# Video Interpolation

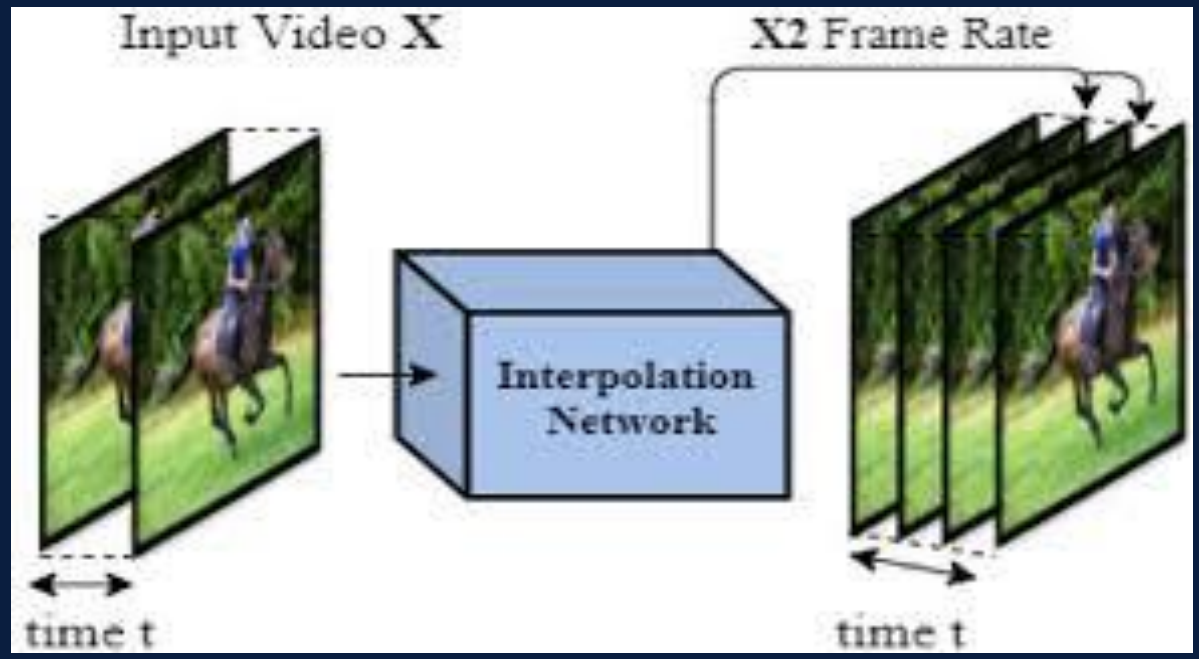
Kevin Puig, Tomas Ambrosini, Nathaniel Rowe, Tahjae Shamari Chamberlain & Nathan Castillo

FIU
FLORIDA INTERNATIONAL UNIVERSITY

COLLEGE OF ENGINEERING AND COMPUTING

# Purpose of the Software

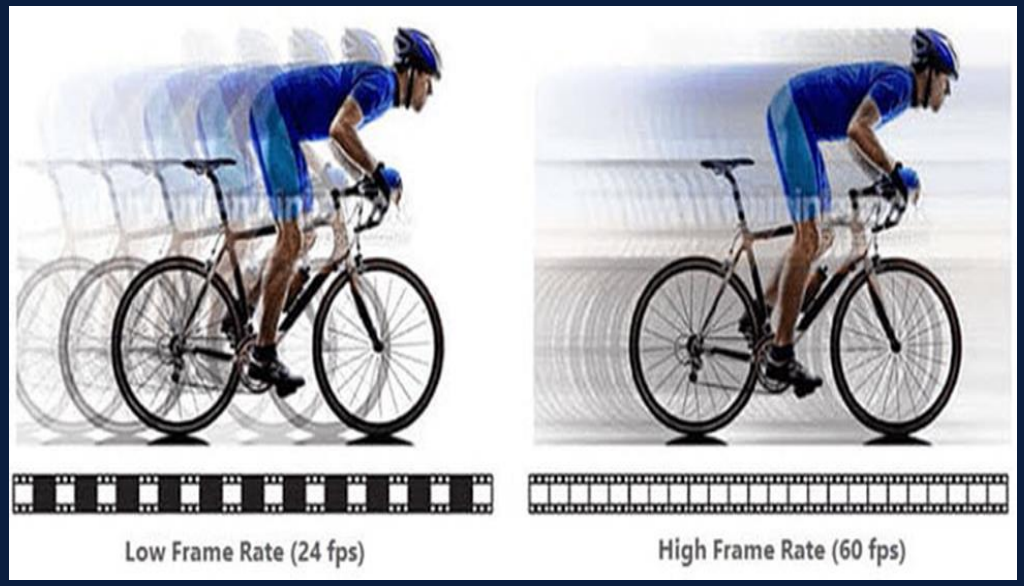
- This program was created to double the frame rate of a video. This helps to improve the quality of the video and viewing experience for the user. The user can upload any video of their choosing and make it look better than it was previously.

