CISC 322/326 Assignment 3
Concrete Architecture of Kodi

Group 11
Schuyler Good
Dylan Walker
Tim Mah
Akshay Desale
Qays Ebrahim
Lucas Gordon

December 5th

Abstract (0.5 Pages)

Our team came up with many good ideas and iterations on how to enhance Kodi's current features and modify its architecture. After discussion and analysis, we decided to go with the enhancement of adding a colour blind mode to Kodi. This section in the settings would allow users to enable different colour blind modes catered to different types of colour blindness such as protanopia, deuteranopia, and tritanopia. The effect this would have on the architecture of the software is mostly centered around the views manager, GUI, and settings components. We discussed how the enhancement would affect the interactions between subsystems, as well as outlined how the effects would take place in two use cases. To play a video in colour blind mode, the user would have to navigate to settings, enable the colour blind mode, then play a video as usual, behind the scenes, the views manager would have switched to the colour blind view, so when the video is sent to the player core, it displays its colour blind variant to the user in the GUI. The effects this has on the system are minimal in terms of Maintainability and performance. Overall the report dives into the specifics of our proposed colourblind mode enhancement and the implementation, effects, and use cases it may have.

Interactions with other features (0.5 - 1 Pages)

If our enhancement were to be integrated into Kodi it would change quite a few of the interactions between subsystems within the application. While colour blind mode would not require its own component, adding it to the architecture would create new dependencies and interactions between components.

Views Manager

The views manager will be the main component affected because it would have to also include the various types of colourblind skins/views that users would be able to select. The views manager would get input from the settings and update the other components in the architecture accordingly. It will also output the correct view to the user interface.

Settings

The settings component will need to be updated and connected to the views manager to facilitate this addition. Users will be able to turn on and off coloblind mode through a setting in this panel and they will also be able to select which type of colourblind mode they wish to use. That information will need to be communicated to the views manager to select the correct view.

User Interface

The user interface will get user inputs regarding colour blind settings and send them to the settings panel where they will be processed. It will also be responsible for displaying the correct view to the user that it receives from the views manager similar to how skins are displayed.

Metadata

Metadata will also need to be updated based on which videos have which specific colourblind compatibility. When a video is added to each user's library, one of the metadata categories we would propose to add would be colour blind compatibility and which types of colourblind views it supports. Therefore when a user wishes to watch titles that are compatible with colourblind mode they can sort for them using a tag.

Database

Similar to metadata, the database would also hold tags for videos with each type of colourblind compatibility. The database would be sortable by these tags, so when the views manager communicates to the database that the user wants to view titles with that tag, it can sort and return the relevant media.

Current State of the System (0.5 Pages)

