
enhan(t)

Kepler

Apr 14, 2021

CONTENTS

1	Features	3
2	Why use enhan(t)	5
3	Indices and tables	33

enhan(t) is an open source toolkit that allows smart note-taking and makes you super productive with the existing video conferencing solutions like Zoom, MS Teams and Jitsi. It enables you to take real time notes, bookmark important moments, and capture screenshots. It also helps you annotate within the opened page using drawing, highlighting and text overlay.

As an efficient assistant, enhan(t) analytics module enables you to extract helpful insights like engagement, sentiment and questions from your online video interactions. It can be used during your sales calls, product walkthrough, interviews and online learning as well.

FEATURES

- Bookmark moments (to mark an important part of the meeting)
- Capture screenshots of entire screen or selected area.
- Take notes (to take manual notes using your voice along with when it was taken)
- Use voice commands such as “Take screenshot” or “Bookmark Moment” to do the same.
- Select area / text on the page to add it as a note.
- Download meeting data (which contains above mentioned bookmarks, screenshots and notes in .txt and .pdf file along with meeting transcript)
- Delete a note from the timeline.
- Annotate within tab
 - Draw over
 - Highlight content
 - Add text
 - Change colors of above
 - Erase/hide annotations
- Whitelist websites to enable enhan(t) on non-default sites
- Collapse / expand the toolbar as per your preference.
- Power mode
 - Engagement metrics
 - Sentiment metrics
 - Extract interrogatives (find what questions were asked during the meeting and when)
- Get host side transcription (via microphone) in the Basic mode and both host and guest side transcription (via tab audio) in the Power mode (with the companion transcription service and analysis CLI)
- View meeting data in a comprehensive dashboard

WHY USE ENHAN(T)

Taking notes during a meeting has been a historically cumbersome task. People take manual notes, take screenshots with a Win + Print Screen or a Cmd + Shift + 3 or the likes, do a recording, etc.

But can this experience be enhanced?

That is what enhan(t) is all about. By leveraging existing web based meeting solutions and providing a Chrome extension, the user is able to now take a bookmark to mark an important moment, take screenshot of the tab with a single click and still provide the flexibility of taking manual notes by hand-typing it or using their voice. All of these come along with when these actions were taken, so that at a later point of time, this data could be overlaid on top of a meeting recording to provide a lot more context to the important parts of the meeting. This is all apart from the full meeting transcript.

You can also annotate content during presentation to better convey your point across by drawing over, highlighting content and adding text in the page. You can take the screenshot to save it for future purpose.

Additionally, it provides Power users like customer support agents, financial advisors and the likes more pertinent meeting metrics like engagement and sentiment along with the questions which were asked during the meeting.

enhan(t) therefore goes beyond the traditional note taking and provides much more value by serving contextual data and meeting metrics.

Knowing the document organization will help you find and use features effectively and quickly:

- **Getting Started** is guide for starting enhant service in shortest amount of time. Start here if you're new to enhant.
- **Extension Usage** are recipes for enhant users. They guide you through the steps involved in addressing use-cases key issues.
- **Developer guides** are recipes for enhant developer, who want to contribute and extend enhant functionality.
- **Troubleshooting and FAQ** Frequently asked questions, tips and techniques.
- **API reference** contain technical reference for APIs, mostly auto generated from underlying code.

2.1 Getting started in 2 mins

To get started with enhan(t) in the time you make your cup noodles, one needs to at least have the Chrome extension set up. The easiest way to install it is via the Chrome store.

2.1.1 Installing the enhan(t) extension - Chrome

1. Navigate to Chrome Webstore [enhan\(t\) Note Taking Power Tool](#) page via your Chrome browser.
2. Click the 'Add to Chrome' button. You would be able to see the 'enhan(t) Note Taking Power Tool' extension under the 'Extensions' jigsaw puzzle icon in your toolbar.
3. Pin the extension for easy access.

2.1.2 Installing the enhan(t) add-on - Firefox

1. Navigate to Firefox add-ons store [enhan\(t\) Note Taking Power Tool](#) page via your firefox browser.
2. Click the 'Add' button. You would be able to see the enhan(t) icon in your toolbar.
3. When you open a Zoom, MS Teams or Jitsi meeting, you will find the enhan(t) toolbar on the top right of your tab.

Note: Speech Web API is disabled by default on firefox. To enable it check the guide here [Speech to text does not work in firefox?](#).

2.1.3 Using the enhan(t) extension

1. When you open a Zoom, MS Teams or Jitsi meeting, you will find the enhan(t) toolbar on the top right of your tab, with icons all grayed out and disabled.
2. Click on the enhan(t) extension icon in the toolbar to activate the extension. All the icons (except the icon indicating power mode) will turn blue.
3. To start the meeting - start capturing notes using Bookmark, Screenshot or Notes icon or click on the record icon.
4. Screenshot entire page or selected area using the toolbar under "Camera".
5. enhan(t) is voice enabled - Say "Take Screenshot" or "Bookmark Moment" to take a screenshot and bookmark important moment respectively. You can also click on the icon as mentioned in step 3.
6. Select the text on the page or an image and click on "Add Note" to add selection as a note.
7. You can click the pencil icon to open up various annotation tools. The functionality of different tools below:
 - a. Pen icon to draw over the page (with color selection)
 - b. Highlighter icon to highlight content (with color selection)
 - c. Text icon to add text anywhere in the page (with color selection)
 - d. Eraser icon to erase content drawn by pen, highlighter or text annotation tools
 - e. Eye icon to temporarily hide all annotations
 - f. Trash icon to delete all annotations
8. Collapse / Expand the toolbar as per preference.
9. Once the meeting is over, hit the animated record icon to stop enhan(t). A zip file containing the meeting data is downloaded in your default download location.

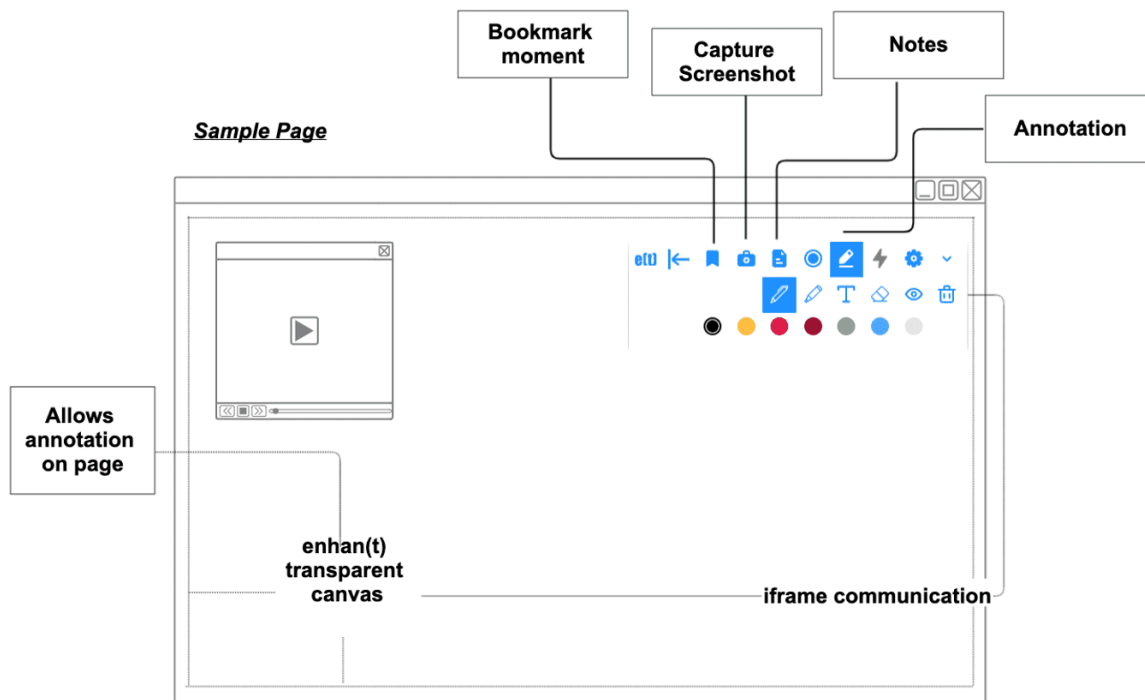
2.1.4 Viewing the meeting data

1. To view the meeting details, go to <https://keplerlab.github.io/enhant-dashboard-viewer/>
2. Click on 'Select meeting data(.zip)' button to add the meeting data zip file.
3. Optionally, add the meeting recording by clicking on 'Optional - Select meeting recording (.mp4)'.
4. Click on 'View meeting' button to see all your bookmarked moments, captured screenshots and taken notes in context.