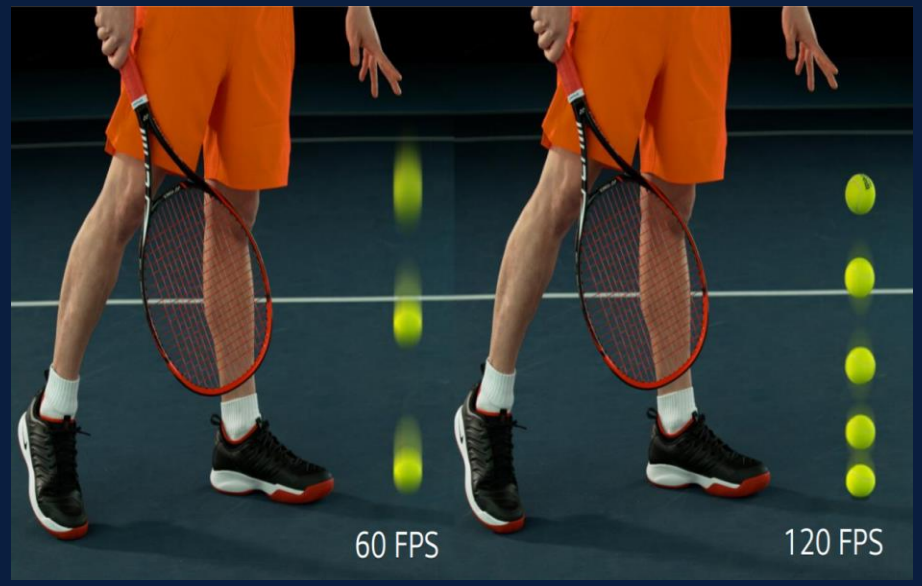



FIU Engineering & Computing

# Scope of the Software

- Improving a video's frame rate, makes it so the video is smoother and more fluid. These improvements makes the video quality better, which in turn makes the video look better visually overall. This can help make many users experience of watching the video much better.




FIU Engineering & Computing

# Software Requirements

## Functional

- Users must be able to upload video files through a web interface.
- The system must assign a unique task ID to each video.
- Videos must be processed via a trained neural network to generate interpolated outputs.
- Status updates (queued, processing, processed, or errored) must be available.
- Users must be able to download the final processed video.
-

## Non-Functional

- The system must respond to user requests within a reasonable time frame (<2s for uploads/status checks).
- Video processing must happen asynchronously.
- The interface must be accessible and intuitive for non-technical users.
- The architecture must support modular codebases across front-end (React), back-end (Java REST API), and processing engine (Python PyTorch).