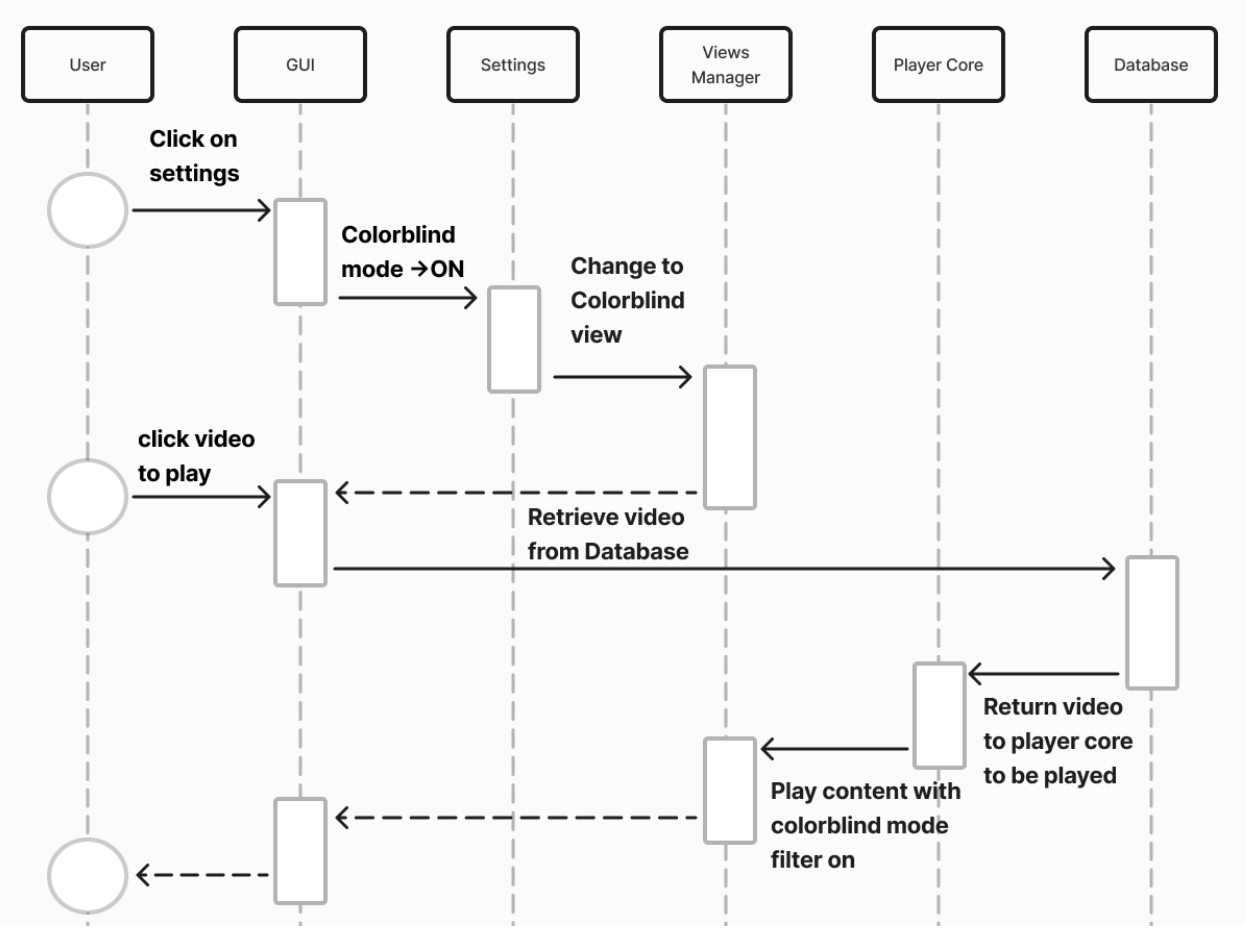
The current state of the system for playing a video relative to the new addition; colour blind mode, is as follows: The user will select the play button on their video using a peripheral device such as a keyboard or a mouse. The UI element for the video that was clicked on will parse this click and send it off to the Metadata component to retrieve the metadata about this video. The metadata is then sent to the Views Manager which takes this data and retrieves the actual video file from the Database where it then sends to the Player core to be played. The Player Core then plays the media to the UI.

Looking at the concrete architecture and this use case, we can see that the location where the colour blind component would go is relatively close to the use being placed around the player core or the views manager, which are both linked directly to the user interface. Since the proposed component would make use of either the Player Core, the Views Manager, or both, it makes sense that these components would first go through the new component on its way to the user interface, or at the very least have a connection to the component which also has a connection to the user interface. This way the videos that are passed to the user must go through this component to get there.

