



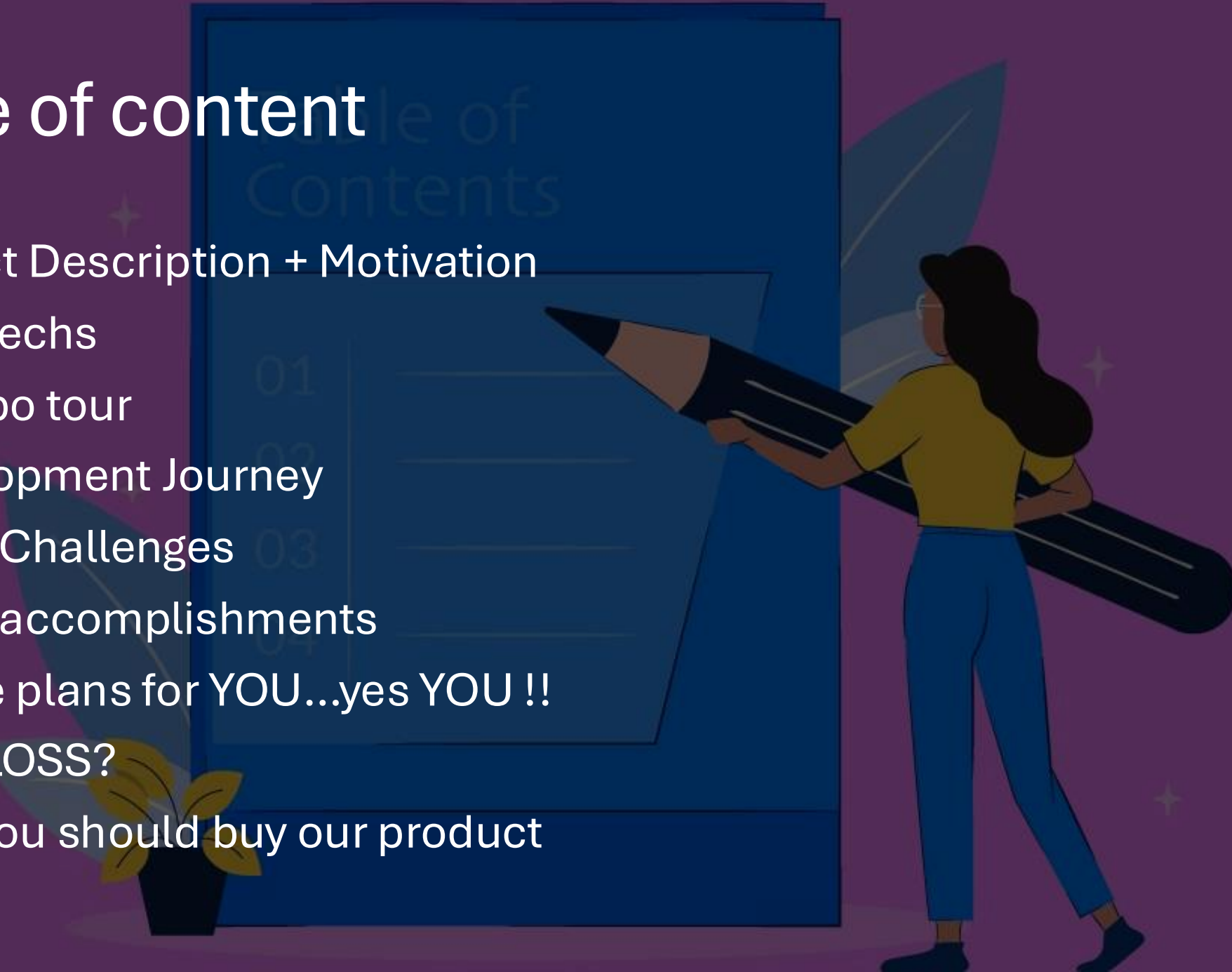
# Unix Project

By: Ryan Dong

Yakin Succès

# Table of content

- Project Description + Motivation
- Core techs
- Git repo tour
- Development Journey
- Major Challenges
- Major accomplishments
- Future plans for YOU...yes YOU !!
- Is it FLOSS?
- Why you should buy our product



# Project Description + Motivation

## 1-Media Player Design

- A Raspberry Pi streams Netflix content through Kodi using the Netflix add-on with a focus on seamless user experience.

