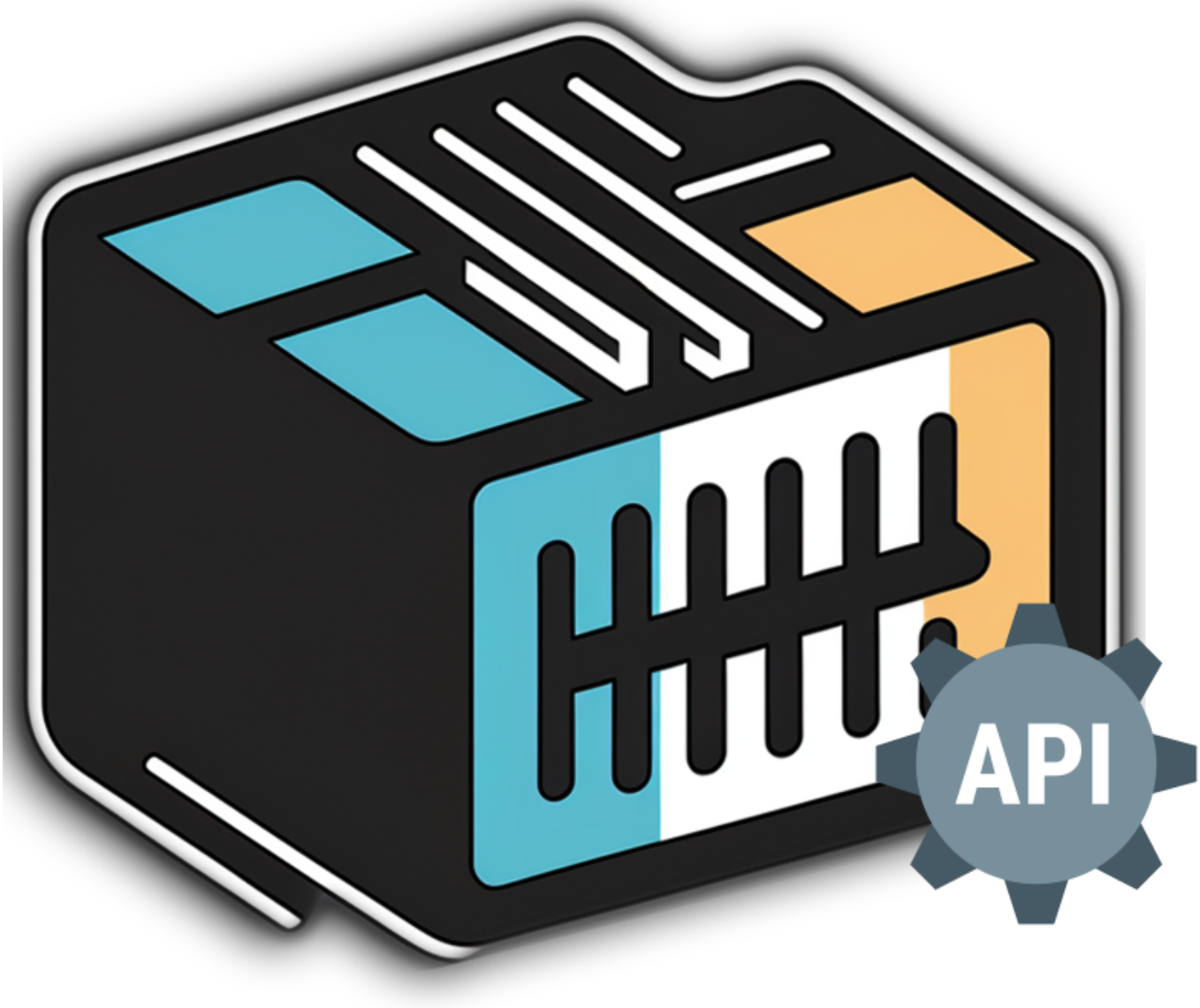
E-Jam API



RUST ACTIX UBUNTU RASPBERRY PI RESTFUL API

Welcome to the E-Jam API documentation

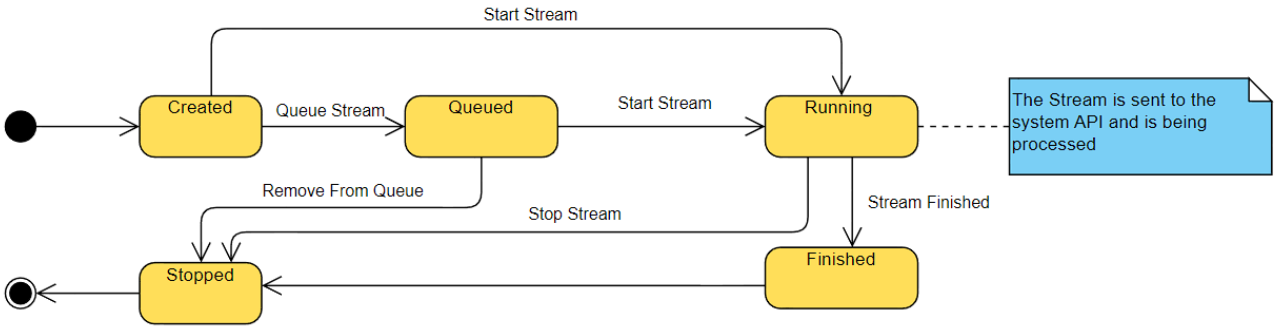
This API is used to create and manage streams. The E-Jam API is a REST API that allows you to manage the list of streams in the E-Jam application. The API is implemented using the Actix Web framework and Rust.

