Chat Conversation

Note: *This is purely the output of the chat conversation and does not contain any raw data, codebase snippets, etc. used to generate the output.*

User Input

Project Title: AI-Assisted Face Detector Web Application

Context:
This is a web application that consist in an browser based face detection application that runs entirelly on client side.
This project is part of a University exam.

Main Goal:
design and implement a browser-based face detection application that runs entirely client-side
(e.g. using JavaScript, WebAssembly, or existing in-browser ML libraries),
supporting live camera input and basic visualization of detected faces (bounding boxes).

What is important:

- The software must recognize faces efficiently and quickly
- It must be a web application (in the browser)
- Everything must work within the browser, without external servers
- It must work on all major browsers
- It must respect privacy; nothing goes outside the browser, everything must remain local

Design:

- is a web app designed for desktop (16:9)
- dark palette for the theme
- for the component layout refer to the linked image

Features:

- button for screenshots of the output of the webcam with the bounding boxes
- a button to toggle on and off face detection
- some statistics

Ask me any questions you may have.

*Listed directory [AI-Web-Face-Detector-V2] (file:///home/mario/Desktop/Unipi/Magistrale/advanced-programming/Progetto-esame/AI-Web-Face-Detector-V2) *

Searched filesystem

Edited relevant file

Edited relevant file

Edited relevant file

Edited relevant file

User accepted the command `npm create vite@latest ./ -- --template vanilla`

Checked command status

User accepted the command `npm install && npm install @mediapipe/tasks-vision`

*Listed directory [AI-Web-Face-Detector-V2] (file:///home/mario/Desktop/Unipi/Magistrale/advanced-programming/Progetto-esame/AI-Web-Face-Detector-V2) *

*Listed directory [src] (file:///home/mario/Desktop/Unipi/Magistrale/advanced-programming/Progetto-esame/AI-Web-Face-Detector-V2/src) *

*Viewed [task.md](file:///home/mario/.gemini/antigravity/brain/f6984359-edb7-47f6-913e-8ac5959f7978/task.md) *

Edited relevant file

User accepted the command `mkdir -p src/core src/ui && rm src/counter.js src/javascript.svg`

Checked command status

Edited relevant file

Edited relevant file

Edited relevant file

Edited relevant file

Edited relevant file

Edited relevant file

Edited relevant file

Edited relevant file

Edited relevant file

User accepted the command `npm run build`

Edited relevant file

Edited relevant file

User accepted the command `npm run dev`

User Input

I have notice that the bounding box label are mirrored and are painted with a color that is difficult to see.
mirror back the label on the bounding box and add in the top center what the % value mean
than change the bounding box color to something still visible but also more readable

*Viewed [Renderer.js](file:///home/mario/Desktop/Unipi/Magistrale/advanced-programming/Progetto-esame/AI-Web-Face-Detector-V2/src/ui/Renderer.js) *

Edited relevant file

Edited relevant file

User accepted the command `npm run build`

Edited relevant file

Edited relevant file

User accepted the command `npm run dev`

User Input

Answer me some questions: the vision model run on cpu or gpu? and if it runs on gpu, it can anyway work on every laptop?

*Viewed [Detector.js](file:///home/mario/Desktop/Unipi/Magistrale/advanced-programming/Progetto-esame/AI-Web-Face-Detector-V2/src/core/Detector.js) *

*Viewed [main.js](file:///home/mario/Desktop/Unipi/Magistrale/advanced-programming/Progetto-esame/AI-Web-Face-Detector-V2/src/main.js) *

Planner Response

Based on the current implementation in src/core/Detector.js, here are the answers to your questions:

1. Does the vision model run on CPU or GPU?
It is currently configured to run on the **GPU**.
The code explicitly sets `delegate: 'GPU'` when initializing the `FaceDetector`.

2. If it runs on GPU, does it work on every laptop?
Yes, for the vast majority of cases.

- **How it works**: The "GPU" mode in the browser uses **WebGL** (or WebGPU), which is a standard web technology supported by all major browsers (Chrome, Firefox, Edge, Safari) and works on almost all modern laptops, including those with integrated graphics (like Intel HD/UHD Graphics or Apple Silicon).
- **Privacy**: Even on the GPU, everything runs entirely **client-side**. No images are sent to a server.
- **Compatibility Edge Cases**: If a laptop is extremely old (10+ years), lacks valid graphics drivers, or has hardware acceleration disabled in the browser settings, the GPU mode might fail.

