИСЬ	16
5.1. ОПИС ЗАСТОСУНКУ ТА ОБҐРУНТУВАННЯ ТЕМИ Й ОБРАНИХ ТЕХНОЛОГІЙ	16
5.2. ОПИС ЗАСОБІВ РОЗРОБКИ	16
5.3. ПРОЕКТУВАННЯ ГРАФІЧНОГО ІНТЕРФЕЙСУ	17
5.4. ЗБЕРЕЖЕННЯ ТА ЗЧИТУВАННЯ ДАНИХ ТА ЇХ ПРЕДСТАВЛЕННЯ	18
5.5. АРХІТЕКТУРА ЗАСТОСУНКУ	20
5.6. РОБОТА ЗАСТОСУНКУ	24
5.7. ПРОБЛЕМИ ЯКІ ВИНИКЛИ ПРИ РОЗРОБЦІ.....	25
ВИСНОВКИ	27
СПИСОК ЛІТЕРАТУРИ.....	29

Анотація

Проаналізовано поняття Smart TV, в чому його користь та стан ринку Smart TV. Подано перелік найпопулярніших платформ Smart TV та технологій розробки під них. Також проаналізовано і перелічено загальні особливості різних платформ з точки зору користувача та розробника, проблеми розробки та рішення цих проблем. Також розроблено застосунок під одну з платформ Smart TV і надано опис процесу розробки для отримання власного практичного досвіду і аналізу проблем з якими довелося зіткнутись і рішень цих проблем.

Ключові слова: Smart TV, Платформа, Технологія, Розробка, Проблема, Особливість, Способи, Додаток, Застосунок, Телевізор, Рішення.

Вступ

Актуальність теми дослідження. Нині розвивається багато різних платформ Smart TV, вони стають більш зручними, потужними, ефективними, до них додається багато нових можливостей. Крім цього, Smart TV користується популярністю, за останні 4 роки кількість користувачів Smart TV у США зросла на 45 млн досягнувши цифри у 109 млн [1]. Зазвичай, Smart TV користуються для перегляду відео онлайн, прослуховування музики, інтернет серфінгу, або того ж самого перегляду телевізійних каналів, але за допомогою технології під назвою Internet Protocol Television (IPTV). Також Smart TV дозволяє завантаження сторонніх додатків, розширюючи свою функціональність. Саме тому актуальним є висвітлення підводних каменів з якими стикаються розробники додатків для Smart TV.

Метою курсової роботи на тему “Особливості розробки на Smart TV” є розкриття основних проблем з якими стикаються розробники застосунків для різних Smart TV, як їх подолати, які основні платформи існують на сьогодні та їх особливості.

Для досягнення цієї мети необхідно вирішити наступні завдання:

1. Дослідити поняття Smart TV.
2. Дослідити існуючі платформи Smart TV та їх особливості як розробки застосунків для них, так і користування ними.
3. Дослідити статті розробників застосунків щодо проблем з якими вони зіткнулися під час розробки своїх додатків та як вони їх вирішили.
4. Розробити застосунок для однієї з платформ Smart TV та надати у письмовому вигляді процес розробки для його детального аналізу.

До курсової роботи з теми “Особливості розробки на Smart TV” об’єктом дослідження є платформи Smart TV. Предметами дослідження є проблеми розробки на Smart TV та способи їх вирішення.

1. Що таке Smart TV

Smart TV [2] – це телевізор з вбудованим доступом до Інтернету. Крім традиційної для телевізорів чи телевізійних приставок функції трансляції телеканалів, Smart TV пристрої також можуть забезпечити інтернет-телебачення, веб-серфінг, трансляцію відео або аудіо контенту, завантаження та роботу із застосунками й інше. Фактично, Smart TV є комп'ютером у форм-факторі телевізора чи телевізійної приставки. І як і комп'ютери, Smart TV для функціонування потрібна операційна система, або як її ще називають – платформа.

2. Перелік основних платформ Smart TV

На сьогодні основними платформами Smart TV, їхніми перевагами та недоліками [3] є:

- Android TV
 - Переваги:
 - Чисте компонування.
 - Рядок з рекомендованим контентом.
 - Гарний універсальний пошук.
 - Велика кількість застосунків.
 - Недоліки:
 - Більшість збірок мають багато помилок та схильні до збоїв.
- WebOS
 - Переваги:
 - Гарна швидкість роботи.
 - Швидка навігація.
 - Інтеграція Google Assistant та Alexa.
 - Недоліки:

- Значні недоліки відсутні.

- Amazon Fire TV

- Переваги:

- Глибоке з'єднання з Amazon Video.

- Недоліки:

- Самі телевізори на яких встановлена ця операційна система доволі поганої якості.

- SmartCast

- Переваги:

- Вбудований Google Chromecast.

- Недоліки:

- Повільна швидкість роботи.

- Tizen

- Переваги:

- TV Plus пропонує безкоштовні канали.

- Гарна швидкість роботи та навігації.

- Недоліки:

- Мала кількість існуючих застосунків.

- Поганий універсальний пошук.

- MyHomeScreen

- Переваги:

- Гарна швидкість роботи.

- Недоліки:

- Трохи простий інтерфейс.

- Roku TV

- Переваги:

- Найкращий універсальний пошук.

- Дуже проста платформа у користуванні.
- Гарна швидкість роботи.
- Недоліки:
 - Трохи простий інтерфейс.

3. Огляд основних рис Smart TV та стан платформ на 2020 рік

Загалом, можна сказати про те, що не існує загально стандартної операційної системи для Smart TV, найближчою до цього є Android TV, як і її аналог на ринку смартфонів, але і для неї існують варіації залежно від бренду. Найкраще рішення на основі Android TV запропонувала Sony. Але існує багато інших виробників телевізорів, що використовують цю операційну систему, серед них: Philips, Sharp, Hisense. Також вона використовується у пристроях Nvidia Shield, які призначені для трансляції відеоігор з віддалених комп'ютерів Nvidia.

