Міністерство освіти і науки України
Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра інформатики та програмної інженерії

Звіт
до лабораторної роботи № 4 з дисципліни
«Розробка мобільних застосувань під Android»

Виконав ІП-24 Цюх В.М.

Перевірив Орленко С.П.

Київ 2025

Лабораторна робота №4

Тема: ДОСЛІДЖЕННЯ СПОСОБІВ РОБОТИ З МЕДІАДАНИМИ

ЗАВДАННЯ

БАЗОВЕ (12/20 балів). Написати програму під платформу Андроїд, яка має інтерфейс для запуску аудіо-файлів та відео-файлів. Мінімально інтерфейс має надавати можливість Програвати/Зупиняти/Призупиняти відтворення відео-файлу або аудіо-файлу, який зберігається у внутрішньому сховищі.

ПОВНЕ (20/20). Функціональність базового додатку додатково розширюється наступними можливостями: - надати вибір типу файлу для відтворення (аудіо або відео) з будь-якого сховища на мобільному пристрої; - надати вибір завантаження файлу з Інтернету; - використовувати для реалізації обробки медіа-даних спеціалізовані інструменти (особливу увагу приділити програванню відео).

Лістинг програмного коду

Файл App.tsx

```tsx
// Головний компонент застосунку, що дозволяє вибирати файл, вводити URL,
// завантажувати медіа та відтворювати його.
import React, { useState } from "react";
import { View, Text, StyleSheet, ScrollView } from "react-native";
import FilePicker from "./components/FilePicker";
import UrlInput from "./components/UrlInput";
import DownloadManager from "./components/DownloadManager";
import MediaPlayer from "./components/MediaPlayer";

const App: React.FC = () => {
  const [fileUri, setFileUri] = useState<string | null>(null);
  const [downloadUrl, setDownloadUrl] = useState<string>("");

  const handleFilePicked = (uri: string) => {
    setFileUri(uri);
  };

  const handleUrlSubmit = (url: string) => {
    setFileUri(url);
  };

  const handleDownloadUrlSubmit = (url: string) => {
    setDownloadUrl(url);
  };

  const handleDownloadComplete = (uri: string) => {
    setFileUri(uri);
```

```
    setDownloadUrl("");
  };

  const handleStop = () => {
    setFileUri(null);
  };

  return (
    <ScrollView contentContainerStyle={styles.container}>
      <Text style={styles.title}>Media Player App</Text>
      <View style={styles.section}>
        <Text style={styles.subtitle}>Завантаження медіа</Text>
        <FilePicker onFilePicked={handleFilePicked} />
        <UrlInput
          onUrlSubmit={handleUrlSubmit}
          onDownloadUrlSubmit={handleDownloadUrlSubmit}
        />
      </View>
      {downloadUrl !== "" && (
        <DownloadManager
          downloadUrl={downloadUrl}
          onDownloadComplete={handleDownloadComplete}
        />
      )}
      {fileUri && (
        <View style={styles.section}>
          <MediaPlayer fileUri={fileUri} onStop={handleStop} />
        </View>
      )}
    </ScrollView>
  );
};

const styles = StyleSheet.create({
  container: {
    padding: 20,
    alignItems: "center",
    justifyContent: "center",
    flexGrow: 1,
    backgroundColor: "#fff",
  },
  title: {
    fontSize: 24,
    marginVertical: 20,
    fontWeight: "bold",
  },
  subtitle: {
    fontSize: 18,
    marginVertical: 10,
  },
  section: {
    width: "100%",
    marginBottom: 20,
    alignItems: "center",
  },
});
```

```
export default App;
```