Recommendation:
If you want to guarantee 100% reliability even on problematic hardware, we could add a **fallback mechanism**: try to initialize with GPU first, and if that fails, automatically restart with CPU. Would you like me to implement this fallback logic?

User Input

i want you to implement bundling for this projetc with webpack, ask me what to bundle and what not before doing it

`*Viewed [package.json] (file:///home/mario/Desktop/Unipi/Magistrale/advanced-programming/Progetto-esame/AI-Web-Face-Detector-V2/package.json) *`

Planner Response

To switch from Vite to Webpack, we need to decide how to handle the assets.

Here is what we can bundle:

1. **Application Code**: Your JavaScript files (main.js, Camera.js, etc.) and CSS (style.css). **(Recommended to bundle)**
2. **Libraries**: The `@mediapipe/tasks-vision` library. **(Recommended to bundle)**
3. **Heavy Assets (WASM & AI Model)**:
    - Currently, the application loads the **WASM binary** and the **Face Detection Model (`.tflite`)** from a CDN (remote server) at runtime.
    - **Option A (Online-Only)**: Keep loading them from the CDN. This keeps the initial bundle small, but the app requires internet to start.
    - **Option B (Offline-Ready)**: Download these files and bundle them locally. The app will work offline, but the project size will increase by ~5-10MB.

Which option do you prefer for the Heavy Assets (Option A or B)?

User Input

include 1, 2 and 3 with option B and exclude for bundle every .pdf .typ .txt .md .png files

`*Edited relevant file*`

`*User accepted the command `npm install --save-dev webpack webpack-cli webpack-dev-server style-loader css-loader html-webpack-plugin copy-webpack-plugin clean-webpack-plugin`*`

`*User accepted the command `mkdir -p src/assets/models && curl -L https://storage.googleapis.com/mediapipe-models/face_detector/blaze_face_short_range/float16/1/blaze_face_short_ran`

ge.tflite -o src/assets/models/face_detector.tflite`*

Edited relevant file

Edited relevant file

Edited relevant file

User accepted the command `npm run build`

User accepted the command `mv webpack.config.js webpack.config.cjs`

User accepted the command `npm run build`

Edited relevant file

Edited relevant file

User accepted the command `pwd`

User accepted the command `cd dist`

User accepted the command `ls`

User accepted the command `pwd`

User Input

show me how to use the builded production version

Planner Response

To run the production version you just built in the `dist/` folder, you need to serve it with a web server. You cannot simply double-click index.html because of browser security restrictions (CORS) regarding camera access and loading WASM files.

Here are a few ways to run it:

Option 1: Python (easiest, built-in)

Since you are on Linux, you likely have Python installed.

```
cd dist
python3 -m http.server 8000
```

Then open `http://localhost:8000` in your browser.

Option 2: `serve` (Node.js)

You can use the `serve` utility directly with `npx`:

```
npx serve dist
```

Option 3: VS Code "Live Server"

If you use VS Code, right-click dist/index.html and select **"Open with Live Server"**.

I will update the **Walkthrough** artifact to include these instructions so you don't forget them.

(I'll update the artifacts now).

Edited relevant file

User Input

i tried opening index.html in browser but model isn't loading

*Viewed [Detector.js] (file:///home/mario/Desktop/Unipi/Magistrale/advanced-programming/Progetto-esame/AI-Web-Face-Detector-V2/src/core/Detector.js) *

*Viewed [webpack.config.cjs] (file:///home/mario/Desktop/Unipi/Magistrale/advanced-programming/Progetto-esame/AI-Web-Face-Detector-V2/webpack.config.cjs) *

Planner Response

The issue is that modern browsers **block** loading external resources like WASM files or performing `fetch` requests when you open a file directly (double-clicking, which uses the `file://` protocol). This is a strict security policy (CORS).

You **must** run a local web server to use the production build.

Please run this command in your terminal from the `dist` folder:

```
python3 -m http.server 8000
```

(Or if you are in the project root: `cd dist && python3 -m http.server 8000`)

Then, open your browser to:
http://localhost:8000

This will work because `http://localhost` is treated as a secure origin, whereas `file://` is not.