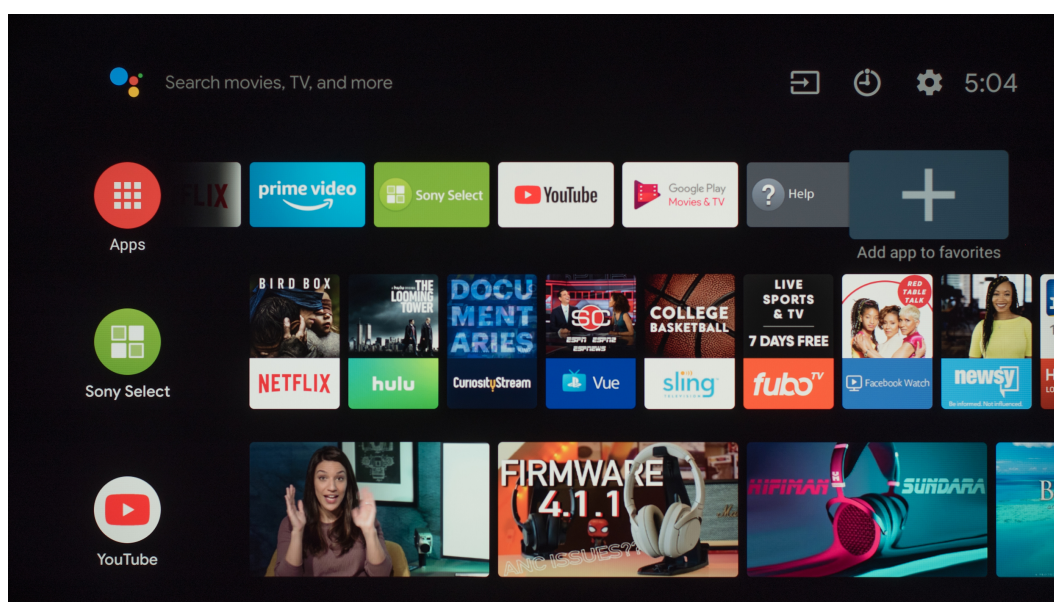


Рисунок 2.1 - інтерфейс Sony Android TV [4]

Більшість платформ схильні робити мінімалістичний інтерфейс, окрім Android TV, інтерфейс якої заповнено різноманітними шарами контенту (див.

рисунок 2.1). SmartCast, яка присутня на телевізійних пристроях компанії Vizio, спробувала взяти функції Android TV та помістити їх у більш мінімалістичний інтерфейс та логічніше компонування, з цією задачею вона справилась, але швидкість і стабільність її роботи доволі погана, через що вона користується не дуже великою популярністю, як і інша операційна система Amazon Fire TV. У цьому випадку, мала популярність викликана поганою якістю телевізорів компаній Toshiba та Insignia, хоча сама операційна система доволі гарна.

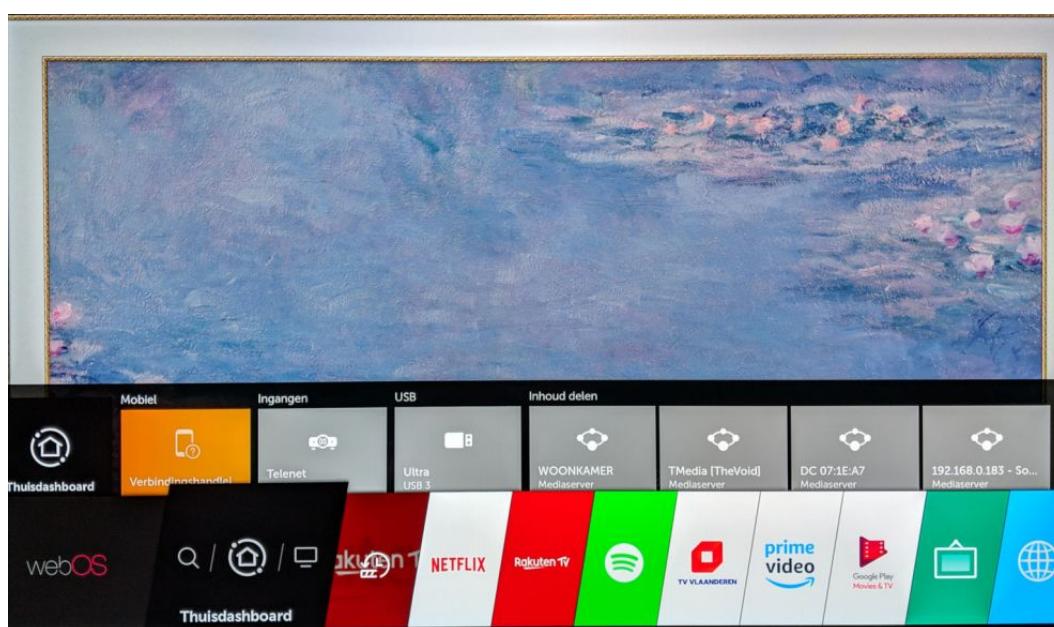


Рисунок 2.2 - Інтерфейс WebOS 4.5 [5]

Іконою мінімалістичного інтерфейсу серед платформ Smart TV є WebOS (див. Рисунок 2.2). Ця операційна система розроблена компанією LG. Її можна знайти лише на телевізорах вироблених цією компанією. Окрім гарно продуманого інтерфейсу, платформа може похизуватись можливістю програвати відео з YouTube у режимі 360-градусів та підтримку Google Assistant і Alexa. Іншим прикладом мінімалістичного інтерфейсу, можливо навіть дуже мінімалістичного, можна назвати MyHomeScreen, яку можна знайти на телевізорах марки Panasonic. Окрім простого інтерфейсу, ця

платформа може похизуватись дуже гарною можливістю налаштовувати цей інтерфейс, але він так само залишається доволі простим. Іншим гарним прикладом вдало створеного мінімалістичного інтерфейсу є операційна система Tizen, яка взяла багато рис від WebOS. Розробниками цієї платформи є компанія Samsung, і знайти цю платформу можна лише на їх телевізорах. Поки що ця платформа не є однією з найкращих, але це може змінитись, коли Samsung інтегрує у Tizen свій TV AI, який створить для Tizen перевагу й особливість на фоні інших конкурентів.