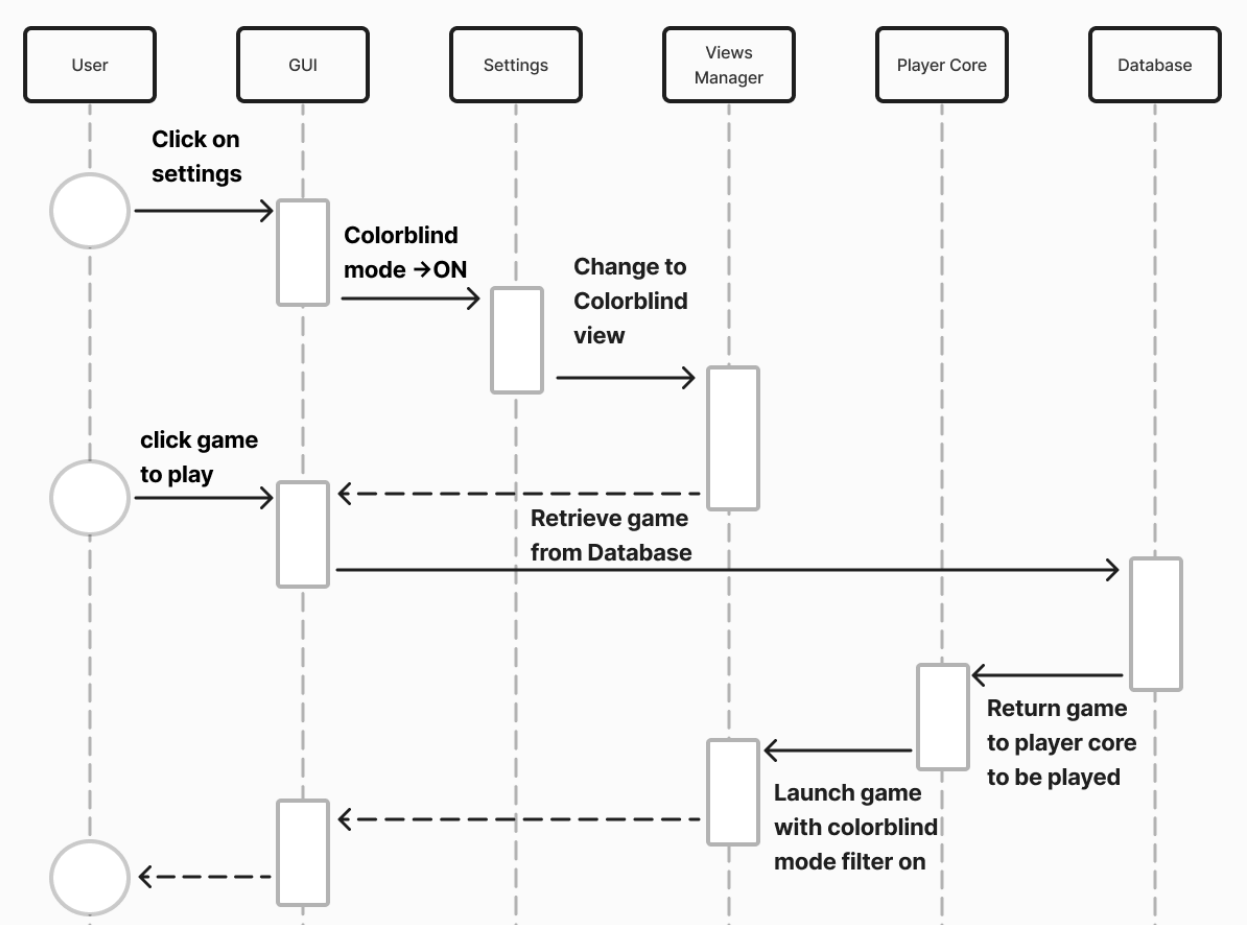
Two use cases and sequence diagrams (1-1.5 Pages)

To envision how our proposed enhancement will interact with our other components, here are two use cases with sequence diagrams outlined below. The first use case showcases the user turning on colourblind mode and playing a video the second use case showcases the user turning on colourblind mode and launching a game.

