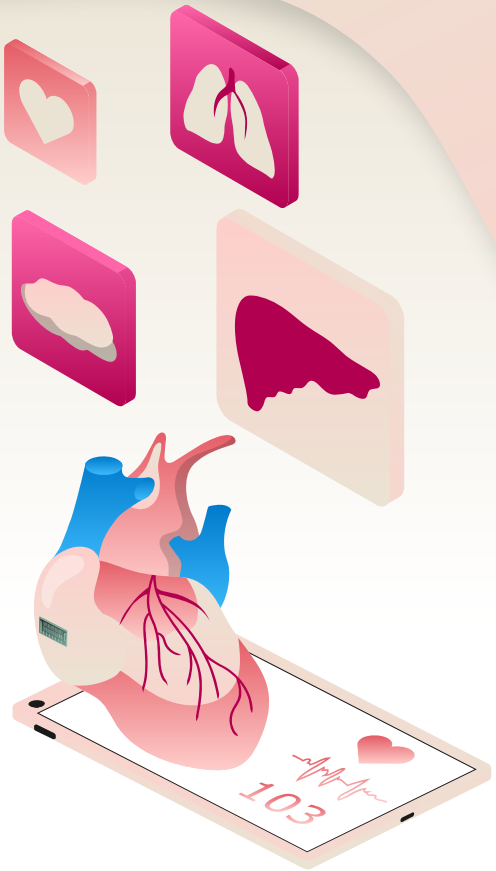
# BUILDING AN ENGAGING DEMO FOR YOUR RESEARCH

Aleksandra Kulbaka




Watch Video Presentation

IMPERIAL

# RESEARCH SHARING PIPELINE

00 Work work work

01 Results!

02 Write a paper

03 Publish code

04 Share with your network

05 Demo?

# WHY DEMO?

## Impactful

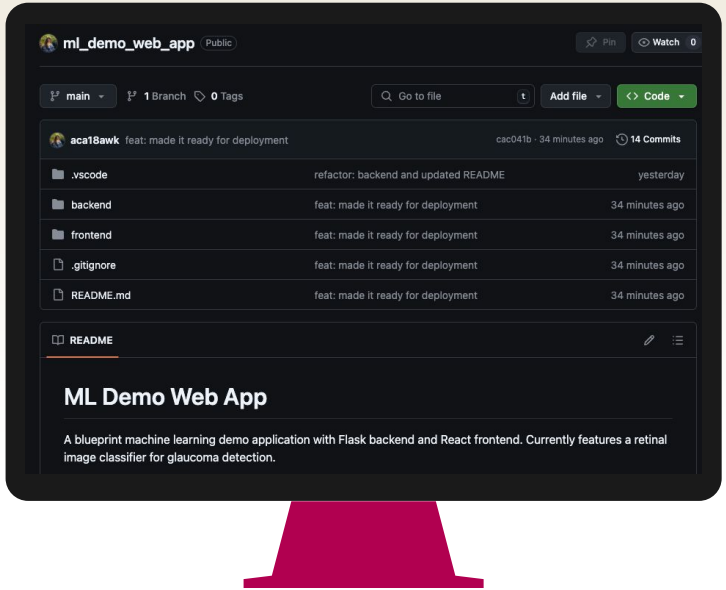
Maximise outreach to end users

## Engaging

To larger audience

## Accessible

For people with no background in ML




# CUSTOMISABLE WEB APP FOR YOUR CLASSIFIER

# WHAT DO YOU NEED

## Model .pt file

Your trained model file (e.g. saved using PyTorch as .pt)