FIU Engineering & Computing

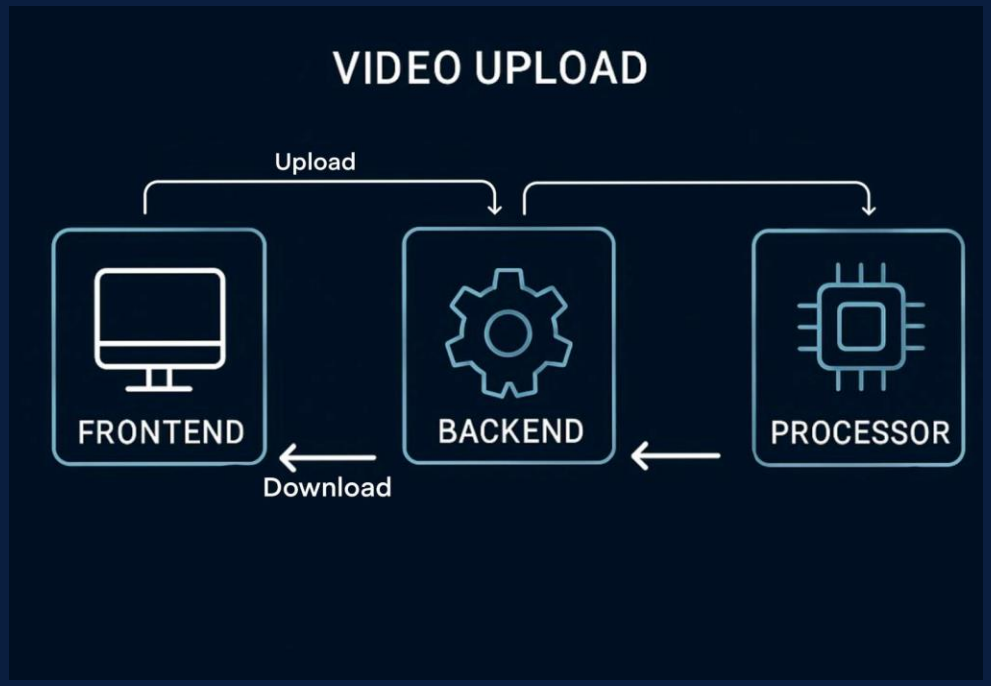
# System Design

Frontend (React): Provides a simple web interface for video uploads and downloads.

Backend (Java REST API): Handles file operations and exposes RESTful endpoints.

Processor (Python + PyTorch): Runs the neural network to generate interpolated videos.
File-Based Queue System: Organizes videos into queue/, processing/, processed/, and error/ folders, allowing loose coupling between Java and Python layers.

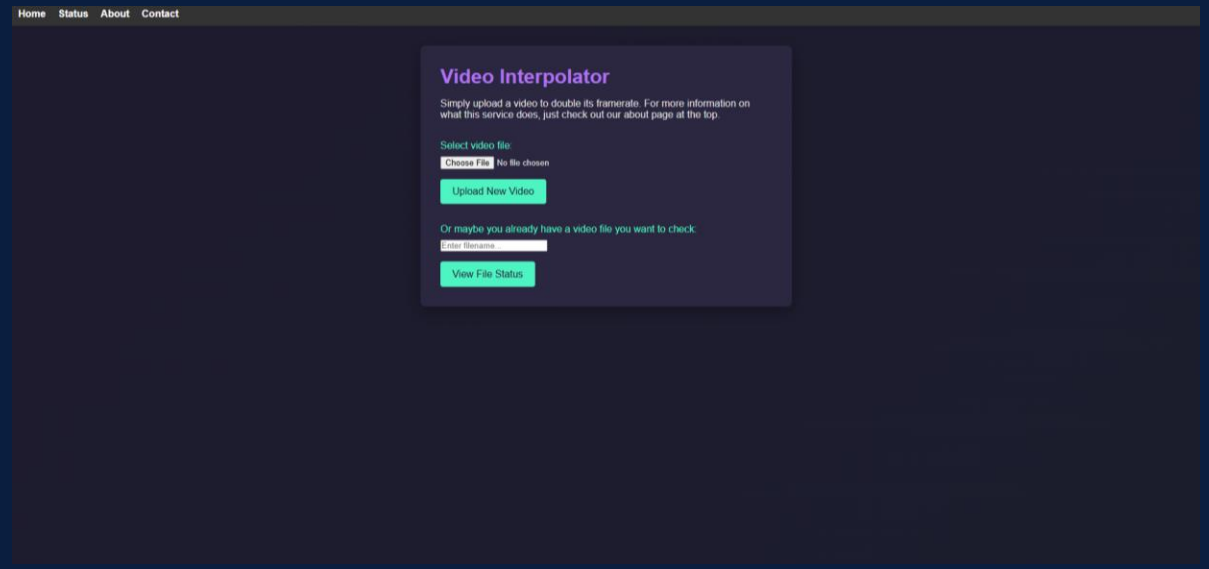
This modular architecture allows each component to function independently and ensures scalability, maintainability, and easier debugging.