2.2 Parts of enhan(t)

- Browser Extension (for chrome & firefox) [required]
- Transcription Service [optional]
- Analysis CLI (Command Line Interface) [optional]
- enhan(t) meeting data viewer [optional]

2.2.1 Browser Extension



Then enhan(t) extension is the minimum requirement to get started with enhan(t) in a meaningful way. It allows basic users to enhance the meetings conducted in Zoom, MS Teams or Jitsi on the Chrome & firefox browser. It enables users to bookmark moments, capture screenshots (take the screenshot of the visible tab area along with the timestamp), take notes (take manual notes by hand-typing it or using voice along with timestamp). On ending the enhan(t) session, the user is provided with a zip download, containing all the data captured. This data can be viewed as plan text after extracting the zip download or can be viewed in the enhan(t) data viewer.

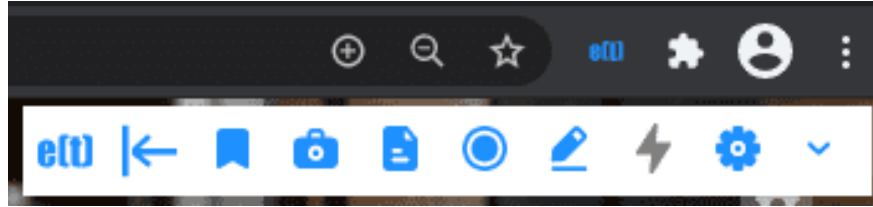


Fig. 1: enhan(t) extension - Basic mode toolbar

The annotation tools in the extension can be used during presentations to convey ones point across in a better way. It can also be used for reference by taking a screenshot. By providing an ability to draw over, highlighting content and adding text over a page, one can enrich meeting experiences and add flavor to learning experiences, just to name two use cases.

The extension can provide more data if used in Power mode alongside the companion transcription service. Once the setup is done, Docker application run and the Power mode is enabled in the extension settings, the extension will now be able to transcribe both the host side (via microphone) and guest side (via tab audio) of the conversation. Additionally, meeting metrics like engagement and sentiment is also provided. Post the call, all the questions asked during the meeting can be extracted via interrogative analysis.

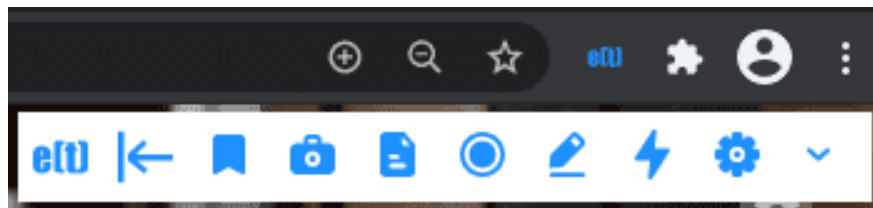


Fig. 2: enhan(t) extension - Power mode toolbar

Note : Power mode is unavailable on firefox because of lack of support for capturing tab audio.

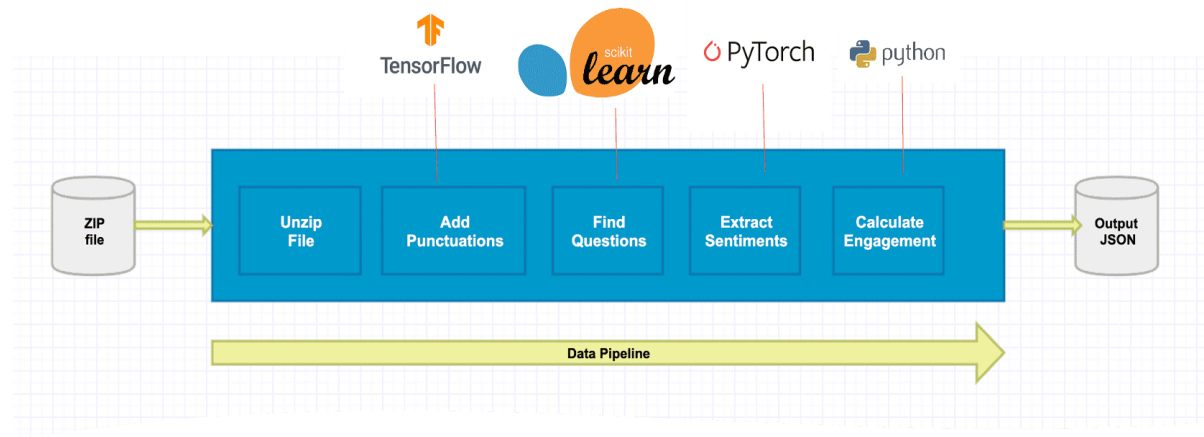
2.2.2 Transcription Service

The transcription service enables the speech to text conversion for the guest side conversations in the Power mode of the Chrome extension.

The transcription service uses an open source speech recognition toolkit called Vosk (<https://alphacephei.com/vosk/>) for speech to text conversion by default.. One drawback of using kaldi-vosk speech recognition is that transcribed text do not have any punctuations. To add punctuations to the text we use [fastpunct](#) library for adding punctuations to the text. You can also use Google Cloud Speech to Text for transcription service as an alternative. For this please read the guide *Installing the Transcription Service and Analysis CLI (Command Line Interface) with Google cloud*.

2.2.3 Analysis CLI (Command Line Interface)

Architecture NLP analyzer



The analysis CLI enables the generation of meeting metrics like engagement and sentiment in the Power mode. The zip file generated in the power mode can be provided to the analysis CLI service to generate an output zip file which would have engagement and sentiment metrics. If you want to read more about current NLP analyzers available in enhan CLI, you can read more [Current NLP Analyzers](#).

2.2.4 enhan(t) Data Viewer

The enhan(t) Data Viewer where a user can view a downloaded meeting data zip file in context.

Once a basic mode data zip file is loaded locally, the user can view the following details along with the time:

- Meeting duration
- Bookmarks
- Screenshots : Full page or selected area.
- Notes: Using Voice or manually.
- Audio or video recording overlayed with bookmarks, screenshots and notes moments (if user uploads them)

If a power mode meeting data zip file is loaded, along with the basic mode details, one can also view:

- Average engagement
- Average sentiment
- Sentiment and engagement overlay graph on the audio or video recording
- Sentiment outliers
- Extracted interrogatives