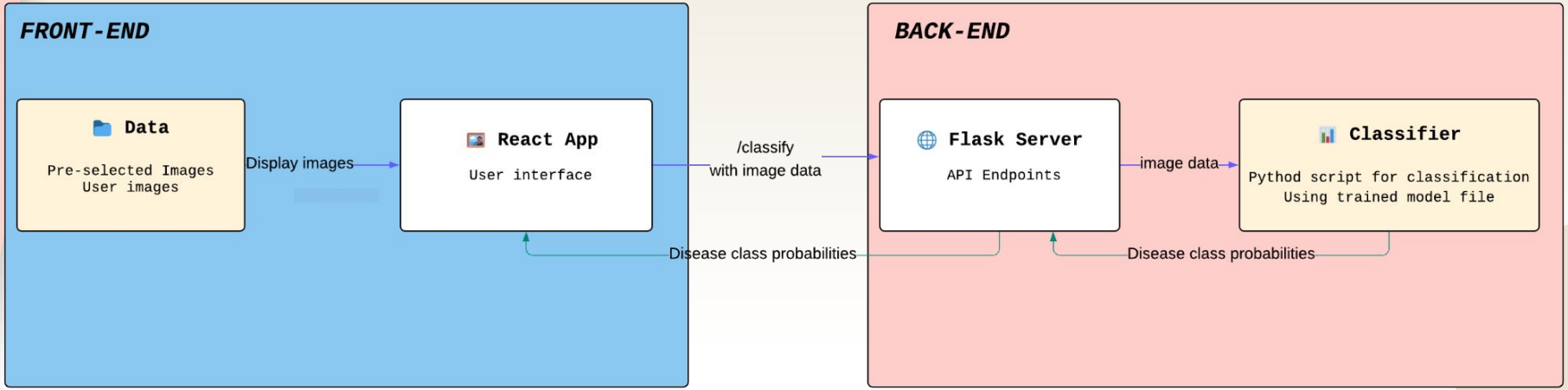
## Python classifier script

Python code to generate the results (classifier script)

## Input images

A few images from your dataset

# ARCHITECTURE OVERVIEW



# CODE WALK-THROUGH

```
ML_demo_web_app/
├── backend/
│   ├── main.py                    # Flask API endpoints
│   ├── Dockerfile                 # Docker configuration for deployment
│   ├── requirements.txt           # Python dependencies
│   └── classifier/
│       ├── get_prediction.py      # Main prediction function
│       ├── best_acc_model.pt      # Trained model weights
│       └── ...                    # Model architecture and utilities
├── frontend/
│   ├── src/
│   │   ├── constants.js           # Configuration constants (API URLs, model info)
│   │   ├── App.jsx                # Main React component
│   │   ├── components/            # Reusable UI components
│   │   └── assets/                # Sample images and static assets
│   ├── package.json               # Node.js dependencies
│   └── vite.config.js             # Vite build configuration
└── README.md
```