FIU Engineering & Computing

# Video Interpolation

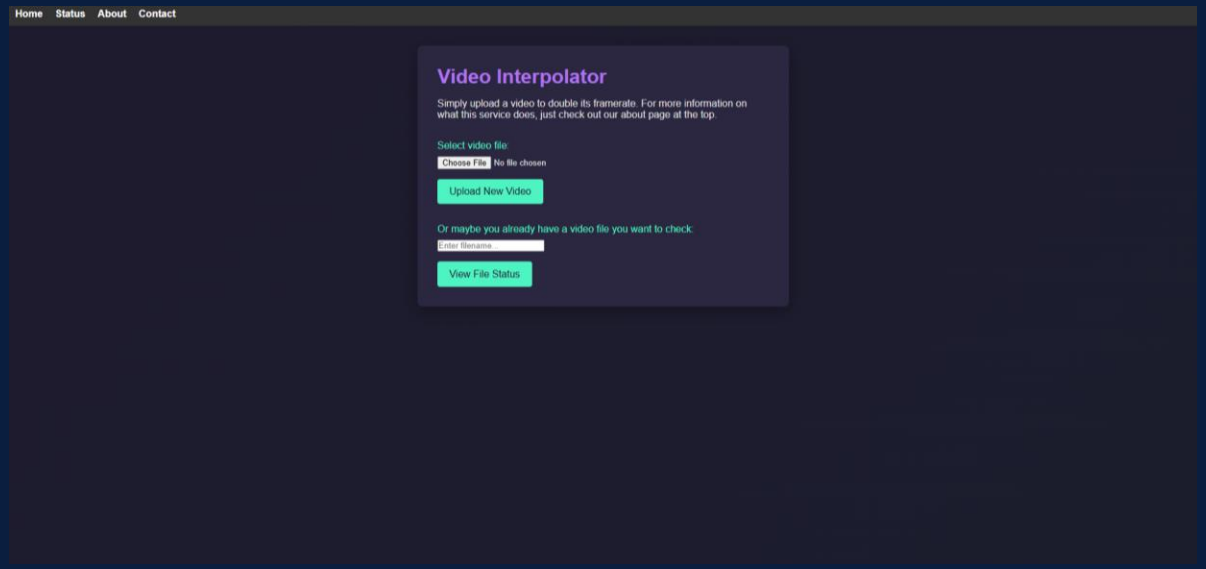
- This the homepage of the website where the user interacts with the software.
- The top navigation bar has four options including home button that navigates to the home page, Status button that states the status of a video interpolation, an About button that describes what our website does, and a contact button to contact us with any issues/bugs!




FIU Engineering & Computing

# Video Interpolation

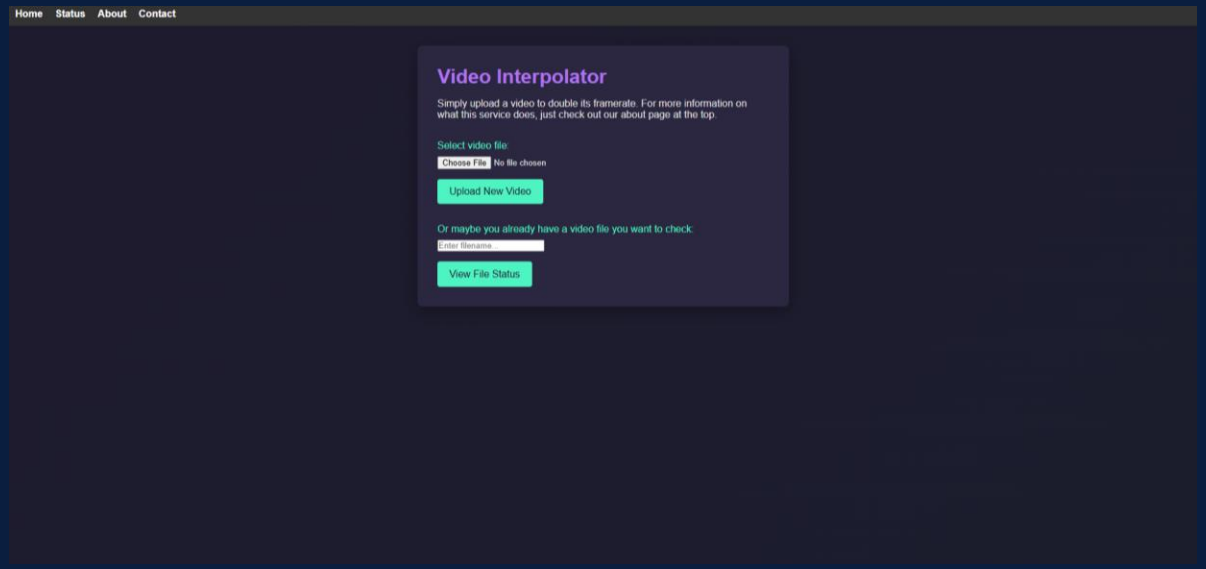
- On the homepage, you are greeted with our implementation of a video interpolator that allows you double the framerate of a video.
- Under the "Select video file" section, there is a "Choose File" button that will open a files window for you to select video file to be interpolated. The "Upload New Video" button will upload your selected video file to be queued for video interpolation.




