Hướng dẫn tích hợp PDF Editor

Tài liệu này hướng dẫn cách tích hợp PDF Editor vào ứng dụng ReactJS và React Native của bạn.

1. Tích hợp với ReactJS (iframe)

1.1. Cài đặt

Không cần cài đặt thêm thư viện, chúng ta sẽ sử dụng iframe HTML thuần.

1.2. Component Example

```
import React, { useEffect, useRef, useState } from 'react';

const PDFEditor = () => {
  const editorRef = useRef(null);
  const [isEditorReady, setIsEditorReady] = useState(false);

  useEffect(() => {
    // Lắng nghe message từ editor
    const handleMessage = (event) => {
      // Trong production cần verify origin
      // if (event.origin !== "YOUR_EDITOR_ORIGIN") return;

      const { type, data } = event.data;

      switch(type) {
        case 'EDITOR_READY':
          setIsEditorReady(true);
          console.log('Editor is ready');
          break;

        case 'PDF_SAVED':
          if (data instanceof Blob) {
            // Xử lý PDF đã lưu
            handleSavedPDF(data);
          }
          break;
      }
    };

    window.addEventListener('message', handleMessage);
    return () => window.removeEventListener('message', handleMessage);
  }, []);

  const handleFileChange = (event) => {
    const file = event.target.files[0];
    if (file && isEditorReady && editorRef.current) {
      editorRef.current.contentWindow.postMessage({
        type: 'LOAD_PDF',
        data: file
      }, '*');
    }
  };

  const handleImageSelect = (event) => {
    const file = event.target.files[0];
    if (file && isEditorReady && editorRef.current) {
      editorRef.current.contentWindow.postMessage({
        type: 'ADD_IMAGE',
        data: file
      }, '*');
    }
```

```
  };

  const handleSave = () => {
    if (isEditorReady && editorRef.current) {
      editorRef.current.contentWindow.postMessage({
        type: 'SAVE_PDF'
      }, '*');
    }
  };

  const handleSavedPDF = (blob) => {
    const url = URL.createObjectURL(blob);
    const a = document.createElement('a');
    a.href = url;
    a.download = 'edited-document.pdf';
    document.body.appendChild(a);
    a.click();
    document.body.removeChild(a);
    URL.revokeObjectURL(url);
  };

  return (
    <div>
      <div style={{ marginBottom: '10px' }}>
        <input
          type="file"
          accept="application/pdf"
          onChange={handleFileChange}
        />
        <input
          type="file"
          accept="image/*"
          onChange={handleImageSelect}
        />
        <button onClick={handleSave}>Save PDF</button>
      </div>

      <iframe
        ref={editorRef}
        src="https://test-editpdf.vercel.app"
        style={{
          width: '100%',
          height: '90vh',
          border: 'none'
        }}
      />
    </div>
  );
};

export default PDFEditor;
```

1.3. Sử dụng Component

```
import PDFEditor from './components/PDFEditor';

function App() {
  return (
    <div>
      <h1>PDF Editor Integration</h1>
      <PDFEditor />
    </div>
  );
}
```