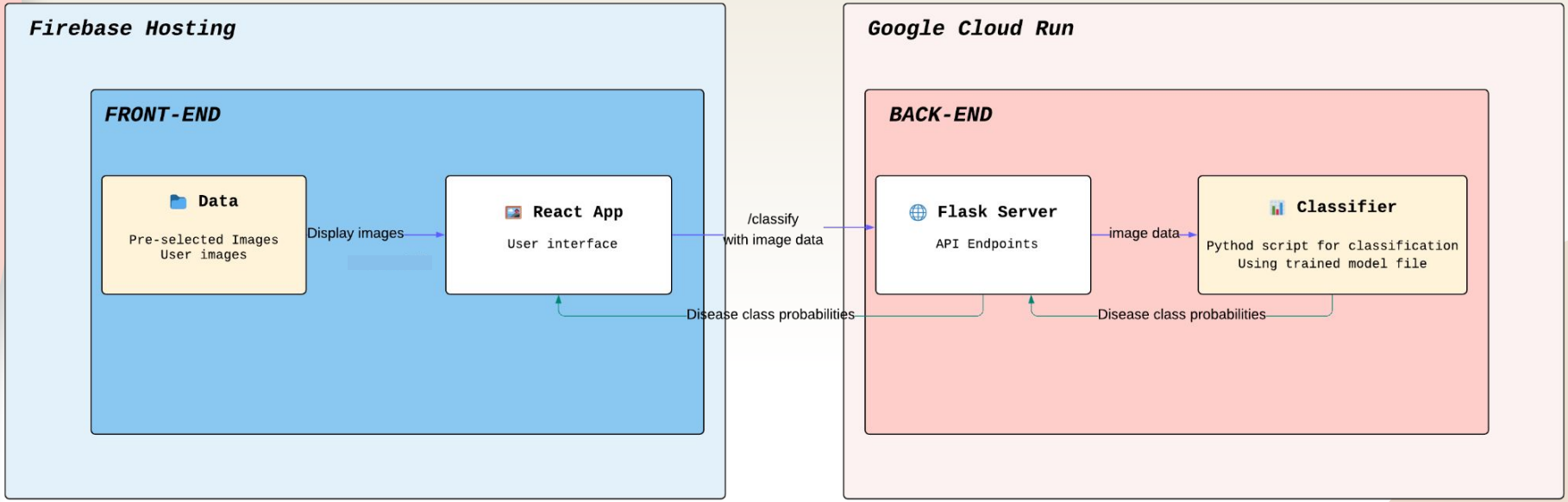
# FILES TO EDIT

```
ML_demo_web_app/
├── backend/
│   ├── main.py                 # Flask API endpoints
│   ├── Dockerfile              # Docker configuration for deployment
│   ├── requirements.txt        # Python dependencies
│   └── classifier/
│       ├── get_prediction.py   # Main prediction function
│       ├── best_acc_model.pt   # Trained model weights
│       └── ...                 # Model architecture and utilities
├── frontend/
│   ├── src/
│   │   ├── constants.js        # Configuration constants (API URLs, model info)
│   │   ├── App.jsx             # Main React component
│   │   ├── components/         # Reusable UI components
│   │   └── assets/             # Sample images and static assets
│   ├── package.json            # Node.js dependencies
│   └── vite.config.js          # Vite build configuration
└── README.md
```

# RUNNING IT LOCALLY

| | **Front-end** | **Back-end** |
|---|---|---|
| *Clone the repo* | `git clone https://github.com/aca18awk/ml_demo_web_app.git` | |
| *Create environment and install packages* | `cd frontend`<br>`npm install` | `cd backend`<br>`python -m venv backend_env`<br>`source backend_env/bin/activate`<br>`pip install -r requirements.txt` |
| *Run the code in 2 terminals* | `npm run dev` | `python main.py` |
| *See the endpoint at:* | `http://localhost:5173/` | `http://localhost:8080/` |

# DEPLOYMENT


Firebase Hosting
FRONT-END
Data
Pre-selected Images
User images
Display images
React App
User interface
/classify
with image data
Disease class probabilities
Google Cloud Run
BACK-END
Flask Server
API Endpoints
image data
Classifier
Pythod script for classification
Using trained model file
Disease class probabilities

