

***Advanced Desktop Application
for Audio Acquisition (ADA3)***

***DOCUMENTATION
(GITBOOK)***

Harsha Vardhan Bayyapu

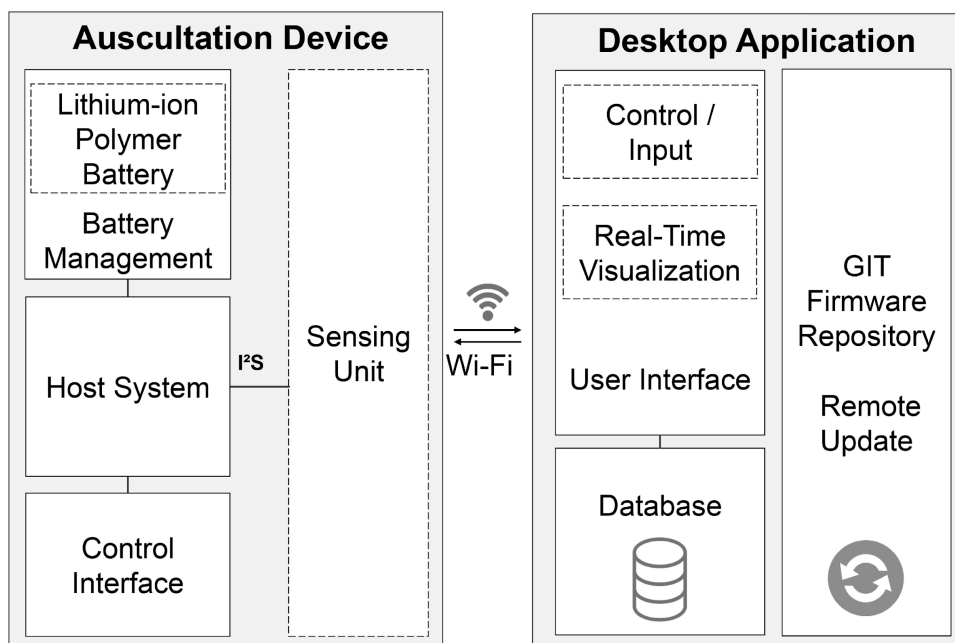
Pranay Teja Arikatla

CONTENTS

1. User manual	
Introduction	-1
Application Installation	-3
Usage Guide	-11
Troubleshooting	-18
2. ADA3 report	
Requirements	-20
Setup a new auscultation device	-22
Desktop Application Development	-23
Dual Communication	-36
LED Implementation	-39
Database (File format)	-41
Application Testing Cases	-44

INTRODUCTION:

For the diagnosis and monitoring of cerebrovascular diseases costly hardware is used, but with the help of CAAS (Computer Assisted Auscultation Systems) we can monitor the anomalies in the blood flow patterns in the Carotid Artery, which can be helpful in identifying any such diseases.



Bodytune is an auscultation device based on Raspberry Pi Zero W that captures the carotid sounds from some regions of the neck and transmits them to the desktop application through wireless communication.

ADA3 is a desktop application consists of a Graphical user interface that provides visualization of the audio data transferred from Bodytune .

In this desktop application we can access 5 different GUIs on a single platform. The application receives streamed data from the bodytune device through Wi-Fi which is visualized in the GUI. Furthermore, all the details entered in the application along with acquired audio signal is saved in a database.

INSTALLATION:

INSTALLATION PRE-REQUISITES:

This installer only works on Windows 10,11 64-bit operating system only.

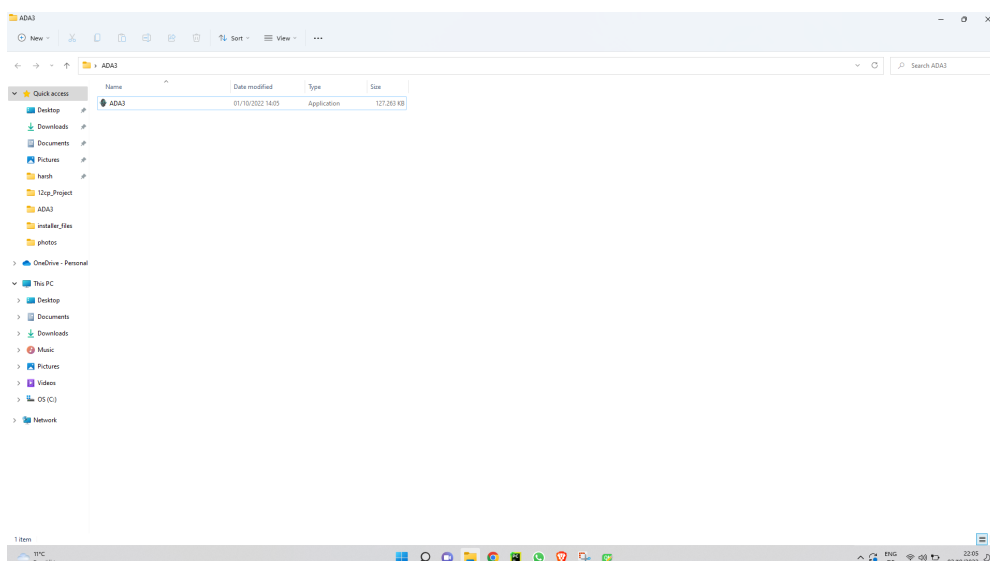
Most reasonable up-to-date Windows machines with Windows 10,11 run a 64-bit OS on a x_64 processor architecture. You can verify your system specifications on Windows like explained in the following:

1. Open the Windows settings
2. Select "System"
3. Select "Info" (German) or "About" (English)
4. Next to "Systemtyp" (Deutsch) or "System Type" (English), your system's OS and architecture is listed there.

Download the ADA3 installer for the windows operating system from the link below.

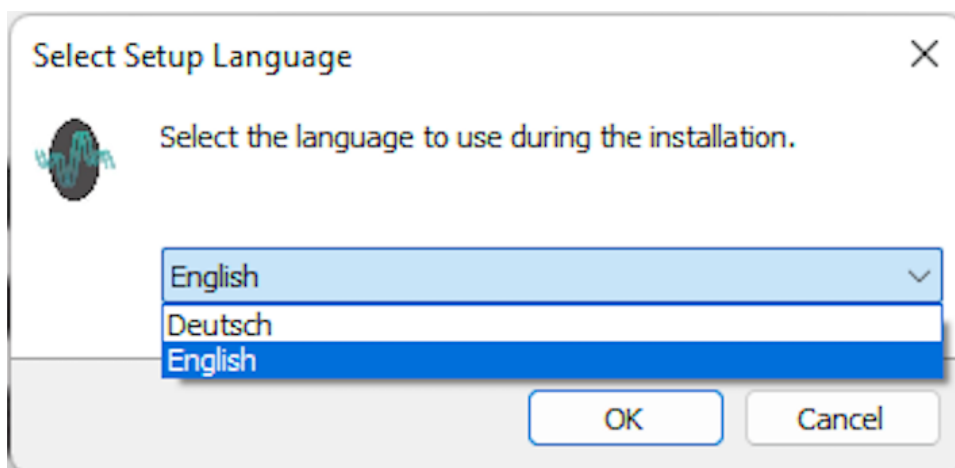
Link: <https://www.dropbox.com/s/vdpn721p7wa06f0/ADA3.exe?dl=0>

Note: Windows will probably try to protect you, since it cannot verify the source of the file you download. You can skip these warnings and continue your downloading.