Use Case 1: Playing a video with colour blind mode



In this use case, the user wants to play a video in the colour blind mode. To do this the player needs to enable colour blind mode and then play a video. To start the user would click on the settings icon in the GUI and turn the colour blind mode to ON. Then the settings component would access the views manager component and change to the colour blind view. Now the user would select a video and then retrieve this video from the database. The database would then return the chosen video to the player core to be played, it then access the views manager to display the content in colourblind mode. Finally the video would be transferred to the GUI for the user to watch.



In this use case, the user wants to play a video game in the colour blind mode. To do this the player needs to enable colour blind mode and then launch the game. To start the user would click on the settings icon in the GUI and turn the colour blind mode to ON. Then the settings component would access the views manager component and change to the colour blind view. Now the user would select their game and then retrieve it from the database. The database would then return the chosen game to the player core to be played, it then accesses the views manager to display the content in colourblind mode. Finally, the game would be transferred to the GUI for the user to play.

Effects on System

Maintainability

Maintaining Kodi with the addition of our proposed colour blind mode would not be difficult, provided proper documentation. Any changes to visuals or metadata will impact our proposed colour blind mode. The documentation should describe what each method and class does in the colourblind sub-system. Additionally, for each colour blind instance or class method, the rationale for each should be documented, in case changes to the system need to be made.

Evolvability

As Kodi grows, new features will need to take into account the new colour blind mode. The colour blind module will need to access new features. All new visuals and GUI elements need to be compatible and fully support all types of colour blind modes. The new feature should be well documented and include comprehensive in-line comments, to help future system maintenance and expansion. All classes that deal with colour blind mode should be tagged, and their class/instance methods need thorough documentation to make them easier to implement in future features.

Testability

Some parts of our proposed colour blind mode can be tested by the developers. The developers can test for bugs, visual glitches, and other functionalities. If there is incorrect output, all the factors affecting it need to be found and recorded. All modules in our sub-system should be independently testable and also work together once combined. Additionally, including testing methods in our colour blind classes could be useful. Lastly, direct user feedback is crucial. Building a colour blind mode that does not improve (or harms) the user experience is bad. Whether the adjustability, customization, and defaults are good will result from feedback from people with color blindness.

Performance

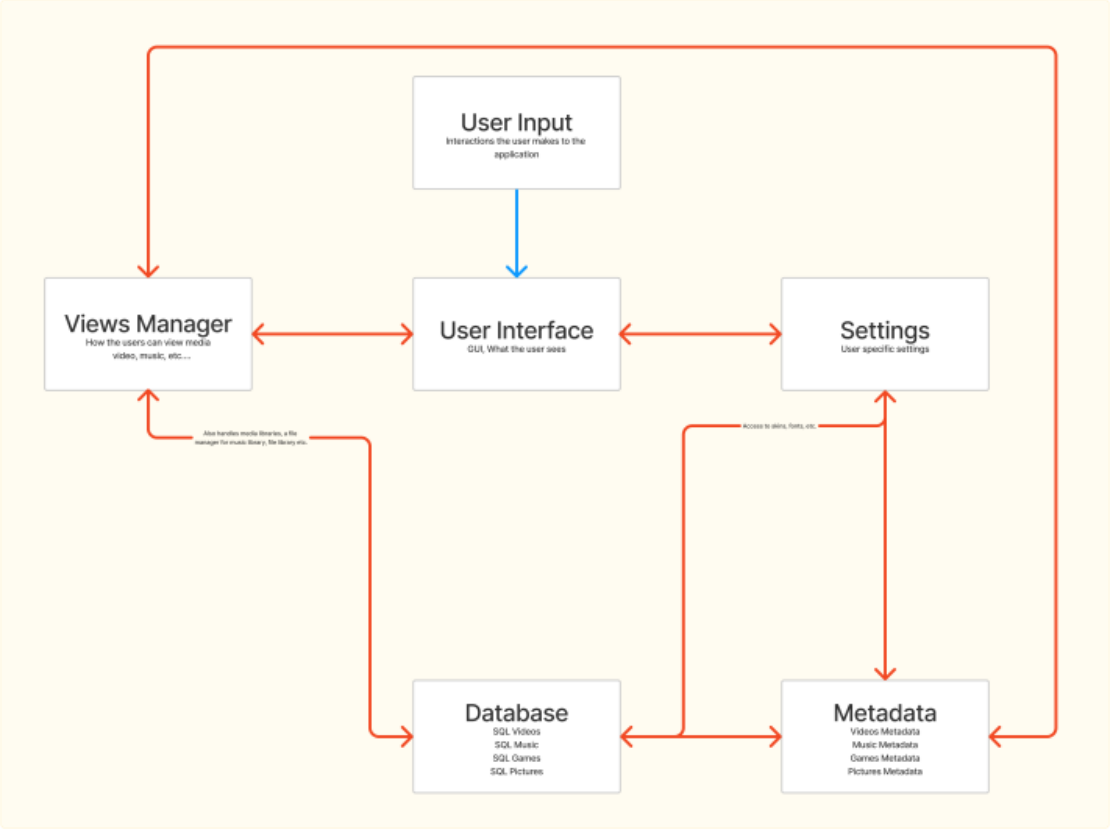
We expect this to make minimal impact to the overall system's performance. Changing the GUI element's colours and contrast would be similar to a Kodi skin/theme, except they would be prebuilt. Visual media will not be converted into a colour blind theme, which could be computationally heavy. Instead, media will be tagged with what colour blind modes it supports. For each mode, there will be a colour/contrasted visual in storage that will play if the mode is on. This means the system does not have to do any major computing. Potentially, a future addition to our proposed new feature could be a one time conversion into all colour blind modes upon media upload. This would eliminate the need for the creator to create separate versions of it (they can if