Файл DownloadManager.tsx

```
// Компонент для завантаження файлу за введеним URL та збереження його в медіа-бібліотеці.
import React, { useState } from "react";
import { Button, Text, StyleSheet, View, Alert } from "react-native";
import * as FileSystem from "expo-file-system";
import * as MediaLibrary from "expo-media-library";

interface DownloadManagerProps {
  downloadUrl: string;
  onDownloadComplete: (uri: string) => void;
}

const DownloadManager: React.FC<DownloadManagerProps> = ({
  downloadUrl,
  onDownloadComplete,
}) => {
  const [isDownloading, setIsDownloading] = useState(false);
  const [progress, setProgress] = useState(0);

  const downloadAndSaveFile = async () => {
    const extension = downloadUrl.endsWith(".mp4") ? ".mp4" : ".mp3";
    const fileUri = FileSystem.documentDirectory + "downloadedFile" + extension;
    setIsDownloading(true);
    setProgress(0);

    try {
      const downloadResumable = FileSystem.createDownloadResumable(
        downloadUrl,
        fileUri,
        {},
        (downloadProgress) => {
          const prog =
            downloadProgress.totalBytesWritten /
            downloadProgress.totalBytesExpectedToWrite;
          setProgress(prog);
        }
      );
      const result = await downloadResumable.downloadAsync();
      if (result && result.uri) {
        const { status } = await MediaLibrary.requestPermissionsAsync();
        if (status !== "granted") {
          Alert.alert(
            "Permission required",
            "Permission to access media library is required to save file."
          );
          setIsDownloading(false);
          return;
        }
        const asset = await MediaLibrary.createAssetAsync(result.uri);
```

```
        await MediaLibrary.createAlbumAsync("Download", asset, false);

        setIsDownloading(false);
        onDownloadComplete(result.uri);
      } else {
        setIsDownloading(false);
        console.error("Download failed: result is undefined or missing uri");
      }
    } catch (error) {
      console.error("Download error: ", error);
      setIsDownloading(false);
    }
  };

  return (
    <View style={styles.container}>
      <Button
        title="Завантажити файл за URL"
        onPress={downloadAndSaveFile}
        disabled={isDownloading}
      />
      {isDownloading && (
        <Text style={styles.progressText}>
          Завантаження... {Math.round(progress * 100)}%
        </Text>
      )}
    </View>
  );
};

const styles = StyleSheet.create({
  container: {
    marginVertical: 5,
    alignItems: "center",
  },
  progressText: {
    marginTop: 10,
  },
});

export default DownloadManager;
```

Файл UrlInput.tsx

```
// Компонент для введення URL: дозволяє відтворювати або завантажувати медіа-файл.
import React, { useState } from "react";
import { View, TextInput, Button, StyleSheet, Text } from "react-native";

interface UrlInputProps {
  onUrlSubmit: (url: string) => void;
  onDownloadUrlSubmit: (url: string) => void;
}
```

```tsx
const UrlInput: React.FC<UrlInputProps> = ({
  onUrlSubmit,
  onDownloadUrlSubmit,
}) => {
  const [url, setUrl] = useState("");

  const handlePlaySubmit = () => {
    if (url.trim() !== "") {
      onUrlSubmit(url.trim());
      setUrl("");
    }
  };

  const handleDownloadSubmit = () => {
    if (url.trim() !== "") {
      onDownloadUrlSubmit(url.trim());
      setUrl("");
    }
  };

  return (
    <View style={styles.container}>
      <Text style={styles.label}>Введіть URL файлу:</Text>
      <TextInput
        style={styles.input}
        placeholder="http://example.com/file.mp4"
        value={url}
        onChangeText={setUrl}
      />
      <View style={styles.buttonsRow}>
        <Button title="Програти" onPress={handlePlaySubmit} />
        <Button title="Завантажити" onPress={handleDownloadSubmit} />
      </View>
    </View>
  );
};

const styles = StyleSheet.create({
  container: {
    marginVertical: 10,
    width: "100%",
    alignItems: "center",
  },
  label: {
    marginBottom: 5,
    fontSize: 16,
  },
  input: {
    width: "90%",
    height: 40,
    borderColor: "gray",
    borderWidth: 1,
    paddingHorizontal: 10,
    marginBottom: 10,
  },
  buttonsRow: {
```

```
    flexDirection: "row",
    justifyContent: "space-around",
    width: "90%",
  },
});

export default UrlInput;
```

Файл FilePicker.tsx

```
//Дозволяє користувачу обрати аудіо або відео файл із пристрою за допомогою expo-document-picker.
import React from "react";
import { Button, StyleSheet, View } from "react-native";
import * as DocumentPicker from "expo-document-picker";

interface FilePickerProps {
  onFilePicked: (uri: string) => void;
}

const FilePicker: React.FC<FilePickerProps> = ({ onFilePicked }) => {
  const pickFile = async () => {
    try {
      const result = await DocumentPicker.getDocumentAsync({
        type: ["audio/*", "video/*"],
      });
      if ("uri" in result && typeof result.uri === "string") {
        onFilePicked(result.uri);
      } else {
        console.error("Невірний тип або не знайдено URI");
      }
    } catch (error) {
      console.error("Error picking file: ", error);
    }
  };

  return (
    <View style={styles.container}>
      <Button title="Обрати медіа файл" onPress={pickFile} />
    </View>
  );
};

const styles = StyleSheet.create({
  container: {
    marginVertical: 5,
  },
});

export default FilePicker;
```

