**PMD - Final report**

Raul Filea

Department of Computer Science,

West University Timisoara

Second Year in Bachelors Degree

email: [raul.filea00@e-uvt.ro](mailto:raul.filea00@e-uvt.ro)

somewhere in March 2021

1. **Abstract**

This report contains basic information about the app **Harul Lugoj LIVE** (NOTE: name might change as a request from the clients, this is why the title doesn’t contain the app name), it explains which are the users and also shortly describes other similar apps, with pros and cons. In the end it presents a basic (non-final) app diagram, using more black-boxes, as it is not relevant at this stage to go into much detail about the internal code, classes, plugins and anything necessary things for this app to work.

1. **Goal and Users**

The **GOAL** of this app is to be able to livestream the sermons of Grace Baptist Church Lugoj (Biserica Crestina Baptista Harul Lugoj) and to use as little resources as possible (in comparison to the website). It has to be as easy to use as possible concerning some **users**.

The **KEY FEATURE** of this app is that the users can watch the stream with *only* **one click (tap)**, instead of having to browse on the internet for the website or YouTube or Facebook.

The **USERS** will be any people that want to listen or to watch the sermons live. The app was developed *especially* for older people, since not all of them are able to navigate on the internet and need a *1-click (tap)* option to watch the church’s sermons.

1. **Introduction**

The app is a playback client (video player of a video stream) that takes as input the stream from a WOWZA server and displays it in full screen mode on user’s device.

1. **State of art**

Other similar apps:

* **Digi Online** – multiple channel streaming app, account needed, not so good UI and has some bugs.
* **Church TV** – Indian streaming app of a church, straight forward easy to use, stops when locking the phone.
* **Church App Live** – simple streaming app, but has more functions (live, archive, donation social media links and others).
* **Victory Church Live** – another advanced app that contains live streaming option.
* **Eurosport Player** – multiple channel streaming app, international support.

(NOTE: all apps are available in Play Store, not sure about iOs store)

1. **Contribution**

The application is fully developed by me, Raul Filea, for both Android and iOs.

1. **Development Plan**

App written in *React Native* using *Expo modules*.

* 1. **Main idea**

The idea of this app is to display the feed from a live stream. NO controlls are needed, as the streaming engine does not offer DVR (for this stream), so users are not able to watch back. The app starts in landscape mode, so eliminating one click (feature required for elders that are unfamiliar with thechnology). It contains no buttons, as none are needed. The video starts playing automatically and the volume is set by the system.

* 1. **Why React Native**

Choosing React Native helped writing only one code for both Android and iOs. As the modules used are updated, the code needs no changes.

* 1. **Why Expo**

Expo offers support for the needs of this app and is perfefctly working with React Native. The modules it provides are free and up-to-date to the always changing market. By simply downloading the latest updates we are able to just create a new APK or IPA and post it in the store, without needing to update the code as well.

* 1. **Expo Modules**

Three main Expo modules were used (not counting aditional background modules), namely: **expo av, expo video** and **expo ScreenOrientation**. Link to the documentation of these modules in **References**.

* 1. **Future development**

Future development consists of adding a notification for when the stream starts (unable to test it with *this* aplication for the moment, since the link it uses for tests is different from the final link) and publishing it to Play Store and App Store.

1. **The Code**

**App.js** consists of a **<View>** containing the **<Video>** that automatically takes the feed and displays it on the screen from the source:

source={{uri:*https://58ffa4254ef21.streamlock.net:443/harul/smil:harul.smil/playlist.m3u8*',}}

Before the **<Video>** loads, it automatically changes orientation to *Landscape* (either left or right depending on user’s device orientation, left by default):

onLoad={changeScreenOrientation}

…

async function changeScreenOrientation() {  
 await ScreenOrientation.lockAsync(  
 ScreenOrientation.OrientationLock.*LANDSCAPE*,  
 );  
}

The file also contains the style snippet, which is referred by both <View> and <Video>:

const styles = ***StyleSheet***.create({  
 container: {  
 width: ***Dimensions***.get('window').height,  
 height: ***Dimensions***.get('window').height \* (9 / 16),  
 backgroundColor: '#0f2d41',  
 },  
});

1. **References**

https://play.google.com/store

<https://docs.expo.io/versions/latest/sdk/av/>

<https://docs.expo.io/versions/v41.0.0/sdk/screen-orientation/>

<https://docs.expo.io/versions/v41.0.0/sdk/video/>