FIU Engineering & Computing

# Video Interpolation

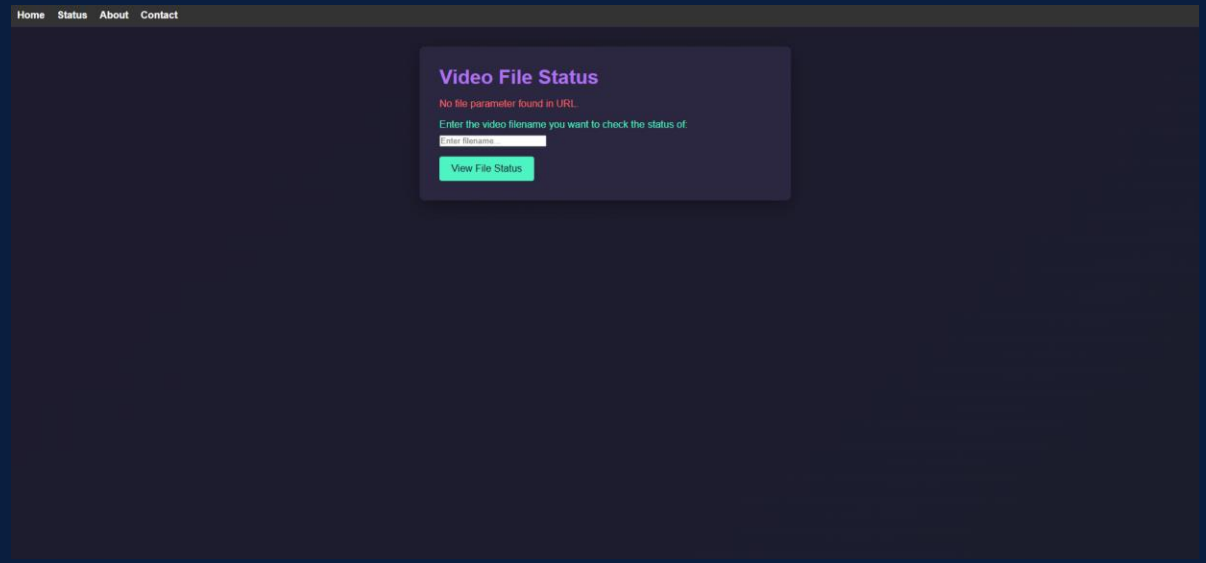
- Once your video file has been uploaded for video interpolation, you may wait for it and check the status by entering the filename into the Entry box below the "Or maybe you already have a video file you want to check" heading.




FIU Engineering & Computing

# Video Interpolation

- When entering a filename, you will directed to the Video File status page. In this page, it will indicate whether the video file has been queued, errored, processing, or processed.
- If you were already on this Status page, you may enter a filename and click the "View File Status" button to view the entered file name's status.




FIU Engineering & Computing

# Hardware Requirements

- A modern GPU with a high amount of VRAM, at least 8GB for smaller resolution video.

# Installation

- Frontend (React)

- Navigate to the frontend directory:
  cd interp-website
- To run locally:
  npm run dev
- To deploy:
  npm run build
- Copy the generated dist/ files to your HTTP server for hosting.

- Backend (Java - Spring Boot)

- Navigate to the backend directory:
  cd nnqueue
- Start the backend server:
  ./gradlew bootRun

- Neural Network Processor (Python)

Navigate to the processor directory:
cd nnprocessor

Create a Python virtual environment:
python3 -m venv venv

Activate the environment:
source venv/bin/activate

Install dependencies:
pip install opencv-python moviepy numpy torch torchvision

Start the processor:
python3 –m nnprocessor

FIU Engineering & Computing