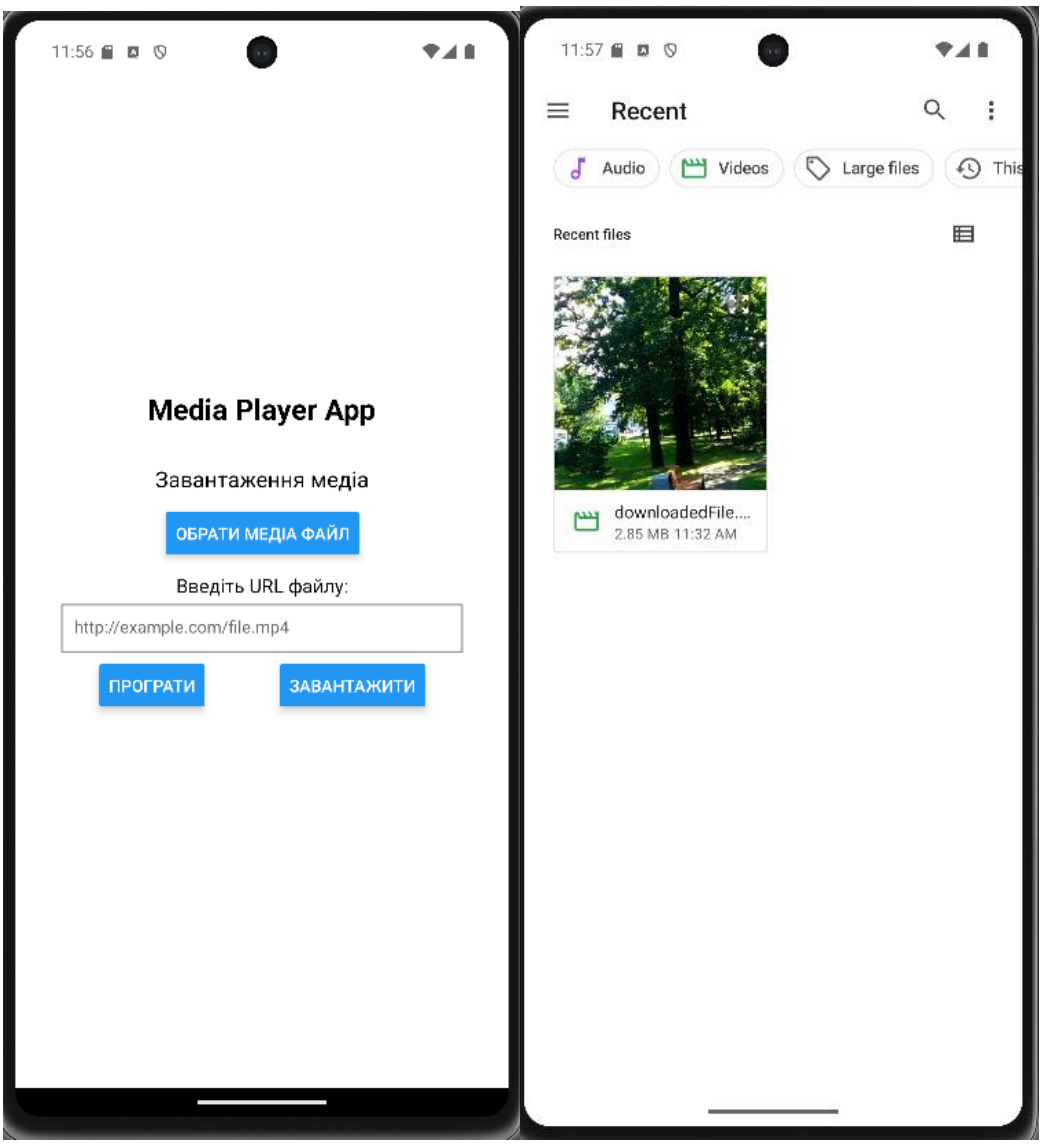
Файл MediaPlayer.tsx

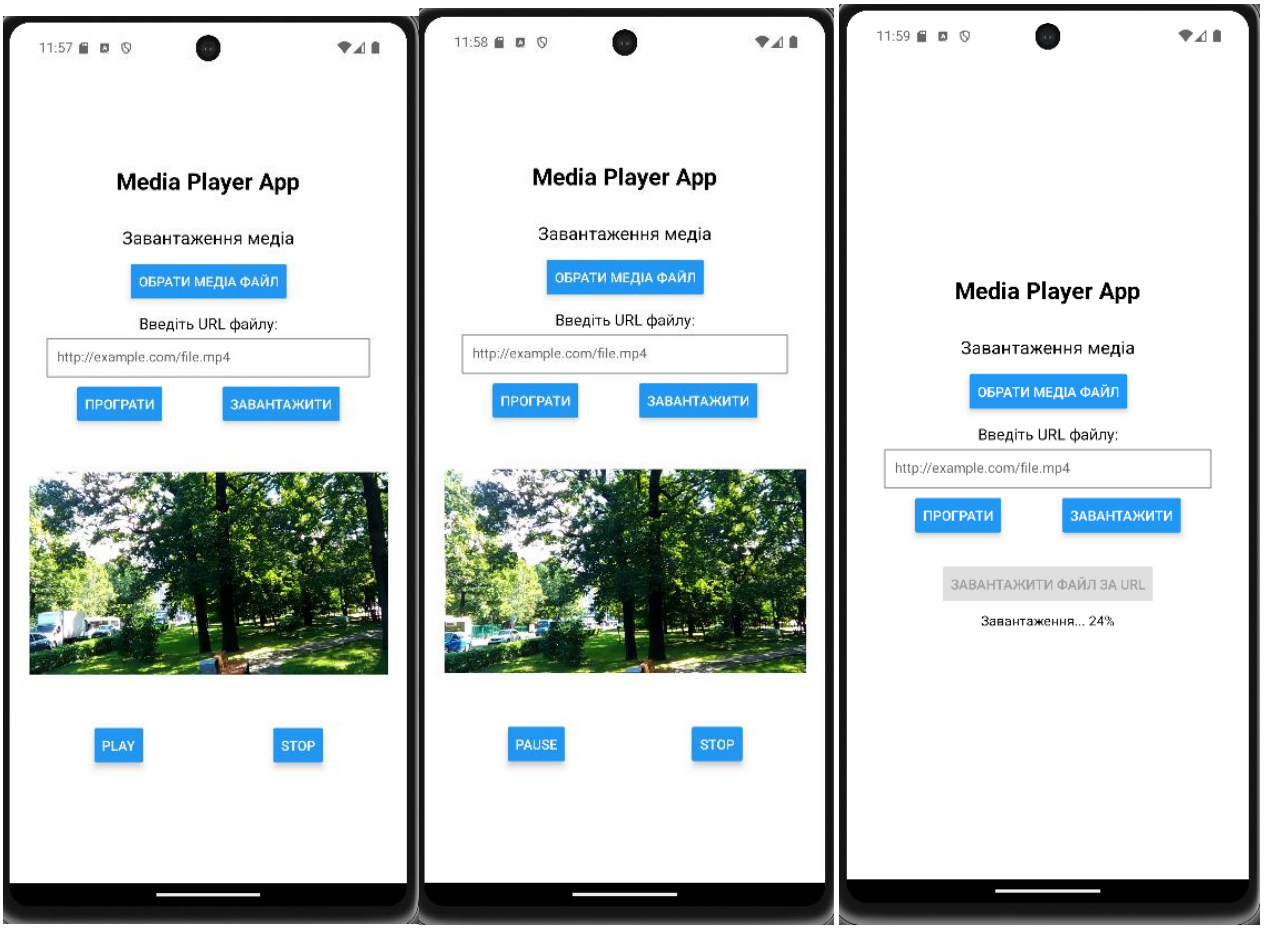
```tsx
// Компонент для відтворення аудіо та відео, підтримує відтворення, паузу та зупинку медіа.
import React, { useState, useEffect, useRef } from "react";
import { View, Button, StyleSheet } from "react-native";
import { Audio, ResizeMode, Video } from "expo-av";

interface MediaPlayerProps {
  fileUri: string;
  onStop: () => void;
}

const MediaPlayer: React.FC<MediaPlayerProps> = ({ fileUri, onStop }) => {
  const [isPlaying, setIsPlaying] = useState(false);
  const [sound, setSound] = useState<Audio.Sound | null>(null);
  const videoRef = useRef<Video>(null);

  useEffect(() => {
    return () => {
      if (sound) {
        sound.unloadAsync();
      }
    };
  }, [sound]);

  const playMedia = async () => {
    try {
      if (fileUri.endsWith(".mp4")) {
        await videoRef.current?.playAsync();
      } else if (
        fileUri.endsWith(".mp3") ||
        fileUri.endsWith(".m4a") ||
        fileUri.endsWith(".wav")
      ) {
        const { sound } = await Audio.Sound.createAsync(
          { uri: fileUri },
          { shouldPlay: true }
        );
        setSound(sound);
      }
      setIsPlaying(true);
    } catch (error) {
      console.log("Error playing media", error);
    }
  };