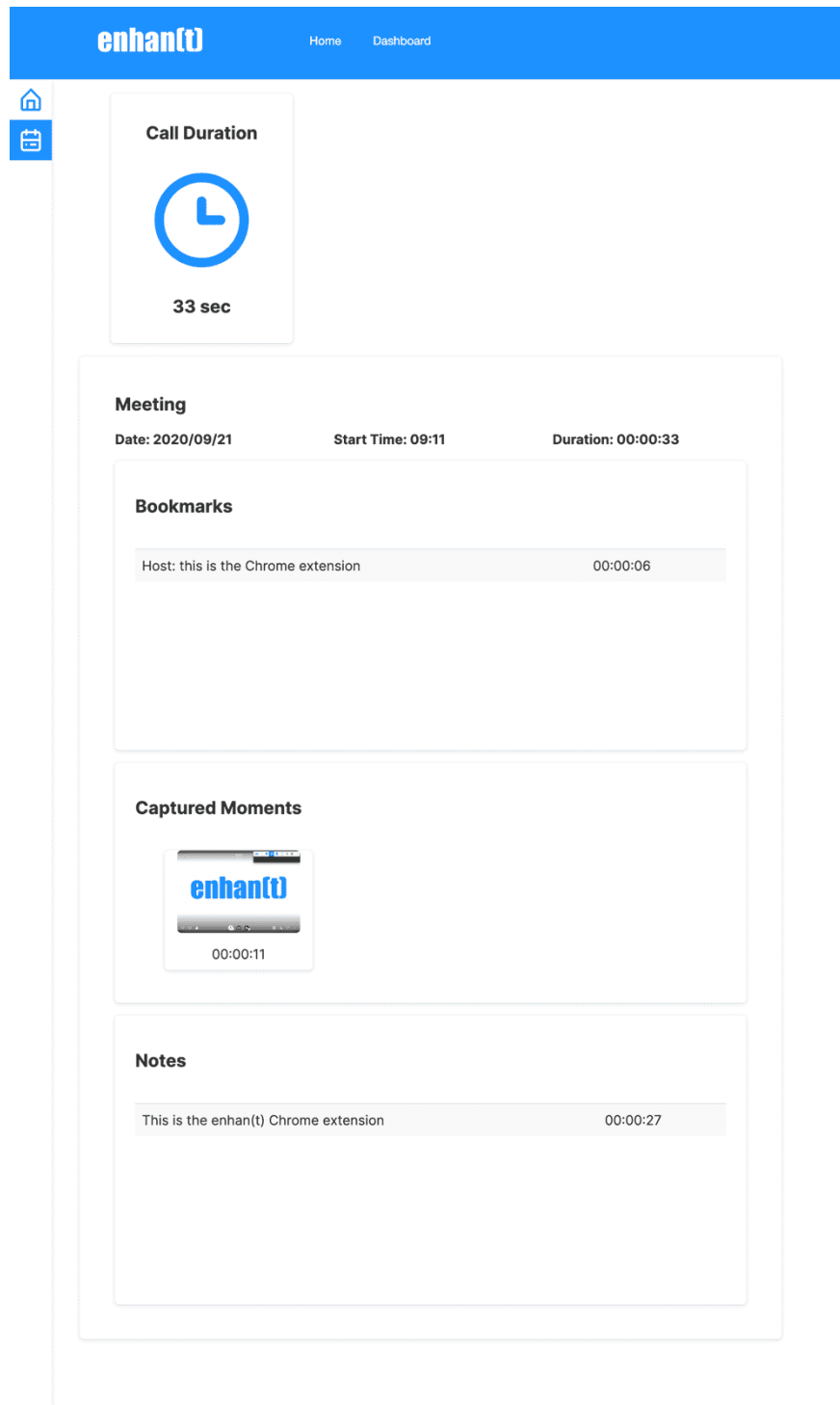


Fig. 3: enhan(t) Data Viewer - Basic mode

2.3 Installing the Chrome Extension

The Chrome extension is the minimal requirement for most users to leverage enhan(t). The easiest way to install it is via the Chrome store.

To install the enhan(t) extension:

1. Navigate to Chrome Webstore [enhan\(t\) Note Taking Power Tool](#) page via your Chrome browser.
2. Click the 'Add to Chrome' button. You would be able to see the enhan(t) icon in your toolbar.
3. When you open a Zoom, MS Teams or Jitsi meeting, you will find the enhan(t) toolbar on the top right of your tab.

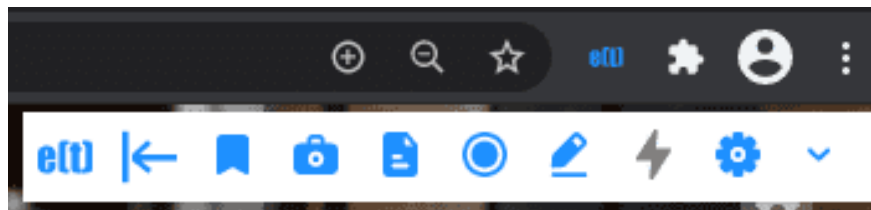


Fig. 4: enhan(t) Chrome extension toolbar on the top right of your tab

2.4 Installing the Firefox Add-on

The firefox add-on is the minimal requirement for most users to leverage enhan(t). The easiest way to install it is via the firefox add-on store.

To install the enhan(t) extension:

1. Navigate to Firefox add-ons store [enhan\(t\) Note Taking Power Tool](#) page via your firefox browser.
2. Click the 'Add' button. You would be able to see the enhan(t) icon in your toolbar.
3. When you open a Zoom, MS Teams or Jitsi meeting, you will find the enhan(t) toolbar on the top right of your tab.

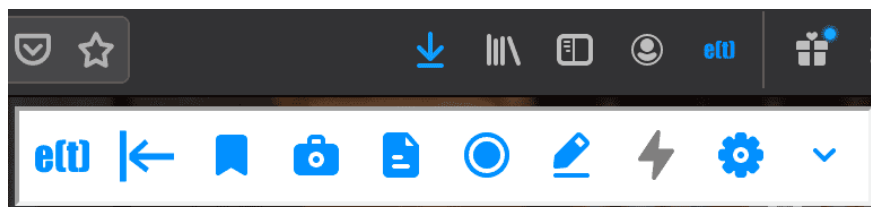


Fig. 5: enhan(t) add-on toolbar on the top right of your tab

2.5 Installing the Transcription Service and Analysis CLI (Command Line Interface)

The transcription service and analysis CLI enables the Power mode in the Chrome extension to work. It enables guest side transcription, provides engagement, sentiment and interrogatives.

Note : Power mode is unavailable on firefox because of lack of support for capturing tab audio.

The transcription service and the analysis CLI can be installed as a Docker compose application. To install both as application using docker application follow these instructions:

1. Make sure you have Docker installed on your system. If not, go to <https://docs.docker.com/get-docker/> and follow the instructions there to get started with Docker.
2. Make sure you have Git installed. If not, go to <https://www.atlassian.com/git/tutorials/install-git> and follow the instructions there.
3. You can find the enhan(t) project on Github at <https://github.com/keplerlab/enhant>. Clone the repository by running the following git clone command on your terminal:

```
git clone https://github.com/keplerlab/enhant.git
```

On **Windows** to prevent line ending issues clone using this command instead:

```
git clone https://github.com/keplerlab/enhant.git --config core.autocrlf=false
```

4. Next, to have locally trusted development certificates we need to install mkcert. Please follow the installation instructions [How to add certificate for localhost for cli utility](#) for detailed instructions
5. Now go to the cloned 'enhant' directory and then run go to certificates directory.:

```
$ cd certificates-and-credentials
```

6. Run following to create the certificates.:

```
$ mkcert -key-file key.pem -cert-file cert.pem localhost 127.0.0.1 ::1
```

7. Change your directory to your cloned repo.
8. Start Docker containers:

```
cd /path/to/enhant-repo/  
docker-compose up
```

9. To Stop Docker containers, Open terminal and run the following commands:

```
cd <path-to-repo> //you need to be in your repo folder  
docker-compose down
```

You can also use Google Cloud Speech for Text for transcription service as an alternative. In most of the cases this results in better transcription accuracy. For this please read the guide [Installing the Transcription Service and Analysis CLI \(Command Line Interface\) with Google cloud](#).

2.6 Supported Hardware and operating system

enhan software is supported on the following host operating systems:

- Linux
- mac OS X
- Windows

Minimum Docker configuration.:

- Processor: 2 cpu cores
- RAM: 4GB of system memory
- Hard disk space: 20 GB
- Google Chrome or chromium browser
- Firefox browser version ≥ 80

Recommended system configuration:

- Processor: 4 cpu cores
- RAM: 6GB of system memory
- Hard disk space: 30 GB
- Google Chrome or chromium browser
- Firefox browser version ≥ 80

2.7 Extension Usage

2.7.1 Basic Mode

This is the minimal mode which can run just by installing the extension. For installation instructions, see [Installing the Chrome Extension](#) or [Installing the Firefox Add-on](#).

Once the enhan(t) extension is installed, and you open a Zoom, MS Teams or Jitsi web meeting on the browser, the enhan(t) toolbar would show up.

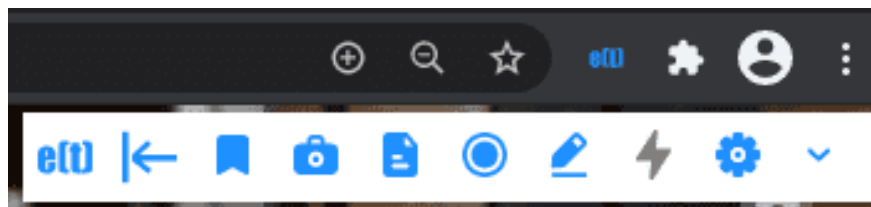


Fig. 6: enhan(t) extension - Basic mode toolbar

If the extension has just been installed for the first time, it would also ask for microphone permission, which needs to be allowed so that the meeting can be transcribed.