Найбільшою кількістю застосунків може похизуватись Android TV, що не дивно. Інші платформи мають меншу кількість додатків, але дефіциту немає ні в кого, крім Tizen, що стає навіть більшим недоліком враховуючи те, що стандартні застосунки недостатні для комфортної роботи хоча б трохи просунутих користувачів.

Іншим важливим критерієм при виборі Smart TV є якість універсального пошуку. Універсальний пошук дозволяє робити пошук з інтерфейсу операційної системи не тільки по ній самій, а й по встановлених застосунках, якщо вони це дозволяють. Найкраще його імплементовано у іншій операційній системі під назвою Roku TV, яка дозволяє робити пошук по більш ніж 30 застосунків, що є можливим завдяки досвіду розробників у створенні медіа транслятора Roku Streaming Stick. Roku TV можна знайти на невеликій кількості телевізорів виробників Haier, Hisense, Insignia, Sharp та TCL. Такий гарний універсальний пошук, наприклад, дозволяє проводити користувачу пошук певного фільму по Netflix, Google Play TV and Movies, Amazon і так далі та знайти найпривабливішу ціну на цей фільм.



Рисунок 2.3 - Приклад універсального пошуку на Roku TV [6]

4. Перелік основних проблем розробки на Smart TV та їх рішення

1. Велика кількість операційних систем

- *Опис:* Як і з усіма новими технологіями, стандартизація на одній операційній системі не відбувається одразу. За перше місце операційної системи для Smart TV досі борються Tizen, WebOS та Android TV. За словами Білла Коутінхо, співзасновника та старшого технічного директора компанії Dexense, сьогодні великою складністю для розробників є відмінності між платформами [7].
- *Рішення:* Можливим рішенням для цього є розробка веб-застосунку за допомогою якогось фреймворку, наприклад React, але це не є панацеєю, оскільки застосунок має гарно працювати на великій кількості різних моделей Smart TV. Розробники зазначають, що під час розробки їм часто доводилось модифікувати чи адаптувати версії застосунків залежно від пристрою. Наприклад, телевізори на операційній системі Tizen

мають власні бібліотеки для стандартних функцій, наприклад для програвача (AVPlay), у той час як більшість інших платформ використовують стандартну функціональність HTML відео програвання, що вимагає більшої складності оперування застосунку, щоб він розумів, що залежності для роботи на Samsung не повинні бути завантажені при програванні на інших платформах і так далі. Іншим прикладом різниці у платформах є різні коди кнопок на пульті, які відповідають за одні й ті ж команди, що теж ускладнює застосунок.

2. Велике обмеження ресурсів

- *Опис:* На відміну від сучасних комп'ютерів, Smart TV мають значно менше оперативної пам'яті та потужності центрального процесору. Так наприклад більшість персональних комп'ютерів мають об'єм оперативної пам'яті від 8 до 16 гігабайт, чотирьох'ядерні процесори з тактовою частотою в межах 3,3-3,69 ГГц [8], у той час як більшість сучасних Smart TV мають об'єм оперативної пам'яті від 2 до 4 гігабайт, теж чотирьох'ядерні процесори, але їх тактова частота коливається в межах від 1,3 ГГц до 2 ГГц [9]. Також потрібно розуміти, що у Smart TV використовуються мобільні процесори, оскільки в них немає активної системи охолодження, що теж впливає на продуктивність. Тому платформи вводять обмеження на те скільки ресурсів оперативної пам'яті може бути доступно застосункам, в середньому цей ліміт становить 250 мегабайт [10], а розробникам у свою чергу доводиться значно економити на ресурсах які потребує їх застосунок. Прикладів може бути багато - надто просунутий інтерфейс з великою кількістю анімацій та/або

функцій і контенту, що звичайно використовує багато оперативної пам'яті, що впливає у аварійне припинення роботи.

- *Рішення:* Рішення цієї проблеми теж доволі очевидне - зменшити кількість контенту, що відображається, мінімізувати кількість анімації. Також покращити продуктивність застосунку може його розробка за допомогою “рідних” для платформи методів - для Android TV, наприклад, за допомогою Android Studio та Java або Kotlin і так далі. Але це породжує першу проблему розробки, якщо застосунок, звичайно, не є ексклюзивним лише для цієї платформи. Розробка одного й того ж застосунку окремо для кожної платформи робить розробку значно дорожчою й повільнішою, тому до такого рішення прибигати лише якщо розробник планує випустити застосунок для однієї або максимум двох платформ. Але можна пришвидшити застосунок і за розробляючи веб-застосунок, наприклад, зробити анімацію вручну за допомогою canvas, написання проблемних ділянок коду за допомогою vanilla js і так далі, що уповільнює та ускладнює розробку, але не до такої степені як попереднє рішення. Також універсальним методом покращення продуктивності є кешування паралелізація та відкладання запитів або обробок, наприклад, під час програвання відео.

3. Мала кількість детальної документації

- *Опис:* Оскільки маркет Smart TV досі знаходиться на стадії народження, зрозуміло що детальна документація та гарні приклади оцінки продуктивності є доволі рідкісними.
- *Рішення:* На жаль, для цієї проблеми немає універсального рішення. Великі компанії мають самі розробляти свої методології

розробки. Іншим залишається чекати до тих пір, доки домінуюча операційна система не з'явиться на ринку, як сталося з індустрією смартфонів.

4. Складна навігація

- *Опис:* На відміну від навігації по графічному інтерфейсу комп'ютера за допомогою миші й клавіатури, навігація по графічному інтерфейсу Smart TV відбувається за допомогою пульта керування, який має значно більше обмежень, що у свою чергу робить навігацію більш складною. Це стає проблемою, коли застосунок має багато опцій або вимагає багато кроків для конфігурації. Також особливістю навігації є фокусування на певному елементі меню, оскільки втрата фокусу є втратою можливості навігації по меню застосунку. Також часто при переході від дочірнього меню до батьківського, потрібно пам'ятати на якому елементі був фокус до цього, через що процес навігації теж стає складнішим.
- *Рішення:* Рішенням цієї проблеми слугує оптимізація спеціалістом по UX інтерфейсу користувача, а саме: викладення найкращих шляхів по меню застосунку для користувача, мінімізування кількості кліків, а тим самим переходів, та надання цьому всьому відчуття логічності. З фокусуванням допомагає надання кожному елементу реакції при фокусуванні на цьому елементі, аби користувач знав на якому елементі в даний момент відбувається фокусування. Для того щоб не втрачати фокусування та зберегти його при поверненні варто надати кожному елементу на який можна сфокусуватись унікальний id.

