Android RSS Reader with Text-to-Speech

Objective:

Develop an Android app that:

1. Fetches RSS news from a newspaper feed.
2. Displays headlines in a scrollable list (RecyclerView).
3. Reads the selected headline aloud using **Text-to-Speech**.
4. Includes a play/pause button for each headline item.

- Start a new project with empty views activity
- Add Internet permission to AndroidManifest
- Use ChatGPT for help (but no code copy-paste from others)

📁 Project Structure

```
RSSReaderApp/
├── model/
│   └── NewsItem.kt
├── network/
│   └── RssParser.kt
├── ui/
│   └── NewsAdapter.kt
├── MainActivity.kt
└── res/
    ├── layout/
    │   ├── activity_main.xml
    │   └── news_item.xml
```

Remember model, network, ui are created as packages

📌 Functional Requirements

✅ 1. RSS Feed Parsing

- Go to https://github.com/bakinazik/rss and choose an rss link for a newspaper, or, from https://rss.com/blog/popular-rss-feeds/#the-most-popular-podcast-rss-feeds
- Fetch and parse the XML to extract:
    - Title
    - Description
- Use a **background thread** to avoid blocking the UI.

✅ 2. RecyclerView for Headlines

- Display each news title as a card.
- Include a **play/pause** button.
- Use a custom RecyclerView.Adapter.

✅ 3. Text-to-Speech (TTS)

- On tapping the play button, read the news title and description aloud.
- Only one item should play at a time.
- Handle repeated taps or double speak issues gracefully (use stop() before speak()).
- Pause button should stop the playback.

🧩 Example Guidance & Prompts (for ChatGPT)

Students should be instructed to **ask ChatGPT smart questions**. Example prompts:

- "How do I parse an RSS feed from a URL using Kotlin?"
- "How do I run a network operation in the background in Android?"
- "How do I set up a RecyclerView with a custom adapter in Android?"
- "How do I use TextToSpeech in Kotlin to read a string?"
- "How do I prevent TTS from speaking twice?"

🔍 Tasks & File Responsibilities

📄 NewsItem.kt

- Define a simple data class with title and description.

📄 RssParser.kt

- Create a method to fetch and parse the XML feed.
- Use XmlPullParser or other options like Jsoup.

📄 NewsAdapter.kt

- Custom RecyclerView adapter.
- Each item should show the title and a play/pause button.
- Handle logic to read the item aloud and prevent duplicates.

MainActivity.kt

- Initialize TextToSpeech.
- Use lifecycleScope.launch to load RSS data in the background.
- Set up RecyclerView.
- Pass TTS instance to adapter.
- Handle proper shutdown of TTS in onDestroy.