2. Tích hợp với React Native (WebView)

2.1. Cài đặt

```
npm install react-native-webview
# hoặc
yarn add react-native-webview
```

2.2. Component Example

```
import React, { useRef, useState } from 'react';
import { View, Button } from 'react-native';
import { WebView } from 'react-native-webview';
import * as DocumentPicker from 'react-native-document-picker';
import * as ImagePicker from 'react-native-image-picker';
import { encode as base64Encode } from 'base-64';

const PDFEditorScreen = () => {
  const webViewRef = useRef(null);
  const [isEditorReady, setIsEditorReady] = useState(false);

  // Xử lý message từ WebView
  const handleMessage = (event) => {
    const { type, data } = JSON.parse(event.nativeEvent.data);

    switch(type) {
      case 'EDITOR_READY':
        setIsEditorReady(true);
        console.log('Editor is ready');
        break;

      case 'PDF_SAVED':
        // Xử lý PDF đã lưu
        handleSavedPDF(data);
        break;
    }
  };

  // Chọn file PDF
  const pickPDF = async () => {
    try {
      const result = await DocumentPicker.pick({
        type: [DocumentPicker.types.pdf],
      });

      // Đọc file thành base64
      const response = await fetch(result.uri);
      const blob = await response.blob();
      const reader = new FileReader();

      reader.onload = () => {
        const base64data = reader.result.split(',')[1];
        // Gửi PDF đến editor
        const jsCode = `
          window.postMessage({
            type: 'LOAD_PDF',
            data: Uint8Array.from(atob('${base64data}').split('').map(c => c.charCodeAt(0)))
          }, '*');
        `;
        webViewRef.current?.injectJavaScript(jsCode);
      };
```

```
      reader.readAsDataURL(blob);
    } catch (err) {
      console.error(err);
    }
  };

  // Chọn hình ảnh
  const pickImage = async () => {
    const result = await ImagePicker.launchImageLibrary({
      mediaType: 'photo',
      includeBase64: true,
    });

    if (result.assets?.[0]?.base64) {
      const jsCode = `
        window.postMessage({
          type: 'ADD_IMAGE',
          data: Uint8Array.from(atob('${result.assets[0].base64}').split('').map(c => c.charCodeAt(0)))
        }, '*');
      `;
      webViewRef.current?.injectJavaScript(jsCode);
    }
  };

  // Lưu PDF
  const savePDF = () => {
    const jsCode = `
      window.postMessage({
        type: 'SAVE_PDF'
      }, '*');
    `;
    webViewRef.current?.injectJavaScript(jsCode);
  };

  // Xử lý PDF đã lưu
  const handleSavedPDF = (base64PDF) => {
    // Implement lưu file theo nền tảng
    // Ví dụ: Sử dụng react-native-fs để lưu file
    // hoặc chia sẻ file qua Share API
  };

  return (
    <View style={{ flex: 1 }}>
      <View style={{ flexDirection: 'row', padding: 10 }}>
        <Button title="Pick PDF" onPress={pickPDF} />
        <Button title="Add Image" onPress={pickImage} />
        <Button title="Save PDF" onPress={savePDF} />
      </View>

      <WebView
```

```
        ref={webViewRef}
        source={{ uri: 'https://test-editpdf.vercel.app' }}
        onMessage={handleMessage}
        javaScriptEnabled={true}
        style={{ flex: 1 }}
      />
    </View>
  );
};

export default PDFEditorScreen;
```

2.3. Sử dụng Component

```
import PDFEditorScreen from './screens/PDFEditorScreen';

function App() {
  return (
    <NavigationContainer>
      <Stack.Navigator>
        <Stack.Screen
          name="PDFEditor"
          component={PDFEditorScreen}
          options={{ title: 'PDF Editor' }}
        />
      </Stack.Navigator>
    </NavigationContainer>
  );
}
```

3. API Reference

3.1. Message Types

Type	Description	Data
EDITOR_READY	Editor đã sẵn sàng nhận lệnh	None
LOAD_PDF	Tải PDF vào editor	File hoặc Uint8Array
ADD_IMAGE	Thêm hình ảnh vào editor	File hoặc Uint8Array
SAVE_PDF	Yêu cầu lưu PDF	None
PDF_SAVED	PDF đã được lưu	Blob hoặc Base64 string

3.2. Lưu ý quan trọng

1. **Security:**
    - Trong production, luôn verify origin của message
    - Sử dụng HTTPS cho editor URL
    - Xử lý lỗi khi đọc/ghi file
2. **Performance:**

- Tối ưu kích thước file trước khi gửi
- Xử lý memory management khi làm việc với file lớn

3. **UX:**

   - Hiển thị loading state khi xử lý file
   - Thông báo lỗi cho người dùng
   - Hỗ trợ cancel operation

4. Troubleshooting

4.1. Vấn đề thường gặp

1. **WebView không load được:**

   - Kiểm tra internet permission
   - Verify URL có thể truy cập
   - Enable JavaScript

2. **File không load được:**

   - Kiểm tra file permission
   - Verify file format
   - Check file size limits

3. **Message không được nhận:**

   - Verify origin configuration
   - Check message format
   - Enable debug mode

4.2. Debug

```
// React/React Native
webView.onError = (syntheticEvent) => {
  const { nativeEvent } = syntheticEvent;
  console.warn('WebView error: ', nativeEvent);
};

// Editor
window.addEventListener('error', (event) => {
  console.error('Editor error:', event.error);
});
```