## 2- Machine Learning Integration

- Once a Netflix session ends, the Raspberry Pi captures viewing data
- A Python-based script utilizes a pre-trained recommendation model to generate personalized movie recommendation

## 3- Recommendation Delivery

- Recommendations are sent to the user's email after viewing



# Core Techs

- Raspberry Pi Setup: Raspberry Pi 4 and LibreElec
- Media Player Platform: Kodi, software to host and stream Netflix add-on. SQLite which manages watch history in the MyVideos75.db database
- Machine Learning Framework: Python for scripting and running the recommendation engine with pre-built Python Docker images. Libraries (3) -> Pandas for data manipulation and analysis, scikit-learn for ML recommendations models and SQLite3 that has database interaction for watch history analysis.





# Core Techs

- Virtualization & Containerization : Used Docker for isolating the Python-based recommendation engine and ensures compatibility dependency while simplifying dependency management across environments.
- Data Storage & Transfer : Persistent storage thanks to the Raspberry Pi that stores python scripts, pre-trained models, metadata files and Netflix SQLite database.
- File Transfer Tools: Used scp to secure file transfer from host to Raspberry Pi and pscp that is the Windows equivalent of scp for transferring files via PuTTY



# Core Techs

- Dependency Management: Used virtual Environments to isolate Python dependencies like *venv* within Docker containers, Cython to build tools like some libraries and apt packages like build-essential for example.
- Email Notification System: Used *smtplib* (python library) to send emails with generated recommendations to users

# Git repo tour



- This repository contains all the code, scripts, and resources for the Netflix streaming and movie recommender system project.
- We will show you how the repo is structured and what the challenges were in the organization of the repository
- There will be a live walkthrough : <https://github.com/ryandong05/unix-project.git>



# Development Journey

- Yakin's journey :
- Research on how everything worked (my teammate came up with the idea and had prior knowledge) -> Raspberry Pi, Netflix integration and Machine learning
- Learned more about Kodi, Python Scripting and recommendation models
- How to extract watch logs from Kodi (accessing Kodi databases or log files) and learned the email integration package for Python
- Started implementing super LATE !!