they want to) but increase the computational power. This would make an already slow upload process even slower and resource intensive.

Alternative Implementation Approaches (1.5 Pages)

Approach 1: Metadata-Based Content Transformation

In approach 1 we can implement the colour blindness mode in 2 parts. Firstly, we can provide the colour blindness mode for the GUI of the application and on the other hand we can support the content of local library or streaming services to be in colour blindness mode. For the first part that is the GUI, we can handle normal and colour blindness mode through views manager, within views manager we can have settings for changing the mode and within setting we can have advance setting where we can handle all the types of colourblind modes namely "protanopia mode" where people have problems perceiving red light and the "deuteranopia"which will be designed for people who have problems perceiving the green light further we can adjust the sensitivity of the colours according to the user, after these settings are set setting component to will interact with GUI component to apply changes on user interface and views manager will toggle to colour-blind mode.
For the other part of implementation which deals with the content of the streaming services and local libraries. Firstly this content for colour blind mode should be generated by the producer of the content. An extra object within metadata will be created to signal if content if available in Colour-blind more or not, (pretty much similar to the way videos are available on streaming services in different audio languages) metadata component will interact with the database to get the metadata, which will be fetched by the views manager and further transferred to the GUI component where it will use this metadata to enable or disable the toggle button for colour-blind mode on the GUI component.