For Basic mode users, you can leave the 'Enable Power Mode' settings unchecked.

If you want to change the transcription language setting for the host side (via microphone), you can select English (US) or English (IN) from the drop down. Hit 'Apply' to save the changes.

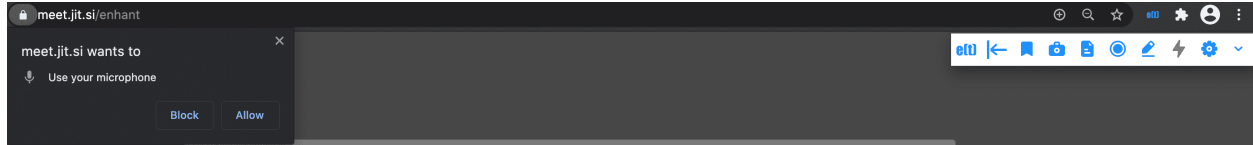


Fig. 7: enhan(t) extension - Permissions

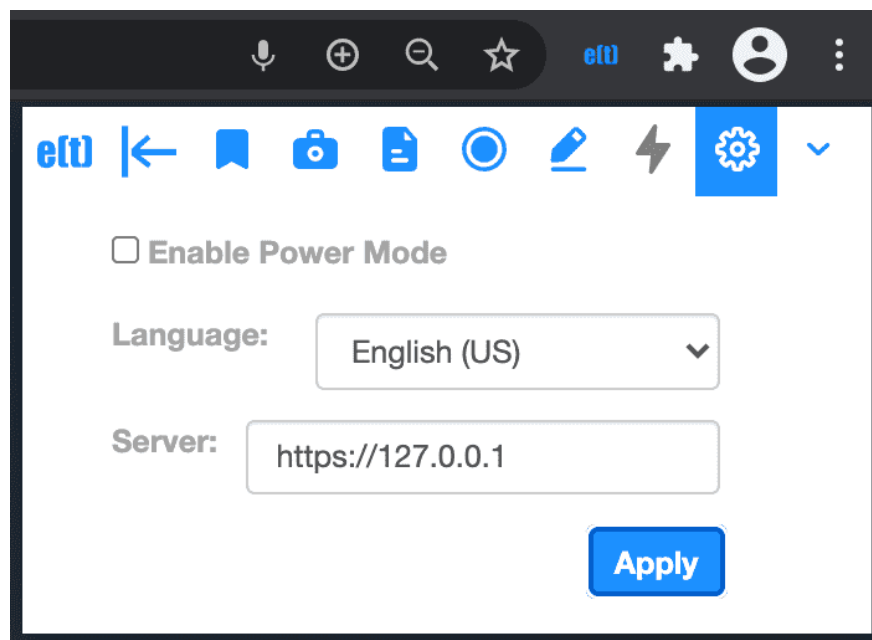


Fig. 8: Basic mode setting - 'Enable Power Mode' unchecked

Once the meeting has been joined, the record icon can be hit to start the enhan(t) session. The record icon will keep animating during the course of the session. The toolbar will expand to show 3 new icons – bookmark, camera and file. These icons provide the following functions:

1. Bookmark moment (bookmark icon): This allows the user to literally bookmark a moment during the meeting. A lot of times, one wants to mark a moment which seems important so that one can go back later and refer to it. This enables this feature.
2. Capture screenshot (camera icon): This allows the user to capture the contents displayed in the tab or selected area along with the timestamp. Like that important slide during a presentation or some subsection within the slide, along with the time.
3. Take notes (file icon): On clicking this, the user is provided with a text area where one can take notes. The user can also record notes using voice by clicking on the mic icon (speak now) in the notes section. On clicking the 'Add Notes' button, the notes are persisted along with the timestamp. Users at times want to add their thoughts during the meeting. This feature enables that.

At any point in time, user can click the rightmost chevron icon to toggle the expansion and collapse the history of actions during the meeting. The latest action appears at the top.

The user can end the enhan(t) session by hitting the animated record icon. Once that is done, a zip file is generated which would contain all the captured meeting data. This can be unzipped and then the user can view the plain text data and images. Alternatively, one can go to the [enhan\(t\) Meeting Data Viewer](#) and view the details of the meeting there.

For firefox - the downloaded file is of the format a5sZQRsx.zip.part. This is due to a bug in firefox 367231 . Remove the .part from the downloaded file and use the zip file as it is to visualize data.

2.7.2 Power Mode

Note : Power mode is unavailable on firefox because of lack of support for capturing tab audio.

This mode of the extension gives additional functionalities to power users when used in conjunction with the companion transcription service. For installation of the transcription service, please refer to *Installing the Transcription Service and Analysis CLI (Command Line Interface)*.

Once the transcription service is installed and run, one can click the settings icon in the extension toolbar, check 'Enable Power Mode' and hit 'Apply' to activate the Power mode.

A lightning icon will now appear in the toolbar, signifying that the Power mode is active.

If you want to change the transcription language setting for the host side (via microphone) and guest side (via tab audio), you can select English (US) or English (IN) from the drop down. Hit 'Apply' to save the changes. English (IN) would be used only if the Google Cloud Speech to Text transcription service is run in the background.

For advanced developers, who want to change ports during local deployment or host the Docker application remotely, the 'Server' textbox can be used to input the URL. Hit 'Apply' to persist the changes.

All other features work as is in the Basic mode. The difference is that now both host (via microphone) and guest (via tab audio) side of the conversation are transcribed in real time.

The meeting zip file generated in Power mode can be further analyzed by using the Analysis CLI application and viewed at the [enhan\(t\) Meeting Data Viewer](#) for further details.

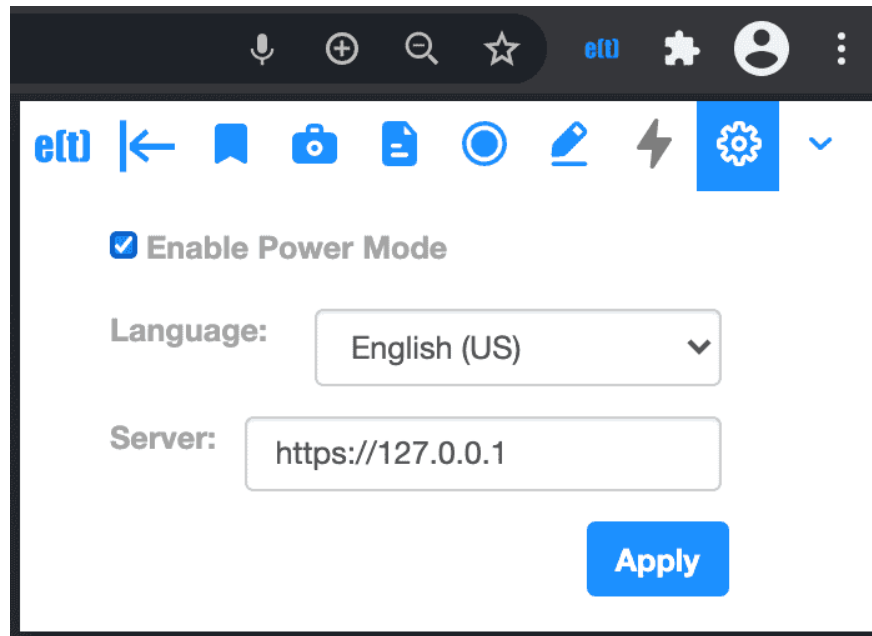


Fig. 9: Power mode setting - 'Enable Power Mode' checked

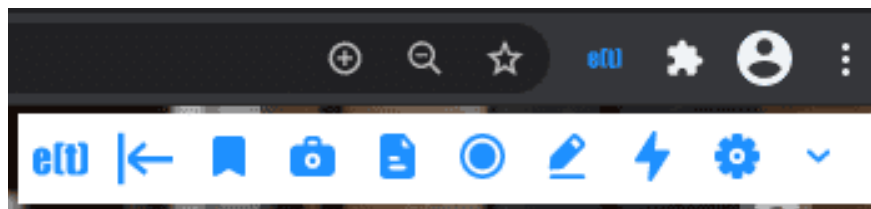


Fig. 10: Power mode icon active in the toolbar

2.8 Analysis CLI (Command Line Interface) Usage

The Analysis CLI helps to generate meeting metrics like engagement and sentiment in the Power mode. It also extracts the questions asked during the meeting. The zip file generated in the Power mode can be provided to the analysis CLI service to generate an output zip file which would have engagement and sentiment metrics.