  const pauseMedia = async () => {
    try {
      if (fileUri.endsWith(".mp4")) {
        await videoRef.current?.pauseAsync();
      } else if (
        (fileUri.endsWith(".mp3") ||
          fileUri.endsWith(".m4a") ||
          fileUri.endsWith(".wav")) &&
        sound
      ) {
        await sound.pauseAsync();
      }
```

```
      setIsPlaying(false);
    } catch (error) {
      console.log("Error pausing media", error);
    }
  };

  const stopMedia = async () => {
    try {
      if (fileUri.endsWith(".mp4")) {
        await videoRef.current?.stopAsync();
      } else if (
        (fileUri.endsWith(".mp3") ||
          fileUri.endsWith(".m4a") ||
          fileUri.endsWith(".wav")) &&
        sound
      ) {
        await sound.stopAsync();
      }
      setIsPlaying(false);
      onStop();
    } catch (error) {
      console.log("Error stopping media", error);
    }
  };

  const togglePlayPause = async () => {
    if (isPlaying) {
      await pauseMedia();
    } else {
      await playMedia();
    }
  };

  return (
    <View style={styles.mediaContainer}>
      {fileUri.endsWith(".mp4") ? (
        <Video
          ref={videoRef}
          source={{ uri: fileUri }}
          style={styles.media}
          resizeMode={ResizeMode.CONTAIN}
          shouldPlay={isPlaying}
          isLooping
        />