Approach 2: RGB to LMS Colour Space Conversion and Vice versa

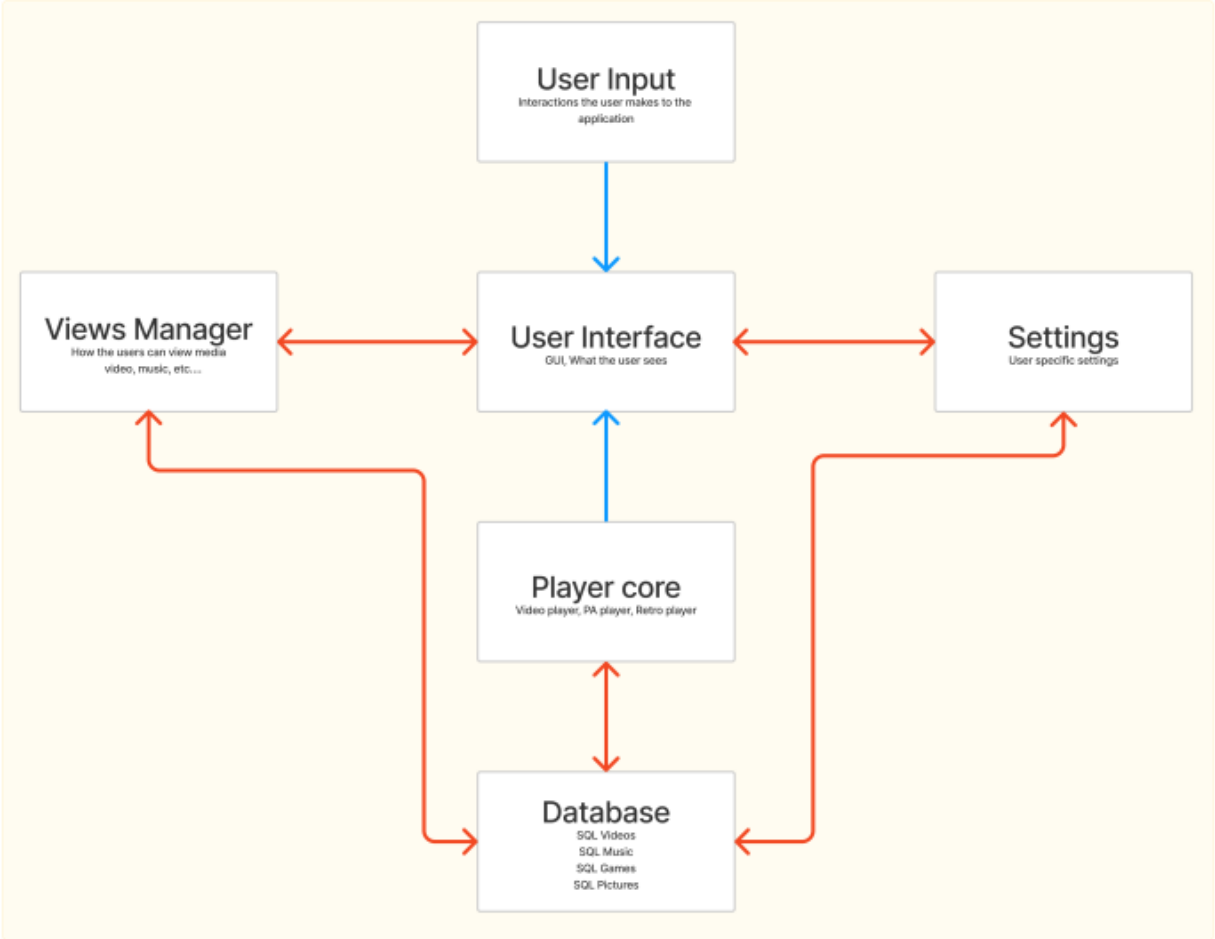
In this approach we develop a colour transformation algorithm which will modify the colours to enhance distinguishability for individuals with vision deficiencies like Protanopia and Deuteranopia. In this method we will leverage colour space conversions like RGB to LMS and vice-versa. It will provide a simulation how people with colour-blindness perceive colous and adjust them accordingly.
For the purpose we will follow following steps:

1. RGB to LMS conversion:
    a. Convert RGB colour values (Red, Green, Blue) into the LMS colour space that mimics human vision perception.
    b. Calculate LMS values from RGB values using transformation matrices or formulas based on cone cell sensitivity in the human retina.
    c. Consider individual colour vision deficiencies (Protanopia, Deuteranopia) by modifying the transformation matrix coefficients to simulate reduced sensitivity to certain wavelengths.
2. Simulating Colour Blindness:

a. Adjust LMS values to simulate colour blindness by attenuating or modifying specific wavelengths that are less perceptible to individuals with colour vision deficiencies.
b. For instance, reduce sensitivity to red light for Protanopia by altering LMS values associated with the red spectrum.