5. Різна швидкість роботи інтернету

- *Опис:* Цей пункт якоюсь мірою стосується другого та й взагалі не зовсім залежить від самої платформи, а скоріш від користувача і вона стосується застосунків які залежать від підключення до інтернету, але оскільки з цією проблемою часто стикаються розробники і практично всі застосунки на Smart TV потребують підключення до інтернету, було вирішено виділити їй окремий пункт. Вона полягає в тому, що компанія розробника не може керувати швидкістю підключення до інтернету користувача, оскільки цим керує інтернет провайдер і це залежить від типу підключення користувача. Одні користувачі використовують дротове з'єднання для Smart TV, інші бездротове, одні користувачі мають безлімітне, швидкісне і надійне оптоволоконне підключення, інші мають нестабільне і лімітоване підключення, тому розробник має передбачити ці можливі варіації і зробити так, щоб застосунок реагував на це.
- *Рішення:* Для вирішення цієї проблеми потрібно й надалі ускладнювати застосунок. Гарним універсальним рішенням є буферизація даних, виконання більшої кількості обробки даних на стороні сервера, швидкість цієї обробки, та загальна якість backend-у. Якщо застосунок вимагає завантаження великих об'ємів даних, можна використовувати так звані Web Workers, які дозволяють запускати скриптові операції у фоновому потоці окремо від основного діючого потоку, автоматичне налаштування часу буферизації залежно від пропускної здатності користувача і так далі.

5. Розробка власного застосунку та висвітлення проблем з якими довелося зіткнутись

5.1. Опис застосунку та обґрунтування теми й обраних технологій

Було вирішено розробити застосунок, який слугує програвачем інтернет-радіостанцій для Android TV. Вибір обґрунтовано наявністю програвання потоку контенту, в даному випадку аудіо контенту, що дозволить втілити наведені раніше рішення проблем програвання, така як буферизація. Також потрібно буде зробити зручний інтерфейс користувача та вирішити проблему фокусування та його відстеження. Android TV як цільова платформа була обрана як один із запропонованих способів вирішення проблеми обмежених ресурсів - розробка “нативного” застосунку.

5.2. Опис засобів розробки

Для розробки було використано середовище розробки під назвою Android Studio, яка найкраще призначена для розробки застосунків спеціально під Android TV. Мовою програмування було обрано Java стандарту версії 8, через наявність гарної документації та стабільності роботи. Цільовою версією Android було обрано версію 9.0 під кодовою назвою Pie, через її відносну новітність та поширеність. Для тестування застосунку було використано Android Virtual Device Manager, який дозволяє емулювати певний Android пристрій. Організації роботи сприяв контроль версій Git.

5.3. Проектування графічного інтерфейсу

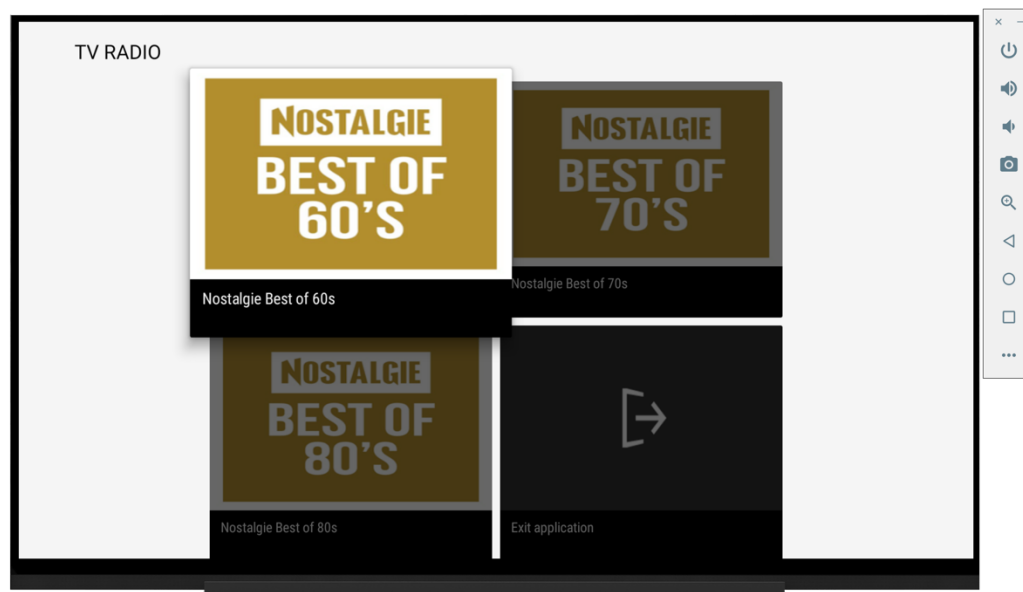


Рисунок 5.3.1 – вигляд головного меню застосунку

Було вирішено зробити інтерфейс у вигляді сітки з двома колонками (див. рисунок 5.3.1), новий елемент сітки, який не вміщується у першу стрічку, переноситься на наступну на місце першої колонки і так далі, таким чином збільшення сітки відбувається вертикально. Останнім елементом є опція виходу з програми. Оскільки користувач може захотіти швидко переміститись до цієї опції, потрібно бути передбачено швидкий перехід по сітці, так якщо користувач знаходиться на першому елементі сітки й натискає на пульті керування кнопку “Нагору”, має бути переміщено фокус на перший елемент останнього рядка і так для всіх крайніх елементів. При натисненні на сфокусований елемент може бути дві реакції - якщо елемент є переходом до радіостанції, то має бути відкрито аудіо програвач, який буде програвати відповідний аудіо потік, або елемент є опцією виходу із застосунку відбудеться завершення його роботи. Також у головному меню повинно відображатись яка радіостанція зараз грає, якщо вона грає. Кольорову схему

було обрано мінімалістичну - чорно-білу. Під час підготовки до початку програвання - користувач має бути сповіщений, що відбувається буферизація та анімоване колесо завантаження, що свідчить про те, що застосунок працює. Заголовок застосунку було обрано розмістити у лівому верхньому куті.