The API is hosted on a Raspberry Pi 4 Model B with 4GB of RAM. The Raspberry Pi is connected to a 1Gbps network. The Raspberry Pi is running Ubuntu 20.04 LTS.

The API is hosted on port 8080. The API is hosted on the IP address

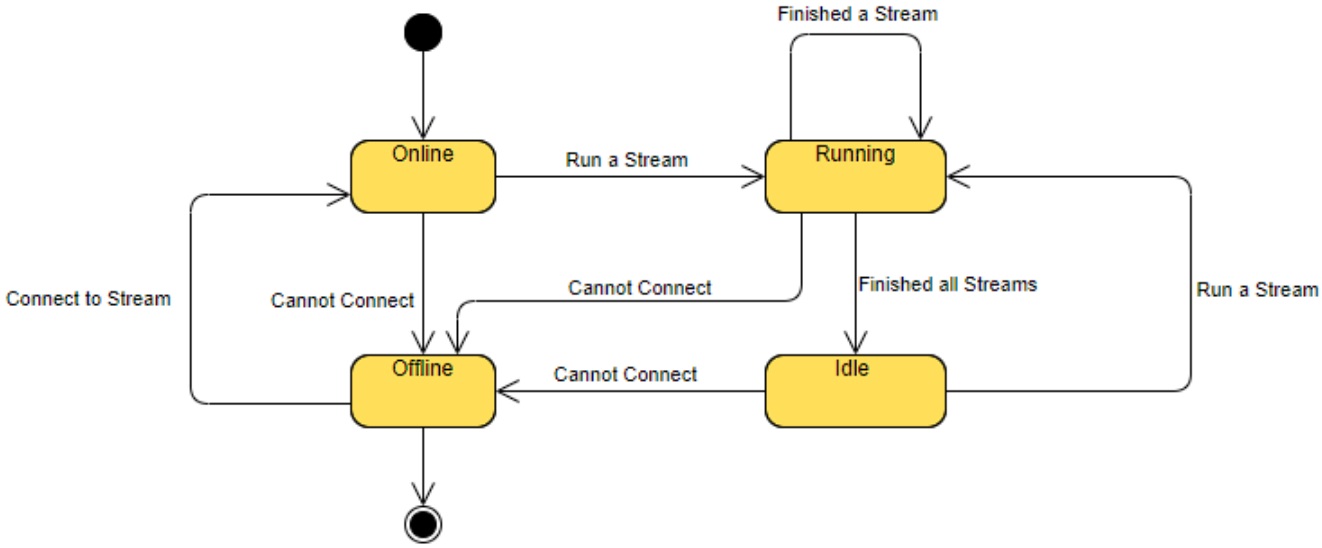
Stream State Machine

The stream state machine is as follows:



note: The stream state finished is only applied when all devices have finished sending and receiving packets.