# DEPLOYMENT

## Back-end

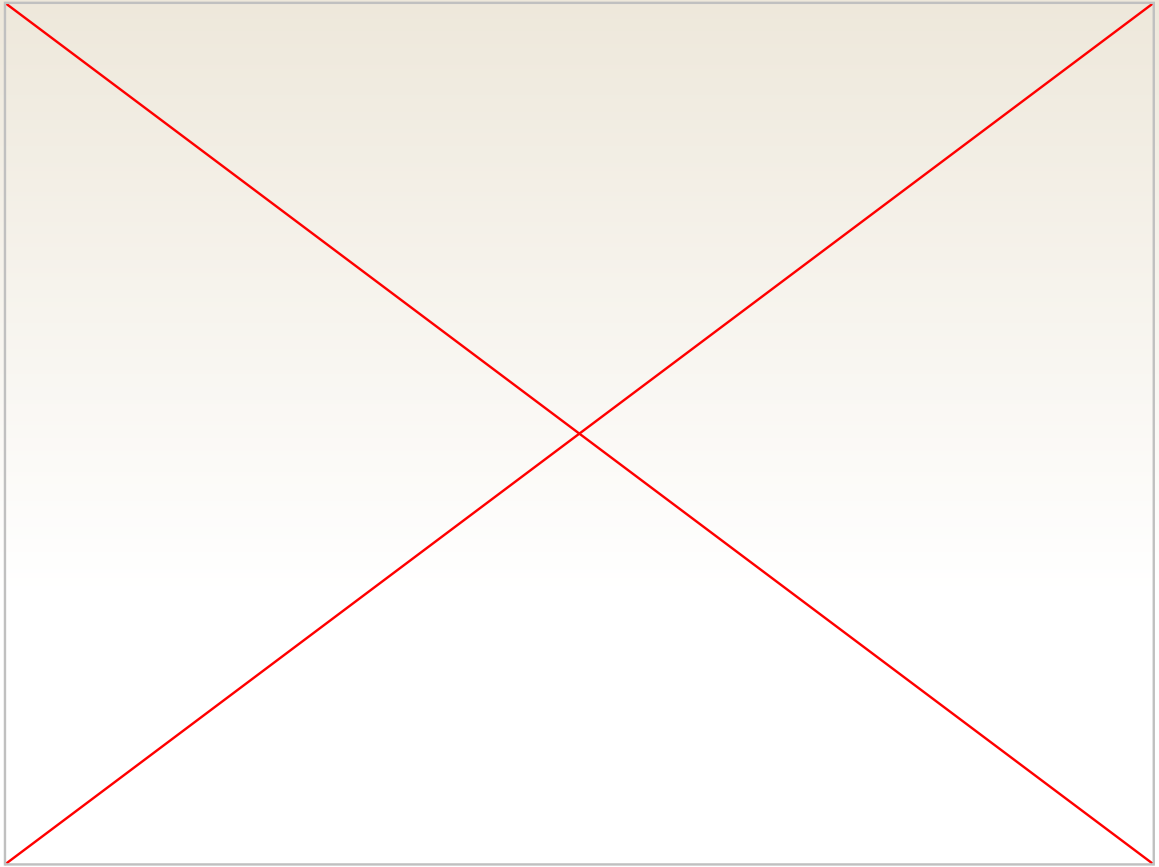
```
# Setup
gcloud init && gcloud auth login
gcloud services enable
cloudbuild.googleapis.com run.googleapis.com

# Deploy
cd backend
gcloud run deploy ml-demo-backend \
--source . \
--platform managed \
--region us-central1 \
--allow-unauthenticated \
--memory 2Gi \
--cpu 1 \
--max-instances 10
```

## Front-end

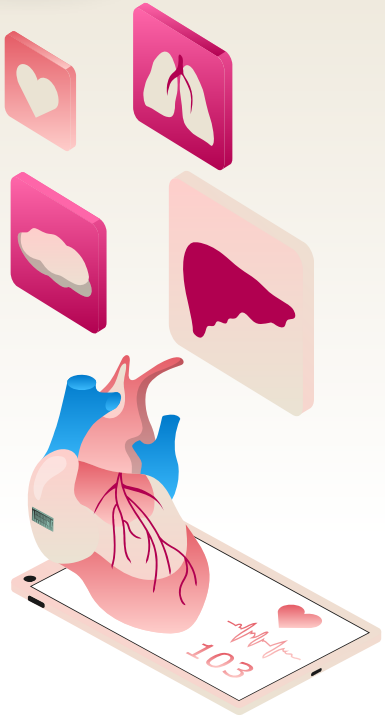
```
# Setup
npm install -g firebase-tools
firebase login

# Deploy
cd frontend
npm run build
firebase init hosting
```

# DEMO

Click to see
the demo ->



# THANK YOU!




CREDITS: This presentation template was created by **Slidesgo**, and includes icons by **Flaticon** and infographics & images by **Freepik**