5.4. Збереження та зчитування даних та їх представлення

```
{
  "cards": [
    {
      "type": "RADIO_ITEM",
      "title": "Nostalgie Best of 60s",
      "localImageResource": "https://guzei.com/online_radio/logo300/5365.png",
      "stream": "http://cdn.nrjaudio.fm/adwz1/fr/30615/mp3_128.mp3",
      "position": 0
    },
    {
      "type": "RADIO_ITEM",
      "title": "Nostalgie Best of 70s",
      "localImageResource": "https://guzei.com/online_radio/logo300/5366.png",
      "stream": "http://cdn.nrjaudio.fm/adwz1/fr/30613/mp3_128.mp3",
      "position": 1
    },
    {
      "type": "RADIO_ITEM",
      "title": "Nostalgie Best of 80s",
      "localImageResource": "https://guzei.com/online_radio/logo300/5359.png",
      "stream": "http://cdn.nrjaudio.fm/adwz1/fr/30605/mp3_128.mp3",
      "position": 2
    },
    {
      "type": "EXIT_ITEM",
      "title": "Exit application",
      "localImageResource": "exit",
      "position": 3
    }
  ]
}
```

Рисунок 5.4.1 – приклад файлу JSON

Дані зберігаються у вигляді JSON файлу в папці ресурсів (див. рисунок 5.4.1). Зчитування даних з цього файлу розділено на два кроки:

1. Зчитування вмісту файлу у вигляді стрічки.
2. Трансформація JSON-стрічки у Java-об'єкт за допомогою утиліти Gson.

```

import com.google.gson.annotations.SerializedName;

public class Card {

    @SerializedName("title")
    private String mTitle = null;
    @SerializedName("localImageResource")
    private String mLocalImageResource = null;
    @SerializedName("stream")
    private String mStream = null;
    @SerializedName("type")
    private Card.Type mType = null;
    @SerializedName("position")
    private int mPosition;

    public String getTitle() { return mTitle; }

    public Type getType() { return mType; }

    public int getPosition() { return mPosition; }

    public String getStream() { return mStream; }

    public String getLocalImageResourceName() { return mLocalImageResource; }

    public enum Type {
        RADIO_ITEM,
        EXIT_ITEM
    }
}

```

Рисунок 5.4.2 – структура класу Card

У програмі дані представляються класами Cards та Card (див. рисунок 5.4.2). Cards містить у собі всі картки інтернет-радіостанцій, Card представляє собою одну картку інтернет-радіостанції й містить у собі всю потрібну інформацію про неї, таку як назва, посилання на потік даних, посилання на зображення обкладинки станції, тип картки та інше.