Further after making these changes we display the media content using the modified RGB or LMS values obtained by transformation. We will also apply these modified RGB or LMS values to user interface elements. Further we can also provide parameter adjustments and real time adjustments in the settings.

In this implementation the Setting component will interact with the GUI component to provide an interface to the user to select the preferred colour blindness or normal mode. After this Setting component interacts with the player-core module where we will implement the algorithm for the colour-blind mode, the Player Core component interacts with the GUI components to make the changes in the immediate effect.



SAAM Analysis (1-2 Pages)

Stakeholders

There are two major stakeholders that need to be considered:

- Users: Users own Kodi for personal use. Users can participate in several activities such as browsing their media library, playing media files, or adjusting their display settings. A user's primary use-case is that they want to want their media in as much ease as possible, or they will likely use another software. Media can include both video and audio files.
- Developers: The developers are the key contributors for the development and maintenance of the Kodi platform. Developers can be individual contributors or work for a third party contributor. Developers bring new features and updates to Kodi when necessary.

Non-Functional Requirements

Stakeholder	**NFRs for Kodi**
User	● The software should run smoothly in general ● The software should run smoothly when colourblind mode is enabled ● The colourblind mode should be easy and straightforward to enable or disable ● The interface changes when colourblind mode is enabled or disabled should be intuitive ● The colourblind mode should improve the usability and accessibility of the software
Developers	● The software should not become over complex with the addition of colourblind mode ● The codebase should not become too complex ● The codebase should not become too hard to maintain ● The software should be able to be automatically tested for the colourblind mode functionality

Evaluating Implementation Methods & Choosing The Best One

NFRs	**Approach 1: Metadata-Based Content Transformation**	**Approach 2: RGB to LMS Colour Space Conversion**
Performance	• Minimal impact on system performance • Only involves changes to the metadata associated with the files and the GUI	• Greater impact on system performance • Real-time colour space conversions
Usability	• Users would manually select media files that support their specific type of colour blindness • Less user friendly	• Auto adjust all colours based on the users imputed colour blindness • More user friendly for the user
Accessibility	• Only improves accessibility for media files that have been specifically marked as compatible with different types of colour blindness	• Improves accessibility for all of the media files • Does not matter if the files have been marked or not as compatible
Maintainability	• Require maintenance for the additional metadata for each media file • Could lead to increased complexity	• Require maintaining a colour conversion algorithm • Does not require any changes to the file metadata
Testability	• Can be tested by checking if the correct metadata is applied to the specific media file • If the user UI responds to the metadata	• Check to see if the algorithm performs as described and converts file display

As we have discussed the advantages and disadvantages above, we have determined that Approach #2 is the best solution to address the NFRs described above and for the stakeholders of Kodi.

Effects on Concept Architecture (1-1.5 Pages)

on High and low-level Conceptual Architecture

Impacted directories and files (0.5 Pages)

To talk about impacted directories and files let us firstly explore our approaches and what features of the Kodi will be affected in each of the approaches of the implementation of the colour-blind mode.
Approach 1: For the implementation of the Metadata based content transformation we have to see that components which are getting affected are GUI, View Manager, Settings, Metadata, Database. Firstly, as we have observed in our previous study that every module has its own GUI component and the setting components inside the directories all the directories containing modules like video, audio, weather, photos will be affected. Other than that all the Utils directory, setting directory, interfaces directory and rendering directory will get affected. All these directories are under the xbmc directory. Other than that video, audio, music and weather directories outside xbmc will also get affected as they have their own metadata, GUI and setting classes in each module. Database will also get affected as now searching and other operations will be involved based on the colour-blindness characteristics of the data file.