To use the Analysis CLI:

1. Go to the cloned repo location and start the CLI service by running

```
$ cd /path/to/enhant-repo/
$ docker-compose run cli
```

2. Copy the zip file generated in Power mode to be analyzed to the meeting-data folder in the cloned repo location.

3. To process the Power mode generated zip file, run

```
$ python enhant_cli_app.py analyze meeting-data/<meeting-data-input-zip-file>.zip
```

4. Depending on the length of the meeting, the processing can take time.
5. Once processing is over, a zip file with the name <meeting-data-input-zip-file>_result.zip is generated.
6. The processed zip file can be viewed at the [enhan\(t\) Meeting Data Viewer](#) for further details.

2.9 enhan(t) Developer Guides

2.9.1 How to add certificate for localhost for cli utility

Install mkcert utility

macOS

On macOS, use [Homebrew](<https://brew.sh/>):

```
brew install mkcert
brew install nss # if you use Firefox
```

or [MacPorts](<https://www.macports.org/>):

```
sudo port selfupdate
sudo port install mkcert
sudo port install nss # if you use Firefox
```

Linux

On Linux, first install *certutil*:

```
sudo apt install libnss3-tools
-or-
sudo yum install nss-tools
-or-
sudo pacman -S nss
-or-
sudo zypper install mozilla-nss-tools
```

Windows

On Windows, use [Chocolatey](https://chocolatey.org):

```
choco install mkcert
```

or use Scoop:

```
scoop bucket add extras
scoop install mkcert
```

or build from source (requires Go 1.10+), or use [the pre-built binaries](<https://github.com/FiloSottile/mkcert/releases>).

If you're running into permission problems try running *mkcert* as an Administrator.

Install local certificate authority using mkcert

```
$ mkcert -install
Created a new local CA at "/Users/filippo/Library/Application Support/mkcert"
The local CA is now installed in the system trust store!
The local CA is now installed in the Firefox trust store (requires browser restart)!
```

Create new certificate using mkcert

```
$ cd certificates-and-credentials
$ mkcert -key-file key.pem -cert-file cert.pem localhost 127.0.0.1 ::1
```

You should see following output with last command:

```
Using the local CA at "/****/*****/**/****/mkcert"
Created a new certificate valid for the following names
- "localhost"
- "127.0.0.1"
- "::*1"

The certificate is at "cert.pem" and the key at "key.pem"
```

2.9.2 Install and use enhant from source (without docker)

Currently we have tested enhant installation on MacOSX and Linux systems only. These instructions assumes you have anaconda python with python version 3.7 already installed on your system.

- 1) Clone enhant repo.

2.9.3 How to use custom transcription service

Due to unavailability of capability of current browsers to do guest side/or screen side audio transcription we cannot do guest side transcription in browser, hence currently host side transcription for enhant plugin is done by command line docker application , for this enhant plugin sends captures screen audio using tabcapture api in chrome, this audio is sent into with following format:

```
type: LINEAR16
number of channels: 1
sample rate: 44100
sample width: 2
```

additionally transcription service is hosted as an websocket server with **https** enabled on port number **1111**

To communicate some of the configuration parameters are communicated between plugin and command line application. Format of data sent by Plugin to application on first connect:

```
{
  "cmd": "start",
  "origin": "speaker",
  "conversation_id": "ID",
  "lang": "en-US"
}
```

On successfully connect application sends following json data back:

```
{
  "cmd": "start",
  "origin": "speaker",
  "conversation_id": "ID",
  "lang_enabled": "en-US",
  "need_punctuation": True
}
```

After this initial handshake browser plugin needs to send continuous packets of raw audio data in format discussed earlier. In return transcription service sends back text transcription data as and when available.

To implement your own transcription service you need to follow the same protocol so that plugin remains interface agnostic to actual transcription service.

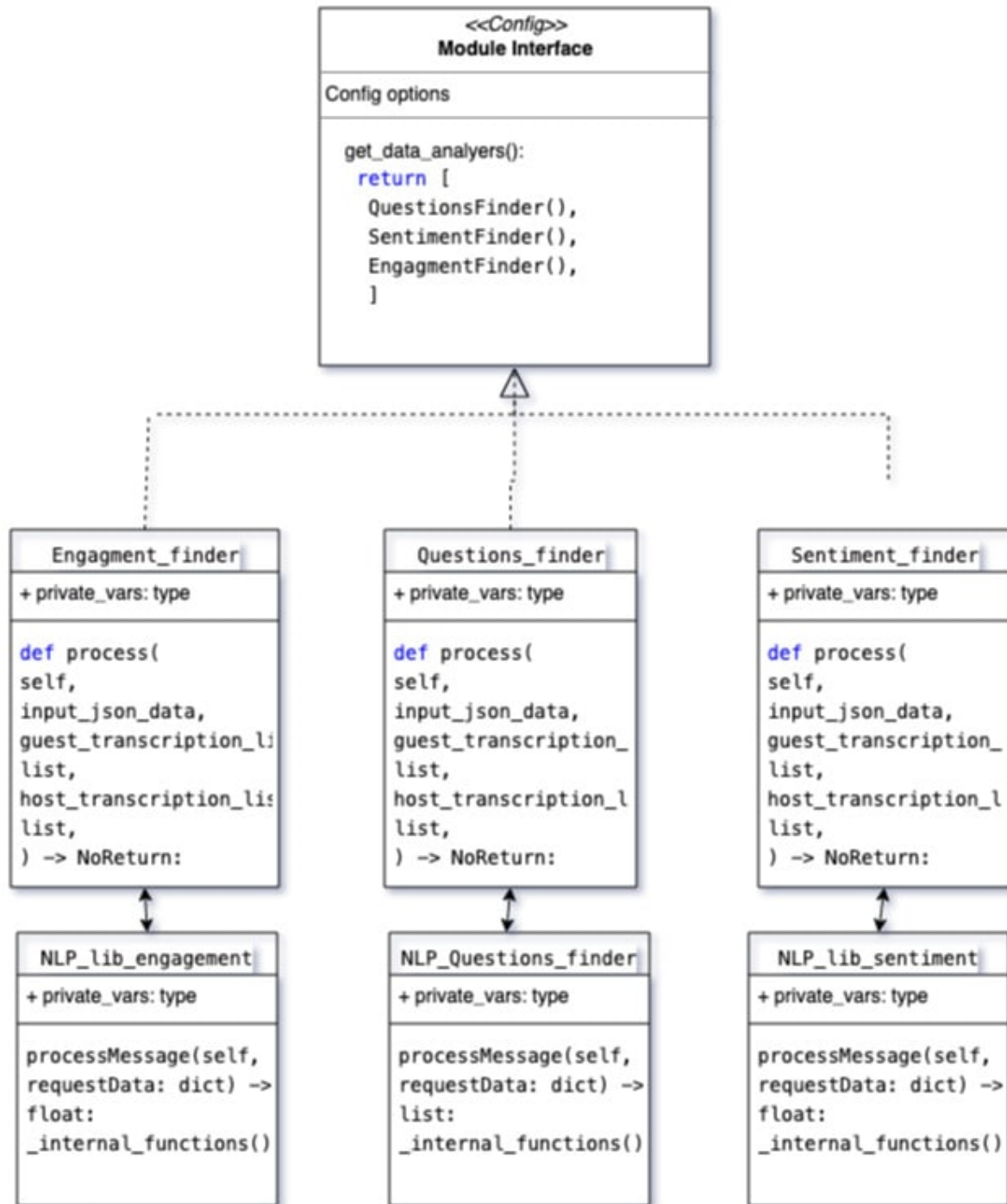
If for example you want to add Mozilla deepspeech as an transcription service you need to first find a streaming api for deepspeech which can take audio chunks as an input and gives back transcription back as text messages, now you can modify any of the two python websocket servers, located **api/transcription_svc/websocket-server/asr_server.py** or **api/transcription_svc/websocket-server/python_websocket.py** you client can than receive streaming audio data from enhant plugin on websocket as mentioned above and then pass on this audio steam through python multiprocessing module to separate os process which can in turn run and call Mozilla deepspeech api for speech to text. This way you can support for your own speech to text engine with enhant transcription service.