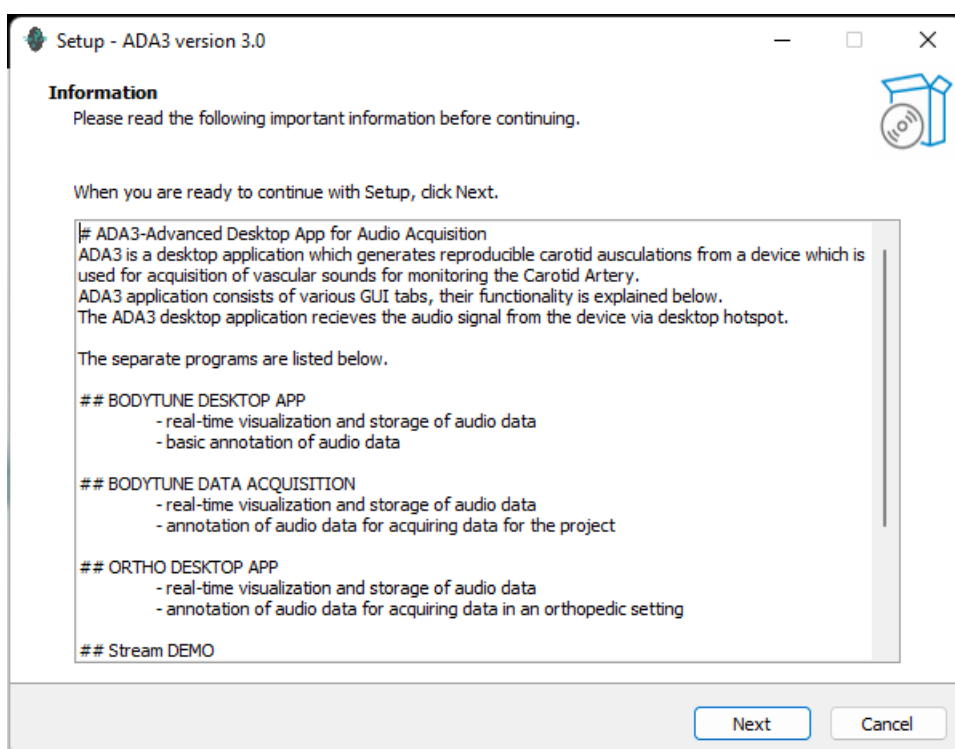


After downloading the installer, follow the steps below to install the application on your desktop.

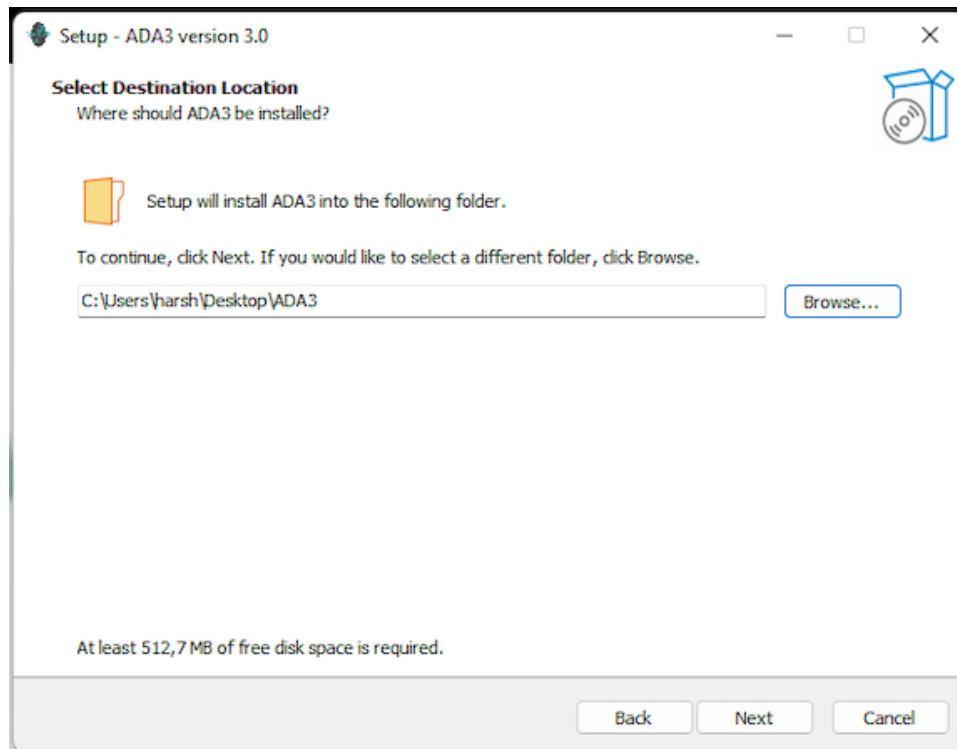
- Double click on the Ada3.exe file.
- Permission to make changes to your desktop dialog pops up to which you must select “Yes” to continue with the installation process.
- Select the setup language to use during the installation process. (English/Deutsch)



- After reading the information press Next.