Approach 2: For the implementation of the RGB to LMS Colour Space Conversion and Vice versa we see that GUI, Settings, Views Manager, Player Core, database these components are getting affected. As our implementation is based on developing an algorithm as discussed in the previous section similar to the above approach all the directories involving the GUI and settings classes will get affected. Other than that rendering directory will get affected which will render the videos and other content based on the outcome of the algorithm and the settings from the user input. On the other hand Player Core will be the directory where we have to write the algorithm for the implementation of this approach so the Player Core directory will get affected during this implementation.

Plans for testing (0.5 Pages)

To test our planned implementation of the colourblind mode feature, we would plan to invite several people from each of the two main types of colour blindness to ensure that the users can see, and more importantly, differentiate between the objects shown on screen. For an estimated sample size, we would aim to test at least 20 people of each type of colour blindness, totalling to 40 people, with an extra 5 people with normal vision as a control group leading to a final sample size of 45 people.

The actual test would consist of each person viewing a predetermined video of several objects moving around the screen, and off of the screen. The objects would be different colours, both similar to the background and other objects, and completely different to each other at times. Specifically, the opposite colourings of the objects should be pairs that users with colour blindness would not be able to discern without help, for example, red and green objects. Then, each participant would attempt to count how many objects appeared on the screen.

The success criteria of our test would be if the colour blind groups can match the normal vision control groups average number to within a 5% margin of error.

This methodology of this test would confirm if the colour blind mode was effective in providing enough colour distinction that the users can easily differentiate between similarity coloured objects, or between objects with which they would have difficulty noticing otherwise. For example, if playing a game where enemies are red and allies are green, then the user should be able to discern between them if the colour blind mode is effective

Potential risks (0.5 Pages)

With the proposed additions to Kodi, there are several risks that are important to measure and consider. The risks are presented below:

Risk	**Description**
Performance	● Whether the implementation of the algorithm will slow down the performance of Kodi
Complexity	● Whether the approach will make Kodi too complex to be maintained by future developers
Usability	● If the mode is implemented correctly, will it be easy enough to be used by users
Testing	● Can the testing of the code be automated to ensure that it is

	functioning as described
Resource Limits	● When a large number of users turn the mode on, will the host computer be a bottleneck for the increase computing power required

To mitigate these risks, Kodi should conduct thorough testing of the new features before they are released.

Conclusion (0.5 Pages)

In conclusion the introduction of colourblind mode into Kodi would enhance user experience within the app. Its introduction would require some new dependencies between the Views Manager, Settings, User Interface, Metadata and Database components. Looking at the current state of the system the addition of colourblind mode would work very closely with the views manager and the user interface. We also demonstrated how the feature would work in our system through two use cases; playing a video with colourblind mode, and launching a game with colourblind mode.

The addition of colourblind mode will also affect the maintainability, evolvability, testability and performance of the system going forward. We believe that as long as the additions are well documented the system will be easy to maintain, it will be able to evolve without much effort. When testing the enhancement, a combination of developer testing and user feedback will ensure a smooth integration into the existing system. Lastly, the enhancement should not damper performance at all and the app will continue to run smoothly with this addition. \

When it comes to implementing the app into the existing framework, there are a few different methods which may be viable. The first would be metadata based content transformation which focuses on changing the view to a color blind mode and modifying metadata to show which titles are playable in said view. The second is to use a colour changing algorithm to map rgb to lms colours and this process would happen in the player core. After reviewing both approaches using SAAM analysis we concluded that the second approach would be favourable because it had the best effect of NFRs that affected our stakeholders. Lastly we identified potential risks that could occur with the addition of this feature, however we concluded that most of these risks could be mitigated through user testing before the application is launched.

Glossary

References

https://www.aao.org/eye-health/diseases/what-is-color-blindness#:~:text=Color%20blindness%20occurs%20when%20you,and%20reds%2C%20and%20occasionally%20blues.