2.9.4 NLP analyzers for enhan(t) analytics module

Introduction enhan(t) NLP analyzers

As discussed in overview section enhan(t) analytics module enables you to extract helpful insights like engagement, sentiment and questions from your online video interactions. All of these tasks are accomplished by help of NLP analyzers on top of speech transcripts of video. Enhant analytics module is built as a service where a developer can add or remove any of these analyzers easily. To understand how we can add and remove analyzers let's first look at Architecture of NLP analyzers:

Architecture NLP analyzer



Take a look at above figure NLP Analyser Architecture, All of analytics is processed by our included cli application, cli application configures which analyzers to run on included transcriptions data using config module, in config module you can their is function you can `get_data_analyzers()` where you can initialize your own custom nlp analyzer or remove any of the existing nlp analyzer.

Each of the NLP analyzers needs to follow this common input output syntax to help with modularity, hence each of them take `input.json` with host and guest side transcriptions in .srt format as input and results are written back in `processed.json` file.

Current NLP Analyzers

Currently insights like engagement, sentiment and questions are extracted from your online video interactions. Each of these features are provided by their respective modules in `nlp_lib`. We will briefly go through each one of them one by one.

Engagement Detection This module compares calculates engagement score for any conversation, Engagement score is a measure of how much is the guest or audience of your call is participating in call, So for any one sided call in which only host is speaking, engagement score is low, but in any call where host and guest both sides are equally speaking, engagement is high, to calculate engagement score, transcription data of both sides is used, for any given time window, engagement score is calculated by a heuristic of ratio of host vs guest side transcription. This analyzer is implemented with the help of `nlk library` <https://www.nltk.org/> and custom python implementation.

Question Detection This module is used for finding out what are the questions being asked in a given meeting, this is done by training a machine learning model (random forest) trained on `nps_chat` data which can classify whether a sentence is a question or not. This analyzer is implemented with the help of `nlk library`, `scikit-learn` and custom python implementation.

Sentiment Detection This module calculates sentiment score of each of the spoken sentence in meeting, In dashboard we can then show sentences which are outliers, means which have highly positive as well as negative sentiment scores. This analyzer is implemented with the help of pre-trained sentiment model from `flair library` and `nlk library`.

Adding your own nlp analyser

As shown in `nlp Architecture` diagram all of the NLP analyzers need to follow common interface which so if you need to add a additional nlp analyzers you need to implement same interface standard public interface. Which is

```
def process(
    self,
    input_json_data,
    guest_transcription_list: list,
    host_transcription_list: list,
) -> NoReturn:
```

This process function can call any of your own pre-processing logic, you can also create a new module implemented inside `nlp_lib`.

After implementing your own nlp analyser you can import it's module inside `cli/config.py` file and add instance of your module class inside `get_data_analyzers()` function. e.g. currently in `config.py` `get_data_analyzers()` function looks like this:

```
def get_data_analyzers() -> list:
    """
    Returns the list of analyzers.
    """
    # Add or remove analyzers here. All the analyzers will update the conversation_
    ↪ JSON
    return [
        QuestionsFinder(),
        SentimentFinder(),
        EngagmentFinder(),
    ]
```

So if you want to add your custome nlp analyser than you can make a class for them with input output formatting similar to current nlp analyzers and then initialize object of the class in `get_data_analyers` function. e.g. if you analyzer is called `my_custom_nlp_finder()` than your new `get_data_analyzers` function will look like this:

```
def get_data_analyzers() -> list:
    """
    Returns the list of analyzers.
    """
    # Add or remove analyzers here. All the analyzers will update the conversation_
    ↪JSON
    return [
        QuestionsFinder(),
        SentimentFinder(),
        EngagmentFinder(),
        my_custom_nlp_finder(),
    ]
```

Remove nlp analyser

Similarly if you need to remove any of the existing nlp analyzers you just need to remove their respective instance from get_data_analyzers function in cli/config file so if you want to remove let's say engagement detection than you can delete it's import statement

```
from engagment_finder import EngagmentFinder
```

and then can delete initialization code from get_data_analyzers function, so new get_data_analyzers will look like this:

```
def get_data_analyzers() -> list:
    """
    Returns the list of analyzers.
    """
    # Add or remove analyzers here. All the analyzers will update the conversation_
    ↪JSON
    return [
        QuestionsFinder(),
        SentimentFinder(),
    ]
```

2.9.5 How to build documentation from source

Build documentation using docker

1. open cli:

```
cd /path/to/enahant-repo
docker-compose run cli
```

2. Goto docs folder:

```
cd ../docs
```

3. Issue build command:

```
make html
```

Please note Documentation will be available outside container at path **enhan-repo/docs/build/html** folder

2.9.6 Installing the Transcription Service and Analysis CLI (Command Line Interface) with Google cloud

The transcription service and analysis CLI enables the Power mode in the Chrome extension to work. It enables guest side transcription, provides engagement, sentiment and interrogatives.

Note : Power mode is unavailable on firefox because of lack of support for capturing tab audio.

The transcription service and the analysis CLI can be installed as a Docker compose application. To install both as application using docker application follow these instructions:

1. Make sure you have Docker installed on your system. If not, go to <https://docs.docker.com/get-docker/> and follow the instructions there to get started with Docker.
2. Make sure you have Git installed. If not, go to <https://www.atlassian.com/git/tutorials/install-git> and follow the instructions there.
3. You can find the enhan(t) project on Github at <https://github.com/keplerlab/enhant>. Clone the repository by running the following git clone command on your terminal:

```
git clone https://github.com/keplerlab/enhant.git
```

On **Windows** to prevent line ending issues clone using this command instead:

```
git clone https://github.com/keplerlab/enhant.git --config core.autocrlf=false
```

4. Next, to have locally trusted development certificates we need to install mkcert. Please follow the installation instructions [How to add certificate for localhost for cli utility](#) for detailed instructions
5. Now go to the cloned 'enhant' directory and then run go to certificates directory.:

```
$ cd certificates-and-credentials
```

6. Run following to create the certificates.:

```
$ mkcert -key-file key.pem -cert-file cert.pem localhost 127.0.0.1 ::1
```

7. Change your directory to your cloned repo.
8. To use Google cloud for speech to text, you need to register for google speech to text API, download the google cloud service credentials file on your system and copy it into folder **<repo_folder>/certificates-and-credentials** folder additionally rename file as *google_credential_file.json*.
9. Start Docker containers:

```
cd /path/to/enhant-repo/  
docker-compose -f docker-compose-google-cloud.yml up
```

10. To Stop Docker containers, Open terminal and run the following commands:

```
cd <path-to-repo> //you need to be in your repo folder  
docker-compose -f docker-compose-google-cloud.yml down
```

2.10 Tutorials

2.10.1 Using enhan(t) in Meetings

enhan(t) can be used to collect notes while you are in a meeting.

When in basic mode, enhan(t) will collect all data in a .zip file (downloaded when the recording is stopped). The zip file contains the following:

1. **host.srt** : A subtitle file containing the transcription data of the host.
2. **images** : Folder containing all the images captured using enhan(t).
3. **notes.pdf** : The contains all the data captured via enhan(t) and in the order it was captured w.r.t time. The pdf file can be shared with others as part of meeting notes.
4. **notes.txt** : This is a raw text file containing the meeting data. It contains relative path to the image inside the images folder. The txt file is plain raw data for developers to consume it in other applications based on their need
5. **notes.md** : Markdown is a standard in lot of note-taking applications and platforms. Users can visualize the notes in either Visual Studio Code or a note-taking app such as Evernote / Joplin. These platforms support Markdown import so you can collect notes via enhan(t) and then import it inside a note-taking app.