activity_main.xml

- Just a full-screen RecyclerView, for example

```
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <androidx.recyclerview.widget.RecyclerView
        android:id="@+id/recyclerView"
        android:layout_width="0dp"
        android:layout_height="0dp"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintEnd_toEndOf="parent" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

news_item.xml

- Use a horizontal layout with:
    - TextView for the title
    - ImageButton for play/pause
- Style it cleanly with card or padding.

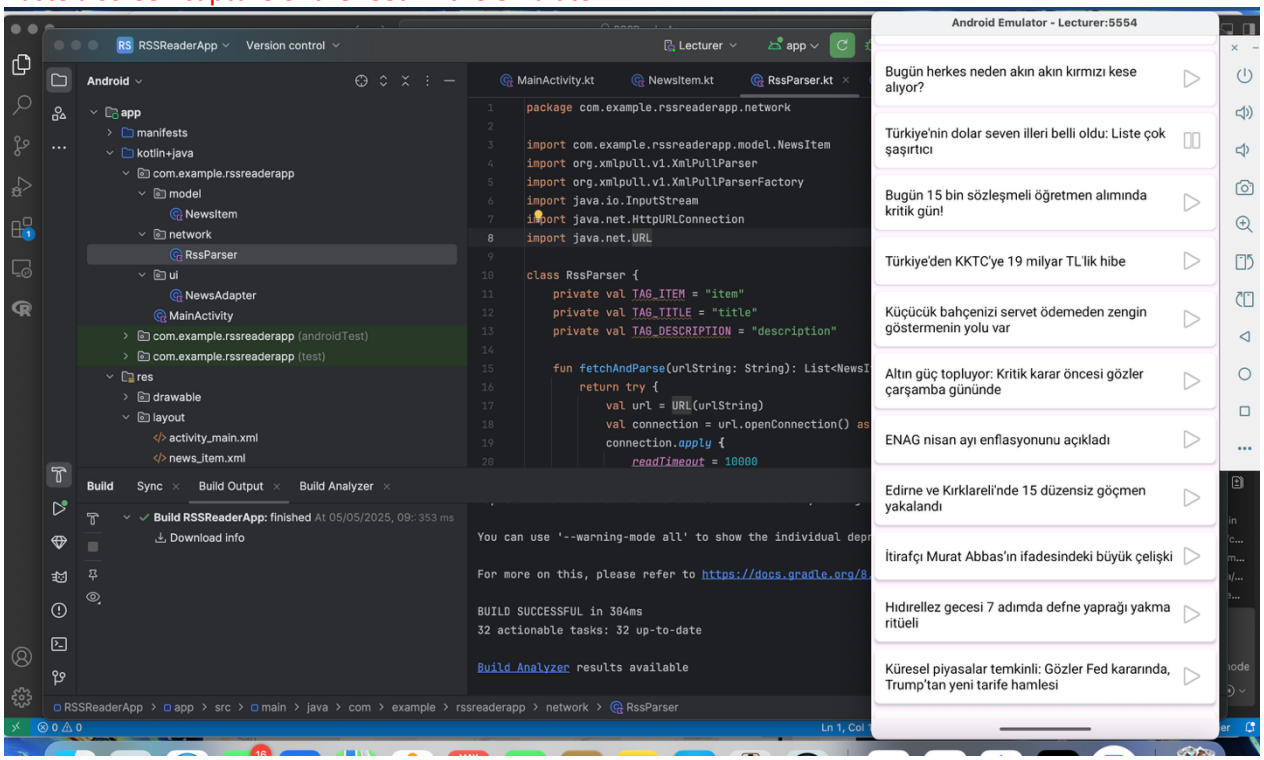
```
<androidx.cardview.widget.CardView
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:card_view="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_margin="4dp"
    card_view:cardCornerRadius="6dp"
    card_view:cardElevation="2dp">

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal"
        android:padding="12dp"
        android:gravity="center_vertical">

        <TextView
            android:id="@+id/titleTextView"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:textColor="#000000"
            android:textSize="16sp"
            android:maxLines="2"
            android:ellipsize="end" />

        <ImageButton
            android:id="@+id/playPauseButton"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:background="@android:color/transparent"
            android:src="@android:drawable/ic_media_play"
            android:contentDescription="Play/Pause" />
    </LinearLayout>
</androidx.cardview.widget.CardView>
```

Paste a screen capture of the feed in the emulator



Does it speak the news, how many times it reads a news for each click

It reads the news on each time user clicks the play button.

Paste RssParse.kt here

```
package com.example.rssreaderapp.network

import com.example.rssreaderapp.model.NewsItem
import org.xmlpull.v1.XmlPullParser
import org.xmlpull.v1.XmlPullParserFactory
import java.io.InputStream
import java.net.HttpURLConnection
import java.net.URL

class RssParser {
    private val TAG_ITEM = "item"
    private val TAG_TITLE = "title"
    private val TAG_DESCRIPTION = "description"

    fun fetchAndParse(urlString: String): List<NewsItem> {
        return try {
            val url = URL(urlString)
            val connection = url.openConnection() as HttpURLConnection
            connection.apply {
                readTimeout = 10000
                connectTimeout = 15000
                requestMethod = "GET"
                doInput = true
            }
```

```
            connection.connect()
            val inputStream = connection.inputStream
            val items = parseXml(inputStream)
            inputStream.close()
            connection.disconnect()
            items
        } catch (e: Exception) {
            e.printStackTrace()
            emptyList()
        }
    }

    private fun parseXml(inputStream: InputStream): List<NewsItem> {
        val items = mutableListOf<NewsItem>()
        var title: String? = null
        var description: String? = null
        var insideItem = false

        try {
            val factory = XmlPullParserFactory.newInstance()
            factory.isNamespaceAware = false
            val parser = factory.newPullParser()
            parser.setInput(inputStream, null)