5.5. Архітектура застосунку

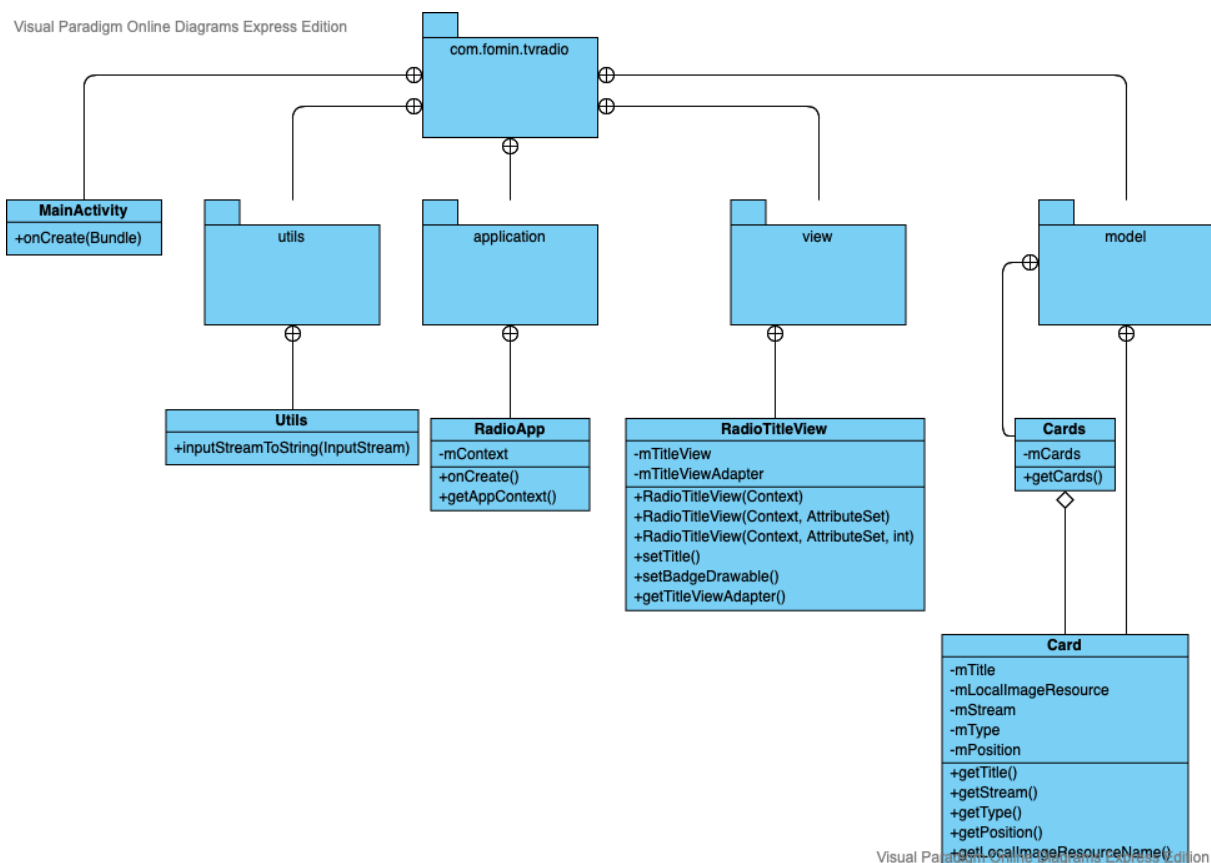


Рисунок 5.5.1 – діаграма класів програми частина 1

Застосунок складається з 12 класів (див. рисунок 5.5.1, рисунок 5.5.2). Клас RadioApp фіксує Context всього застосунку, аби можна було доступитись до нього з будь-якого класу. MainActivity відповідає за створення головного меню. RadioTitle виконує функцію стилю відображення заголовку застосунку.

```

private void createRows() {
    String json = Utils.inputStreamToString(getResources()
        .openRawResource(R.raw.grid_example));
    mCards = new Gson().fromJson(json, Cards.class);
    List<Card> row = mCards.getCards();
    nRows = row.size()/COLUMNS;
    mAdapter.addAll( index: 0, row);
}

```

Рисунок 5.5.2 – зчитування JSON файлу

RadioFragment встановлює текст заголовку та слугує представленням даних у графічному вигляді, у ньому відбувається зчитування даних з JSON файлу (див. *рисунок 5.5.2*), створення Singleton об'єкту класу Radio та налаштування інтерфейсу за допомогою екземплярів допоміжних класів, таких як: VerticalGridPresenter, CardPresenterSelector, ArrayObjectAdapter. Radio відповідає за саме програвання аудіо потоку, оскільки програвання має бути одне і доступне на будь-якому етапі роботи застосунку, було прийнято рішення скористатись патерном Singleton. VerticalGridPresenter слугує конкретизацією якою саме сіткою представляти контент, перед його встановленням налаштовується степінь приближення виділеної картки та число колонок у сітці. ArrayObjectAdapter виконує функцію доступу до моделі даних за допомогою ArrayList, а за презентацію відповідає CardPresenterSelector. Функція CardPresenterSelector у тому, що він обирає який саме Presenter обирати в залежності від типу даних у ArrayObjectAdapter. У даному випадку CardPresenterSelector працює лише з даними типу Card і представляє їх лише за допомогою одного Presenter-а – ImageCardViewPresenter.