      ) : null}
      <View style={styles.controls}>
        <Button
          title={isPlaying ? "Pause" : "Play"}
          onPress={togglePlayPause}
        />
        <Button title="Stop" onPress={stopMedia} />
      </View>
    </View>
  );
};
```

```
const styles = StyleSheet.create({
  mediaContainer: {
    width: "100%",
    height: 300,
    justifyContent: "center",
    alignItems: "center",
  },
  media: {
    width: "100%",
    height: "100%",
  },
  controls: {
    flexDirection: "row",
    justifyContent: "space-around",
    width: "100%",
    marginTop: 10,
  },
});

export default MediaPlayer;
```

Скріни виконання програми у віртуальному телефоні


11:56
Media Player App
Завантаження медіа
ОБРАТИ МЕДІА ФАЙЛ
Введіть URL файлу:
http://example.com/file.mp4
ПРОГРАТИ
ЗАВАНТАЖИТИ
11:57
Recent
Audio
Videos
Large files
Recent files
downloadedFile....
2.85 MB 11:32 AM


11:57
Media Player App
Завантаження медіа
ОБРАТИ МЕДІА ФАЙЛ
Введіть URL файлу:
http://example.com/file.mp4
ПРОГРАТИ
ЗАВАНТАЖИТИ
PLAY
STOP
11:58
Media Player App
Завантаження медіа
ОБРАТИ МЕДІА ФАЙЛ
Введіть URL файлу:
http://example.com/file.mp4
ПРОГРАТИ
ЗАВАНТАЖИТИ
PAUSE
STOP
11:59
Media Player App
Завантаження медіа
ОБРАТИ МЕДІА ФАЙЛ
Введіть URL файлу:
http://example.com/file.mp4
ПРОГРАТИ
ЗАВАНТАЖИТИ
ЗАВАНТАЖИТИ ФАЙЛ ЗА URL
Завантаження... 24%