The Device State Machine is as follows:



API Documentation

The API documentation is available at http://localhost:8080/.

Routes

GET /streams

Returns a list of all streams in the list of streams.

GET /streams/{stream_id}

Returns the stream with the given stream_id.

POST /streams

Adds a new stream to the list

DELETE /streams/{stream_id}

Deletes the stream with the given stream_id.

PUT /streams/{stream_id}

Updates the stream with the given stream_id.

POST /streams/{stream_id}/start

Starts the stream with the given stream_id.

POST /streams/{stream_id}/force_start

Forces the stream with the given stream_id to start.

POST /streams/start_all

Starts all streams in the list of streams.

POST /streams/{stream_id}/stop

Stops the stream with the given stream_id.

POST /streams/{stream_id}/force_stop

Forces the stream with the given stream_id to stop.

POST /streams/stop_all

Stops all streams in the list of streams.

GET /streams/{stream_id}/status

Returns the status of the stream with the given stream_id.

GET /streams/status

Returns the status of all streams in the list of streams.

GET /devices

Returns a list of all devices in the list of devices.

GET /devices/{device_ip}

Returns the device with the given device ip address.

POST /devices

Adds a new device to the list

DELETE /devices/{device_ip}

Deletes the device with the given device_ip.

PUT /devices/{device_ip}

Updates the device with the given device_ip.

Stream object

The structure of the Stream object as a table is as follows:

Field	Type	Required	Default	Min	Max	Validation
stream_id	String	Yes		3	3	stream_id must be 3 characters long, stream_id must be alphanumeric
stream_start_time	u64	No	0	0		stream_start_time must be greater than 0
senders_name	Vec of Strings (name or ip of device)	Yes		1		number_of_senders must be greater than 0
receivers_name	Vec of Strings (name or ip of device)	Yes		1		number_of_receivers must be greater than 0
payload_type	u8	Yes		0	2	payload_type must be 0, 1 or 2
number_of_packets	u32	Yes		0		number_of_packets must be greater than 0
payload_length	u16	Yes		0	1500	payload_length must be between 0 and 1500
seed	u32	Yes				
broadcast_frames	u32	Yes				
inter_frame_gap	u32	Yes		0		inter_frame_gap must be greater than 0
time_to_live	u32	Yes		0		time_to_live must be greater than 0
transport_layer_protocol	TransportLayerProtocol	No	TCP			transport_layer_protocol must be TCP or UDP
flow_type	FlowType	No	BtB			flow_type must be BtB or Bursts
check_content	bool	No	false			check_content must be true or false
check_content	bool	No	false			check_content must be true or false
stream_status	StreamStatus	No	0	0		

Device object

The structure of the Device object as a table is as follows:

Field	Type	Required	Default	Min	Max	Validation
device_name	String	Yes	defaulted to be either mac or ip address from	1		device_name must be greater than 0 characters long

		client side				
device_ip	String	Yes		7	15	device_ip must be between 7 and 15 characters long, device_ip must be a valid ip address
mac	String	Yes				must be a valid mac address

System API endpoints

The following endpoints are available for the system API:

Endpoint	Method	Body	Response	Description
/connect	GET	mac address of the device		Connect to the system API
/finish	POST		Stream_id	Notify the Admin-Client that the Stream has finished only when the stream is finished in the systemAPI side
/start	POST	Stream.to_string()	Success	Try to Start the Stream
/stop	POST	stream_id	Success	Stop a currently running Stream