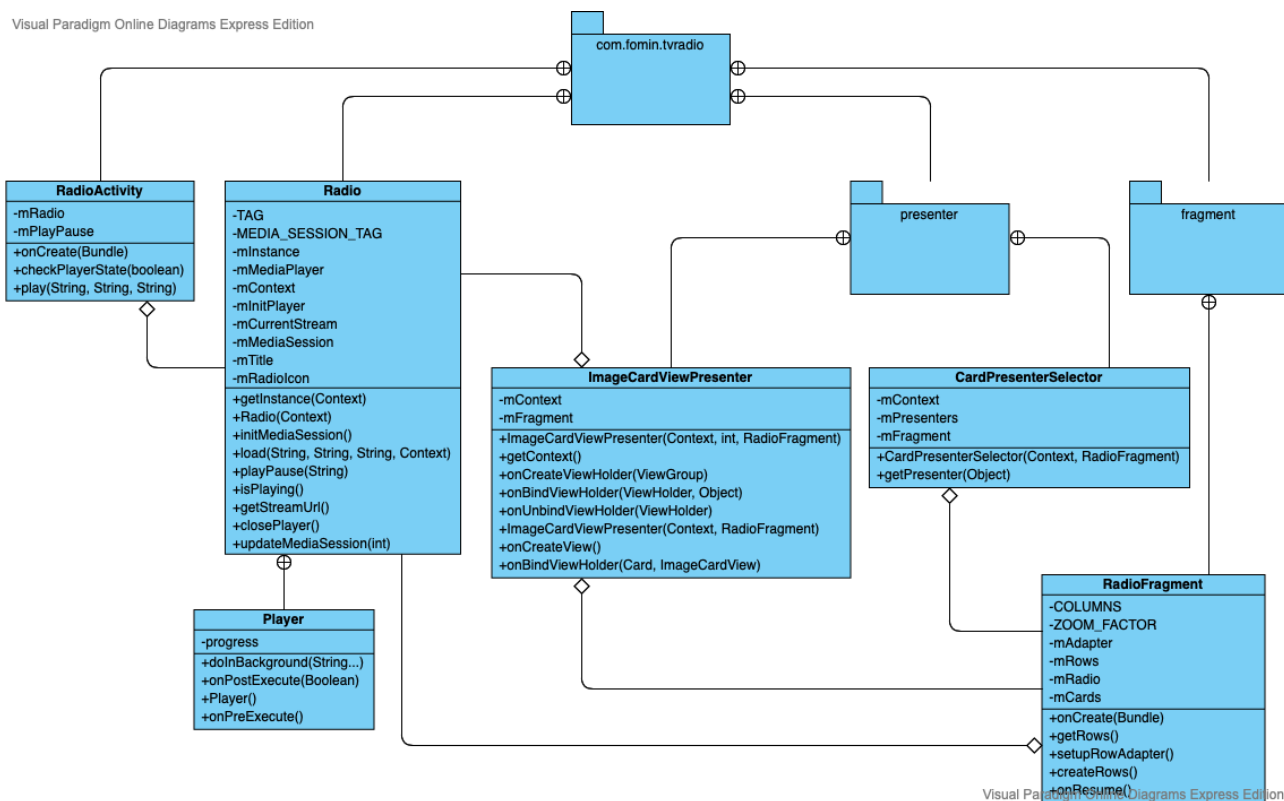


Рисунок 5.5.3 – діаграма класів програми частина 2

ImageCardViewPresenter відповідає за з'єднання моделі даних Card з його представленням у вигляді ImageCardView. ImageCardViewPresenter встановлює колір картки, зображення, заголовок, текст, поведінку при кліку по картці або при натисканні певної клавіші. Саме в цьому класі було перероблено навігацію по сітці, аби вона була більш зручною для користувача.

```

        boolean load(String streamUrl, String radioIcon, String title, Context context) {
            mContext = context;
            if (!mCurrentStream.equals(streamUrl)) {
                mMediaPlayer.reset();
                mCurrentStream = streamUrl;
                mRadioIcon = radioIcon;
                mTitle = title;
                new Player().execute(streamUrl);
                return true;
            }
            return false;
        }

@Override
protected Boolean doInBackground(String... params) {
    boolean prepared;
    try {
        Radio radio = Radio.mInstance;
        radio.mMediaPlayer.setDataSource(params[0]);

        radio.mMediaPlayer.setOnCompletionListener(mp -> {
            radio.mInitPlayer = false;
            radio.mMediaPlayer.stop();
            radio.mMediaPlayer.reset();
        });
        radio.mMediaPlayer.prepare();
        prepared = true;
    } catch (IllegalArgumentException | SecurityException |
            IllegalStateException | IOException e) {
        prepared = false;
        e.printStackTrace();
    }
    return prepared;
}

@Override
protected void onPostExecute(Boolean result) {
    super.onPostExecute(result);
    if (progress.isShowing()) {
        progress.cancel();
    }
    Radio radio = Radio.mInstance;
    radio.mMediaPlayer.start();

    if (radio.mMediaSession == null) {
        radio.initMediaSession();
    }
    AtomicBoolean finished = new AtomicBoolean( initialValue: false);

    new Thread() -> {
        radio.updateMediaSession(PlaybackState.STATE_PLAYING);
        finished.set(true);
    }.start();
    while(!finished.get()){
        try {
            Log.d(TAG, msg: "Waiting for the media session to be updated");
            Thread.sleep( mills: 100);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
    }
    radio.mInitPlayer = true;
}

```

Рисунки 5.5.4, 5.5.5, 5.5.6 – під'єднання до потоку та буферизація

За відображення вікна програвання радіостанції відповідає клас `RadioActivity`, тут зчитуються дані про радіостанцію (посилання на аудіо потік, зображення, назву) відповідні картці радіостанції на яку натиснув користувач на попередньому екрані. Після цього відбувається підготовка програвання аудіо потоку цієї станції, а саме під'єднання до потоку, буферизація і початок програвання потоку (див. рисунки 5.5.4, 5.5.5, 5.5.6).

5.6. Робота застосунку

Після запуску застосунку користувачу надано перелік усіх радіостанцій та картку виходу (див. *рисунок 5.3.1*). Натиснувши на картку радіостанції - застосунок переміщує користувача на вікно програвання радіостанції, натиснувши на картку виходу - застосунок закривається й користувач переходить до головного екрану системи.



Рисунок 5.6.1 – вікно програвання аудіо

Під час створення вікна програвання, застосунок зчитує дані про радіостанцію. Знаходячись у вікні програвання, користувач може зупинити або знову продовжити програвання аудіо потоку (див. *рисунок 5.6.1*).

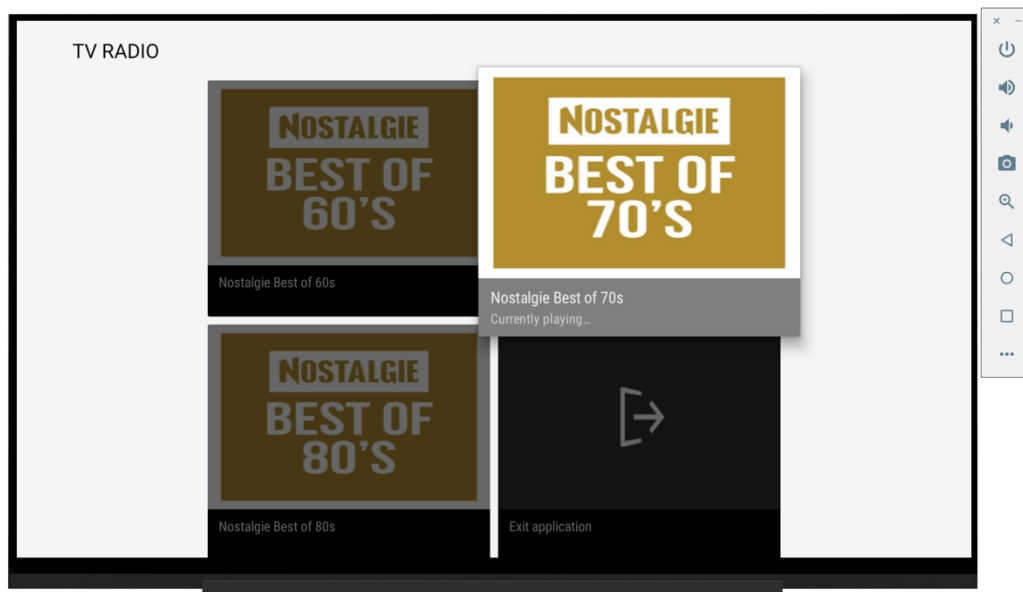


Рисунок 5.6.2 – вигляд головного меню при програванні

Якщо користувач повернувся до головного меню програми і не зупинив програвання аудіо потоку, серед переліку радіостанцій буде відмічена та яка програватиметься зараз за допомогою підпису “Currently Playing...” та сірим кольором картки (див. рисунок 5.6.2).

5.7. Проблеми які виникли при розробці

Під час розробки застосунку довелося зіткнутись з деякими проблемами які були перелічені у розділі 4.

```
cardView.setOnKeyListener((v, keyCode, event) -> {
    if (event.getAction() != KeyEvent.ACTION_DOWN)
        return true;
    if (keyCode == KEYCODE_DPAD_LEFT) {
        mFragment.setSelectedPosition(RadioFragment.COLUMNS-1);
        return true;
    } else if (keyCode == KEYCODE_DPAD_UP) {
        mFragment.setSelectedPosition(RadioFragment.COLUMNS*(mFragment.getRows()-1));
        return true;
    }
    return false;
});
```

Рисунок 5.7.1 – приклад коду з яким виникли проблеми

Першою та найбільшою проблемою була недостатня кількість документації та її неякісність у деяких місцях. Наприклад, під час зміни стандартної навігації по сітці з картками радіостанцій довелось витратити багато часу через те що не до кінця було зрозуміло як працює функція налаштування спостерігача при натисненні певної клавіші. Коли автор курсової роботи зрозумів як працює функція, а саме про що свідчить результат який вона повертає, було вирішено знову подивитись на опис цієї функції у документації, оскільки перед цим автор не бачив, що там описується про що свідчить результат який повертає функція (див. *рисунок 5.7.1*). Але після детальної перевірки це пояснення було знайдено, щоправда воно дуже непомітне, що свідчить про невелику підтримку документації розробниками Android TV, на відміну від документації створеної для операційної системи для смартфонів від тієї ж компанії розробника – Android OS. Інша проблема виникла при спробі налаштування стилю для заголовка застосунку, який відображається у головному меню застосунку. Пояснень на сайті з документацією не було зовсім, а пояснення інших людей на форумах були або застарілими, або незрозумілими. Довелось збирати інформацію з різних джерел і намагатись скомпонувати її.