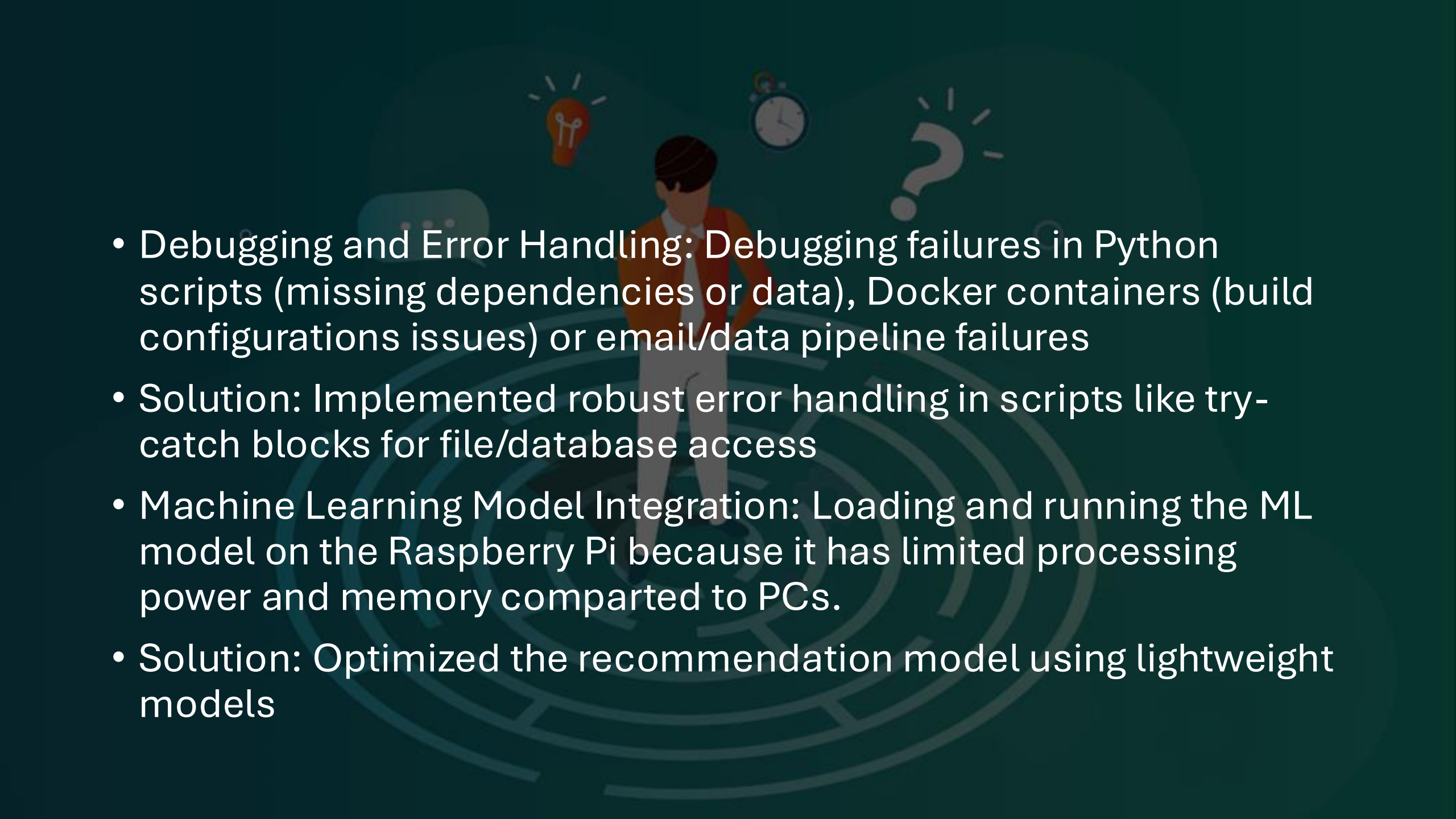
# Ryan's Journey

- Started setting up everything to get going using the instruction manual and added MIT License to Project
- Created a machine learning model using a CBF approach and found a suitable dataset on Netflix movies/TV shows from Kaggle
- Attempted to complete the project and troubleshooted for 12 HOURS STRAIGHT...yes you heard that right
- I might be the FIRST person to have tried running a ML onto the barebones distribution of Kodi, made for watching Netflix and YouTube while it does not have a terminal.

# Major Challenges

- Dependency Management: Installing and managing Python dependencies like pandas and scikit-learn, especially in a limited environment like LibreELEC. (What made me give up)
- Potential Solutions: Used Docker to isolate dependencies and switched to a python based/more general-purpose Linux OS image.
- Step Pray and Install : Had to find version of scikit-learn that contains a setup.py file -> had to go through random versions of git that has it. Run the program and it fails, try using OSMC Kodi BUT there are no 64-bit OSMC versions available so we couldn't try this method.



- 
- Debugging and Error Handling: Debugging failures in Python scripts (missing dependencies or data), Docker containers (build configurations issues) or email/data pipeline failures
  - Solution: Implemented robust error handling in scripts like try-catch blocks for file/database access
  - Machine Learning Model Integration: Loading and running the ML model on the Raspberry Pi because it has limited processing power and memory compared to PCs.
  - Solution: Optimized the recommendation model using lightweight models