            var eventType = parser.eventType
            while (eventType != XmlPullParser.END_DOCUMENT) {
                when (eventType) {
                    XmlPullParser.START_TAG -> {
                        when (parser.name) {
                            TAG_ITEM -> insideItem = true
                            TAG_TITLE -> if (insideItem) title =
readText(parser)
                            TAG_DESCRIPTION -> if (insideItem) description =
readText(parser)
                        }
                    }
                    XmlPullParser.END_TAG -> {
                        if (parser.name == TAG_ITEM) {
                            title?.let { t ->
                                description?.let { d ->
                                    items.add(NewsItem(t, d))
                                }
                            }
                            title = null
                            description = null
                            insideItem = false
                        }
                    }
                }
                eventType = parser.next()
            }
        } catch (e: Exception) {
            e.printStackTrace()
        }

        return items
```

```
    }

    private fun readText(parser: XmlPullParser): String {
        var result = ""
        if (parser.next() == XmlPullParser.TEXT) {
            result = parser.text
            parser.nextTag()
        }
        return result
    }
}
```

Paste MainActivity.kt here

```
package com.example.rssreaderapp

import android.os.Bundle
import android.speech.tts.TextToSpeech
import android.view.View
import android.widget.ProgressBar
import android.widget.TextView
import androidx.activity.enableEdgeToEdge
import androidx.appcompat.app.AppCompatActivity
import androidx.core.view.ViewCompat
import androidx.core.view.WindowInsetsCompat
import androidx.lifecycle.lifecycleScope
import androidx.recyclerview.widget.LinearLayoutManager
import androidx.recyclerview.widget.RecyclerView
import com.example.rssreaderapp.model.NewsItem
import com.example.rssreaderapp.network.RssParser
import com.example.rssreaderapp.ui.NewsAdapter
import kotlinx.coroutines.Dispatchers
import kotlinx.coroutines.launch
import kotlinx.coroutines.withContext
import java.util.*

class MainActivity : AppCompatActivity() {
    private lateinit var tts: TextToSpeech
    private var adapter: NewsAdapter? = null
    private val rssUrl = "https://www.sozcu.com.tr/feeds-rss-category-sozcu"
    private lateinit var recyclerView: RecyclerView
    private lateinit var progressBar: ProgressBar
    private lateinit var errorText: TextView

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        enableEdgeToEdge()
        setContentView(R.layout.activity_main)
        ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main)) {
v, insets ->
            val systemBars =
insets.getInsets(WindowInsetsCompat.Type.systemBars())
            v.setPadding(systemBars.left, systemBars.top, systemBars.right,
systemBars.bottom)
            insets
        }
```

```
        setupViews()
        initializeTTS()
    }

    private fun setupViews() {
        recyclerView = findViewById(R.id.recyclerView)
        progressBar = findViewById(R.id.progressBar)
        errorText = findViewById(R.id.errorText)

        recyclerView.layoutManager = LinearLayoutManager(this)
    }

    private fun initializeTTS() {
        tts = TextToSpeech(this) { status ->
            if (status == TextToSpeech.SUCCESS) {
                tts.language = Locale.getDefault()
                loadRss()
            } else {
                showError("Text-to-Speech initialization failed")
            }
        }
    }

    private fun loadRss() {
        showLoading()
        lifecycleScope.launch {
            try {
                val items = withContext(Dispatchers.IO) {
                    RssParser().fetchAndParse(rssUrl)
                }
                if (items.isEmpty()) {
                    showError("No news items found")
                } else {
                    showContent(items)
                }
            } catch (e: Exception) {
                showError("Error loading news: ${e.localizedMessage}")
            }
        }
    }

    private fun showLoading() {
        progressBar.visibility = View.VISIBLE
        recyclerView.visibility = View.GONE
        errorText.visibility = View.GONE
    }

    private fun showError(message: String) {
        progressBar.visibility = View.GONE
        recyclerView.visibility = View.GONE
        errorText.visibility = View.VISIBLE
        errorText.text = message
    }

    private fun showContent(items: List<NewsItem>) {
        progressBar.visibility = View.GONE
```

```
        errorText.visibility = View.GONE
        recyclerView.visibility = View.VISIBLE

        adapter = NewsAdapter(items, tts)
        recyclerView.adapter = adapter
    }

    override fun onDestroy() {
        super.onDestroy()
        adapter?.stopPlayback()
        tts.stop()
        tts.shutdown()
    }
}
```