Другою проблемою була незручна навігація для телевізорів. Це стосується усіх платформ Smart TV. Потрібно надати повну функціональність застосунку при цьому обмежившись лише декількома кнопками пульта керування телевізором.

Третьою та останньою проблемою була оптимізація коду застосунку для його найбільшої продуктивності для комфортної роботи застосунку і впевненості, що воно не буде завершено аварійно через перевищення ліміту зайнятої оперативної пам'яті. Ця проблема часто змушувала переписувати код багато разів та робила процес розробки більш повільним.

Висновки

Існує ще багато інших дрібних особливостей розробки на Smart TV, на жаль, їх всі неможливо перелічити та розібрати. Проте, спираючись на поставленні завдання вдалося розкрити основні проблеми та особливості розробки на Smart TV з якими доводиться стикатися розробникам, найкращі способи їх подолання та які саме платформи Smart TV користуються попитом і чому.

Спочатку були розкриті базові речі, що включають опис загальних понять про Smart TV. Після цього було проаналізовано ринок Smart TV і згідно цьому зіставлено перелік найпопулярніших платформ. Далі автором курсової роботи було проаналізовано досвід інших розробників на платформи для Smart TV, а саме з якими проблемами їм довелося зіткнутись і як вони їх подолали. Для завершення, автор курсової роботи розробив застосунок для однієї з платформ, аби отримати власний досвід й перевести його у текстовий вигляд для того щоб переконатись у словах інших розробників.

Треба розуміти, що консолідація на певній операційній системі не відбувається одразу. Спочатку ринок є дуже роздробленим. Згадаємо початок ери комп'ютерів та смартфонів, було дуже багато конкурентів та рішень, розробники мали створювати застосунки для всіх популярних платформ. Проте з часом ми отримали лідерів. Так на ринку комп'ютерів лідером є Windows, після якого йдуть MacOS та різні збірки Linux. Більш пізнішим прикладом є ринок смартфонів, де було безліч операційних систем, проте наразі їх лише дві: Android та iOS. Так само буде зі Smart TV, час сформує основні платформи.

Будь-який гарний розробник скаже про важливість зручного графічного інтерфейсу для користувача. Коли ми говоримо про Smart TV – це стає як ніколи актуальним, оскільки навігація по Smart TV доволі складна і

тому при розробці застосунку на будь-яку із платформ Smart TV потрібно пам'ятати про зручність використання інтерфейсу цього застосунку.

Окрім інтерфейсу, потрібно приділяти багато уваги оптимізації роботи застосунку, а саме швидкості його роботи. На жаль, при розробці застосунків для Smart TV не можна відноситись до ресурсів так не уважно як при розробці застосунків на комп'ютери. Smart TV мають значно меншу потужність і можливості, а тому й більші обмеження. Отже це теж є важливим аспектом при розробці застосунку.

Наостанок варто наголосити про важливість навичок відладки програми та розуміння як що та де працює. Коли мова йде про розробку на Smart TV, ці навички стають навіть більш важливими. Це пов'язано з малим обсягом зрозумілої документації для розробників. І це зрозуміло, бо знову-таки, ринок є молодим і не дивно що підтримка платформ не є дуже гарною.

Список літератури

1. Number of smart TV users in the United States from 2016 to 2022 (in millions) Forecast [Електронний ресурс] – Режим доступу до ресурсу: <https://www.statista.com/statistics/718737/number-of-smart-tv-users-in-the-us/>
2. smart TV – Computer Definition [Електронний ресурс] – Режим доступу до ресурсу: <https://www.yourdictionary.com/smart-tv>
3. Best smart TV 2020: every smart TV platform and which set does it best [Електронний ресурс] – Режим доступу до ресурсу: <https://www.techradar.com/news/television/6-best-smart-tv-platforms-in-the-world-today-1120795#section-smart-tv-platforms>
4. The 3 Best Sony TVs of 2020 [Електронний ресурс] – Режим доступу до ресурсу: <https://www.rtings.com/tv/reviews/sony>
5. EVERYTHING YOU NEED TO KNOW ABOUT WEBOS 4.5: TIPS AND RECOMMENDATIONS [Електронний ресурс] – Режим доступу до ресурсу: <https://allhomecinema.com/everything-you-need-to-know-about-webos-4-5-tips-and-recommendations/>
6. The Best Media Streaming Devices [Електронний ресурс] – Режим доступу до ресурсу: <https://thewirecutter.com/reviews/best-media-streamers/>
7. 4 Challenges in Developing Smart TV Applications – Режим доступу до ресурсу: <https://dexence.com/4-challenges-developing-smart-tv-applications/>
8. Данные об оборудовании и ПО пользователей: March 2020 [Електронний ресурс] – Режим доступу до ресурсу: <https://store.steampowered.com/hwsurvey/Steam-Hardware-Software-Survey-Welcome-to-Steam?platform=pc>

9. NVIDIA Shield Android TV (2019) specs [Электронный ресурс] – Режим доступа до ресурсу: <https://www.androidcentral.com/nvidia-shield-android-tv-specs>
10. Проблемы и нюансы при разработке под SmartTV с использованием React.js [Электронный ресурс] – Режим доступа до ресурсу: <https://habr.com/ru/post/467961/>