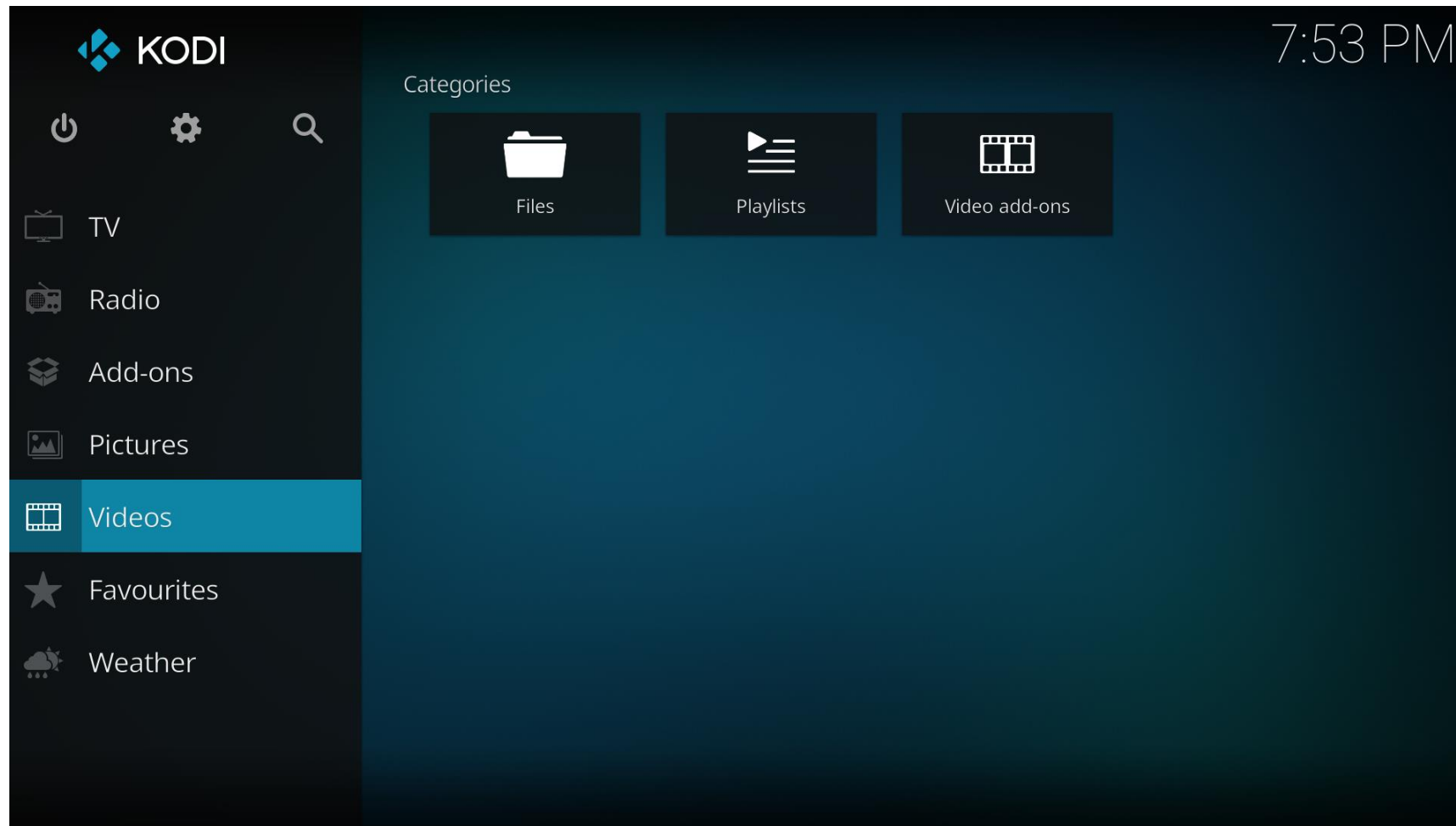
# Major Accomplishments



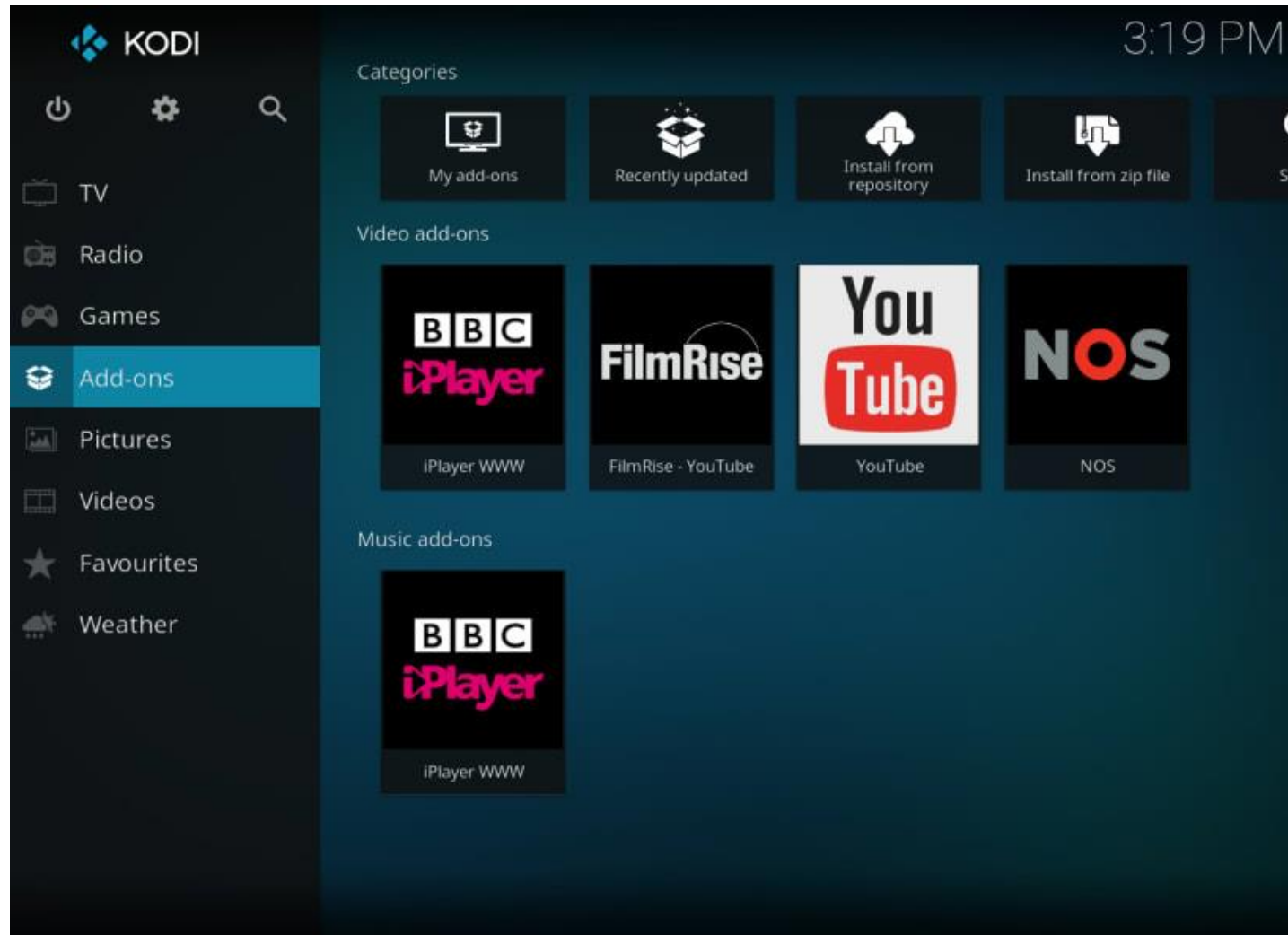
- End-to-End System Integration: Successfully integrated multiple technologies like Raspberry Pi, Kodi, Netflix add-on, Python and Docker into a cohesive system
- Implementation of ML Recommendations: Designed and implemented a pre-trained TF-IDF model with the Netflix database (MyVideos75.db)
- Clear Documentation: Developed a detailed documentation to guide users through installation and usage process while also including troubleshooting tips and alternative setups.
- Effective Use of Raspberry Pi: Raspberry Pi is versatile thanks to it being able to run LibreELEC or OSMC, hosting Docker containers to support Python environment for the ML script



# Kodi GUI



# Kodi GUI





# Demo: Model and Script .py files

- <https://github.com/ryandong05/unix-project>

A close-up photograph of a hand with the index finger pointing directly at the viewer. The hand is positioned in the center of the frame, with the background being a soft, out-of-focus grey. The lighting is even, highlighting the texture of the skin.A solid orange horizontal bar located in the top-left corner of the slide.

# Future Plans for YOU

---

- Improved Recommendation System: Using a more advanced machine learning model for improved accuracy and incorporate real-time learning to update recommendation dynamically.
- Multi-Platform Support: Integrate other streaming services like Amazon Prime Video, Hulu or Disney+ into the recommendation engine
- Broader Accessibility: Add family and group profiles including shared viewing options and support multiple languages
- Advanced Security and Privacy: Use encryption for user data and watch history stored on the device and add support for user authentication to protect profiles and emails





# Is our Project FLOSS?

- Freedom to use and access our media player system and recommendation engine
- Freedom to study everything except for maybe the proprietary software like Netflix or Kodi's Netflix add-on
- Freedom to modify the project's source code, however, the proprietary components like Netflix cannot be legally modified which limits the FLOSS status
- Freedom to share which allows users to freely distribute copies of the software, including modified versions.
- Conclusion: our project is partially FLOSS because the Python scripts, Docket setup and installations steps are qualified as FLOSS while the proprietary dependencies are not.

# Why should someone buy our product?

- Personalized Entertainment Experience: you have PERSONALIZED movie recommendations tailored to YOUR viewing habits, and you receive relevant suggestions based on YOUR preferences, reducing decision fatigue
- Affordable and Customizable Solution: Cheap and our product provides a cost-effective alternative to high-end media systems and users get a fully functional media player with added advantage of being customizable
- Sustainability and Privacy: our product runs on a local Raspberry Pi so it minimizes cloud dependencies and users can retain control of their data, ensuring greater privacy