When in power mode, enhan(t) will also transcribe the audio data for guest (speakers on the other side of the call). This transcription is available in a separate **guest.srt** file inside the zip. When run in conjunction with the analysis CLI, users will also get engagement and sentiment scores, as well as, the outliers (both negative and positive) during the meeting.

For more details on using the power mode, refer the guide [Power Mode](#).

Lets look at some scenarios in which the data from enhan(t) can be consumed:

1. Use enhan(t) visualizer: enhan(t) comes with out-of-the box visualizer, details here [Viewing the meeting data](#). This is a web based visualization tool that works only with enhan(t) zip files. The zip data downloaded using the enhan(t) extension can be uploaded here, with an optional meeting recording.

Visualization for sample data (without meeting recording):

In case, the meeting recording is available, which by default all major video conferencing platforms provide, upload the recording to contextualize the data w.r.t meeting. In this scenario, the video progress bar will be overlayed with information regarding notes captures. This help users mark important moments in the meeting.

Visualization for sample data (with meeting recording):

With meetings, power mode is extremely useful for extracting information such as sentiment score, engagement score, questions asked and outliers. Below is the visualization for sample data when using power mode and analysis CLI.

Visualization for sample data in power mode (with meeting recording):

In the image above real time engagement and sentiment graphs can be seen on video timeline. In addition, you also see questions asked during meeting and outliers.

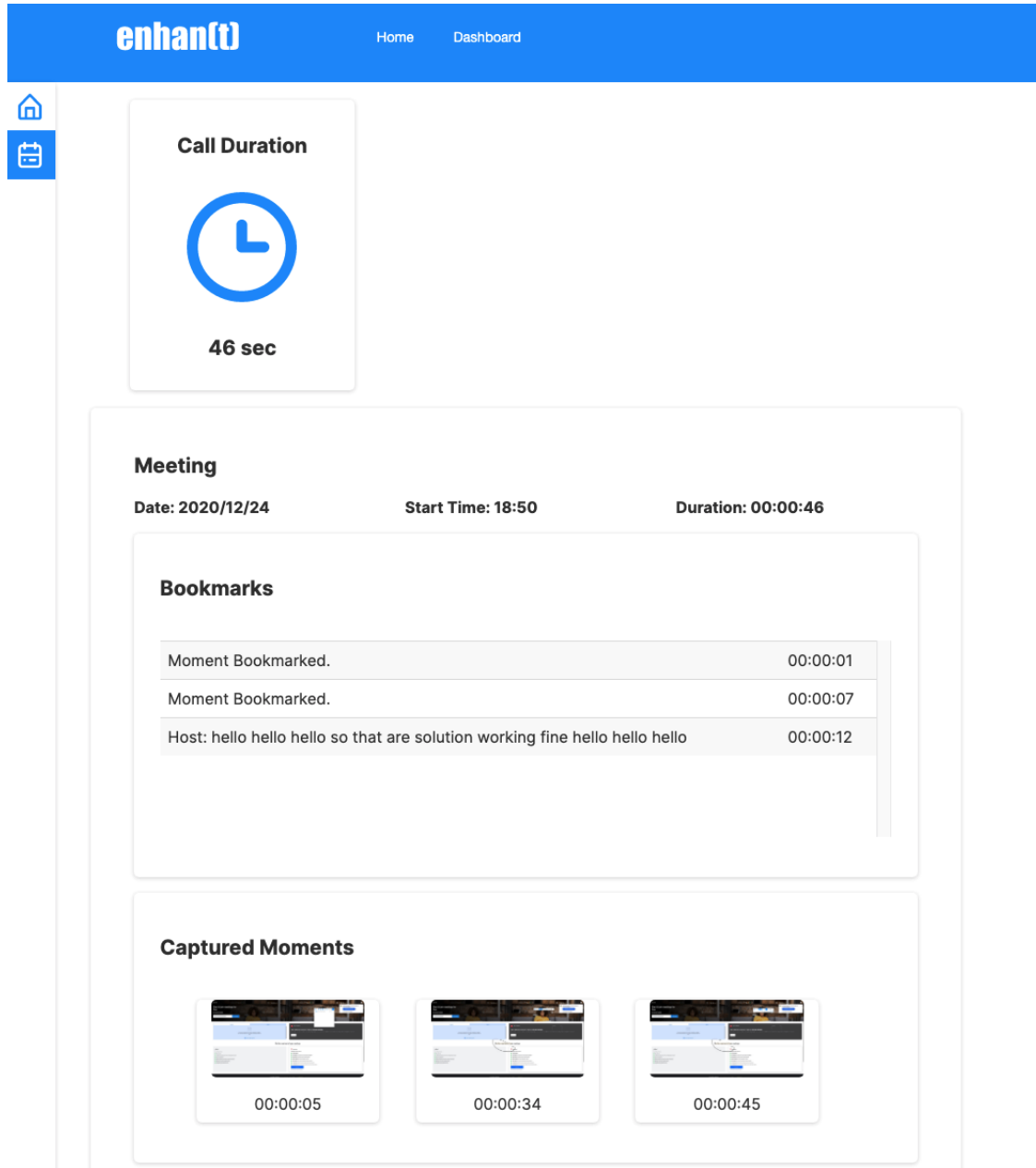


Fig. 11: enhan(t) visual tool - without meeting recording


enhan(t)

Home Dashboard

Home

Calendar

Call Duration




46 sec

Meeting

Date: 2020/12/24

Start Time: 18:50

Duration: 00:00:46



Play

Bookmark

Camera

Bookmark

Bookmark

Share


00:00:00 / 00:00:21

Settings

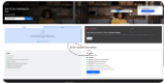
Bookmarks

Moment Bookmarked.	00:00:01
Moment Bookmarked.	00:00:07
Host: hello hello hello so that are solution working fine hello hello hello	00:00:12

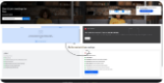
Captured Moments



00:00:05



00:00:34

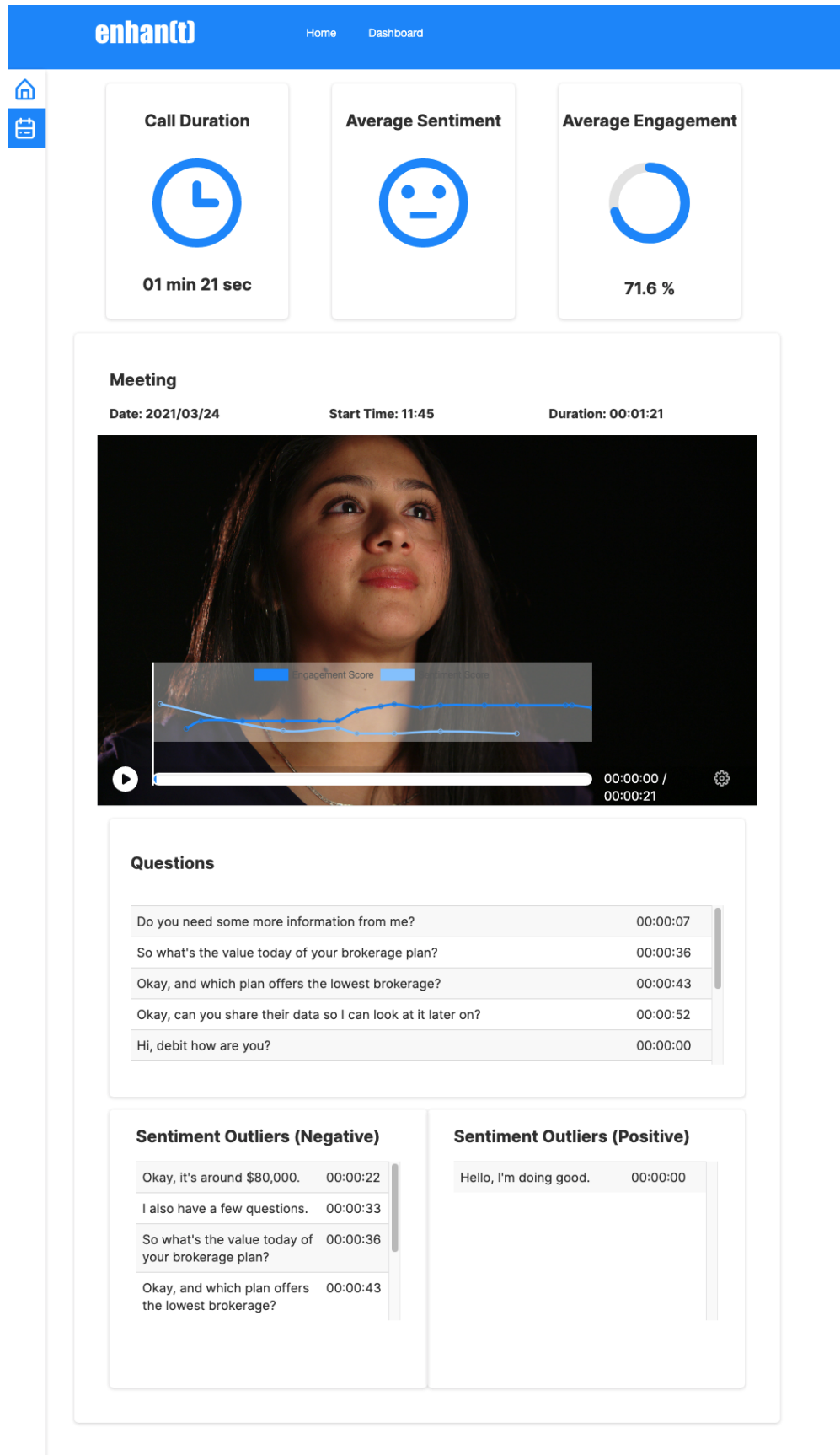


00:00:45

2.10. Tutorials

Fig. 12: enhan(t) visual tool - with meeting recording

27



2.10.2 Using enhan(t) for learning purposes

enhan(t) can be used with e-learning platforms (or any site) for personal development and learning.

When in basic mode, enhan(t) will collect all data in a .zip file (downloaded when the recording is stopped). The zip file contains the following:

1. **host.srt** : A subtitle file containing the transcription data of the host.
2. **images** : Folder containing all the images captured using enhan(t).
3. **notes.pdf** : The contains all the data captured via enhan(t) and in the order it was captured w.r.t time. The pdf file can be shared with others as part of meeting notes.
4. **notes.txt** : This is a raw text file containing the meeting data. It contains relative path to the image inside the images folder. The txt file is plain raw data for developers to consume it in other applications based on their need
5. **notes.md** : Markdown is a standard in lot of note-taking applications and platforms. Users can visualize the notes in either Visual Studio Code or a note-taking app such as Evernote / Joplin. These platforms support Markdown import so you can collect notes via enhan(t) and then import it inside a note-taking app.

When in power mode, enhan(t) will also transcribe the audio data for guest (speakers on the other side of the call). This transcription is available in a separate **guest.srt** file inside the zip. When run in conjunction with the analysis CLI, users will also get engagement and sentiment scores, as well as, the outliers (both negative and positive) during the meeting.

For more details on using the power mode, refer the guide [Power Mode](#).

Lets look at some scenarios on how the data from enhan(t) can be consumed:

1. Use enhan(t) visualizer:

enhan(t) comes with out-of-the box visualizer, details here [Viewing the meeting data](#). This is a web based visualization tool that works only with enhan(t) zip files. The zip data downloaded using the enhan(t) extension can be uploaded here, with an optional meeting recording.

Look at some examples here: [Using enhan\(t\) in Meetings](#).

2. **Use Visual Studio code**: Visual studio comes with markdown support out-of-the-box. Here are the steps to import enhan(t) data inside VSC:

1. Import the zip file inside a folder.
2. Install plugin zip-extract-all (if not installed) from [here](#).
3. Extract all zip files inside the folder.
4. Click on notes.md inside the extracted folder to see the note.

We encourage users to add links in the **notes.md** file to navigate from one note to another. To do this simple add the following to source markdown file:

```
[name_of_the_note](relative/path/to/destination/note)
```

For more details on markdown support with visual studio code, visit <https://code.visualstudio.com/docs/languages/markdown>.

3. **Use Evernote**: Evernote own its own does not fully support markdown, but when in conjunction with Marxico (<http://marxi.co/>) - a delicate markdown editor for evernote, one can import enhan(t) **notes.md** file.

4. **Use Joplin**: enhan(t) generates the **notes.md** file that can be imported into joplin out-of-the-box. Note that the images needs to be imported separately because Joplin does not import relatives path of images on its own. Details on how to import markdown in Joplin is mentioned here : <https://joplinapp.org/>

2.11 Tips, Troubleshooting and FAQs

2.11.1 FAQ

Speech to text does not work in firefox?

The web speech API for firefox is not enabled out-of-the-box. To use the API :

First make sure you are running Firefox Nightly newer than 72.0a1 (2019-10-22). Then type *about:config* in your address bar, search for the *media.webspeech.recognition.enable* and *media.webspeech.recognition.force_enable* preferences and make sure they are set as **true**.

Refer [here](#) for more details.

What does offset mean in enhan(t) Dashboard Viewer?

If you upload recording in addition to the meeting data, you see a separate video section in the dashboard. On top of the video seekbar / progress bar you will see how notes have been captured over time as well as the variation of sentiment and engagement score (in a line chart). On clicking the settings icon in the video media player, you see the option to enable/disable the chart and one to add a time based offset of format hh:mm:ss. This offset is a means for you to indicate time difference between the moment you started the recording and the moment when enhant plugin was started.

For example, if your recording started 10 min before the enhant plugin - you will add the offset value -00:10:00 and if the recording started 10 minutes after the plugin you will add the offset value of 00:10:00. Doing this will shift the timeline for all the data across the dashboard and you will get better sense of guest actions over the course of entire recording.

In a situation where the offset is positive (recording started after the enhant), any data collected before the recording will be represented in negative time. This happens because data is now represented in the context of the recording uploaded.

By default - it is assumed that the plugin and the recording started around the same time.

Note: Offset calculation is a manual process, and needs to be evaluated separately because recording is not part of the enhant system so there is no way to evaluate inside of dashboard automatically.

What are the supported languages?

The Chrome Extension provides the host side transcription (from microphone) in both Basic Mode and Power mode. Currently we provide support for English (US) and English (IN) for host side transcription.

Guest side transcription (from tab audio) is available only in the Power mode when used in conjunction with the transcription service. When the default Vosk service is used, it can only provide transcription optimized for English (US). If the alternate Google Cloud Speech to Text service is used, it can provide transcriptions optimized for both English (US) as well as English (IN), as per the setting applied in the Chrome extension.

Error:- Docker container crashed

If your transcription service crashes or docker goes down, please raise a issue on github with details. Check if you have allocated sufficient RAM to docker container by going to docker dashboard, enhant local transcription service needs at least 4 GB of RAM for it to work without issue, After allocating sufficient memory/RAM restart application. To restart the application, open terminal and run the following commands:

```
cd <path-to-repo> //you need to be in your repo folder
docker-compose up
```

enhant will start again

Error:- certfile=cfg.CERT_FILE_PATH, keyfile=cfg.KEY_FILE_PATH FileNotFoundError: [Errno 2] No such file or directory

Make sure you have installed local ssl certificate before starting service using docker-compose up. Refer [How to add certificate for localhost for cli utility](#) for instructions

How to remove dangling Docker images?

If you are using docker to build and manage enhant it may happen that after running *docker-compose build* multiple times you may start to run out of disk space. To recover some of this disk space you can remove dangling docker images using this command:

```
docker rmi $(docker images -f 'dangling=true' -q)
docker system prune
```

2.12 Reference

2.12.1 enhant nlp lib

enhant AI Service is implemented at **/nlp_lib** in python language. It is further divided into three separate modules. First one is nlp_lib module with

nlp_lib_engagement

nlp_lib_questions_finder module

nlp_lib_sentiment module

2.12.2 enhant cli

enhant cli is an python application located, code for which could be found at **enhant-repo/cli**.

First one is main module with

main enhant-cli-app module

config

engagment_finder module

questions_finder module

sentiment_finder module

INDICES AND TABLES

- `genindex`
- `modindex`
- `search`