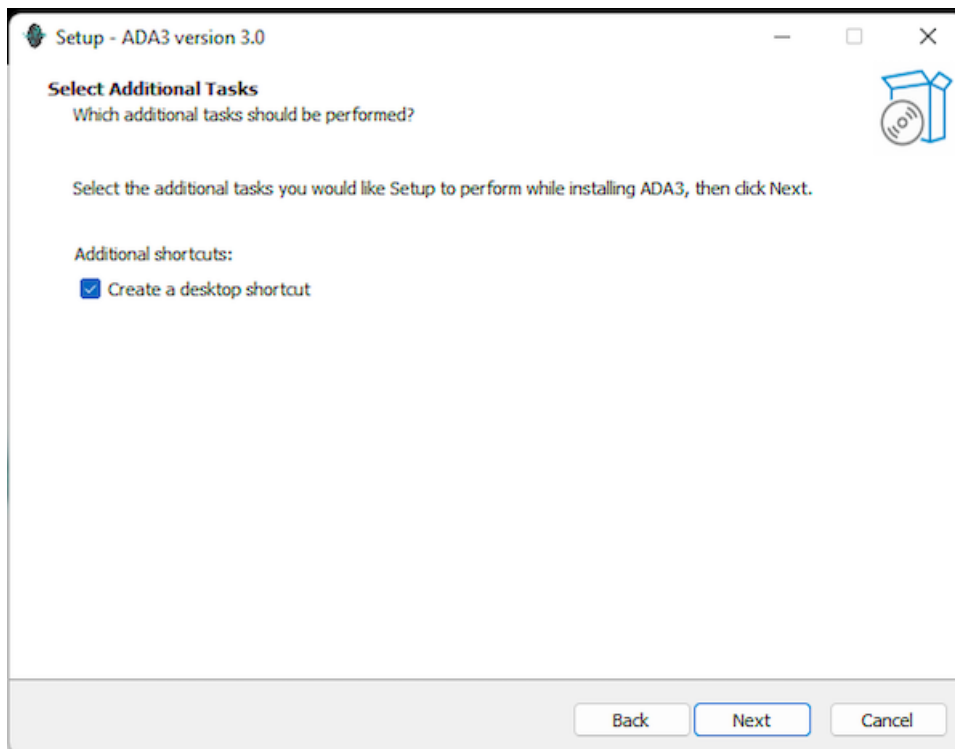


- The installation dialog will open and prompt you to select an installation directory.
- Please click Browse... and then select Desktop as the Destination Location for installation.

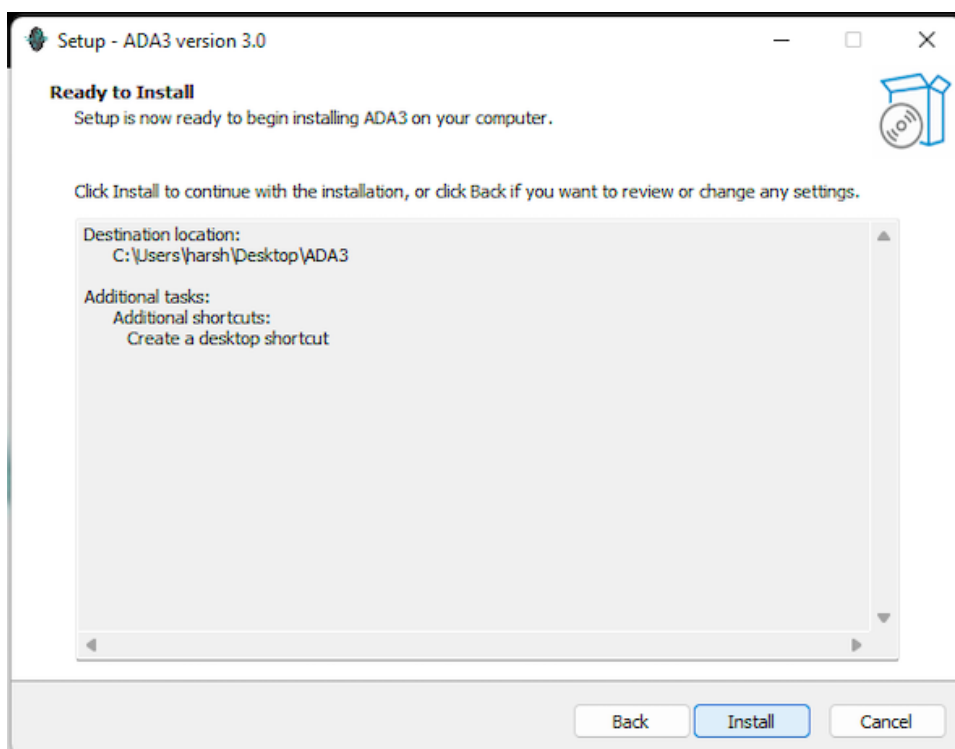


Note: Please choose any directory other than C:\programfiles to avoid problems while opening the application. If the application is installed in the C:\programfiles, you must run as administrator to open the application.

- Make sure there is free disk space (at least 500MB) available on the drive.
- Press the checkbox to create a desktop shortcut to make it comfortable for you to access the application and click next.



- Click Install.



- Uncheck the box to not launch the ADA3 because we need to setup hotspot before launching the application.

Setting the Device Name

1. Open the Windows settings
2. Select "System"
3. Select "Info" (German) or "About" (English)
4. Select "Rename this PC"

The new device name **must** contain one of the words laptop, desktop, rainer, bodytune or msi. This is important as it enables the auscultation device to identify the device it should connect to.

Some examples for valid device names:

- bodytune_host321
- windowsmsi
- windows10desktop
- LAPTOP-063G42W

Some examples for invalid device names:

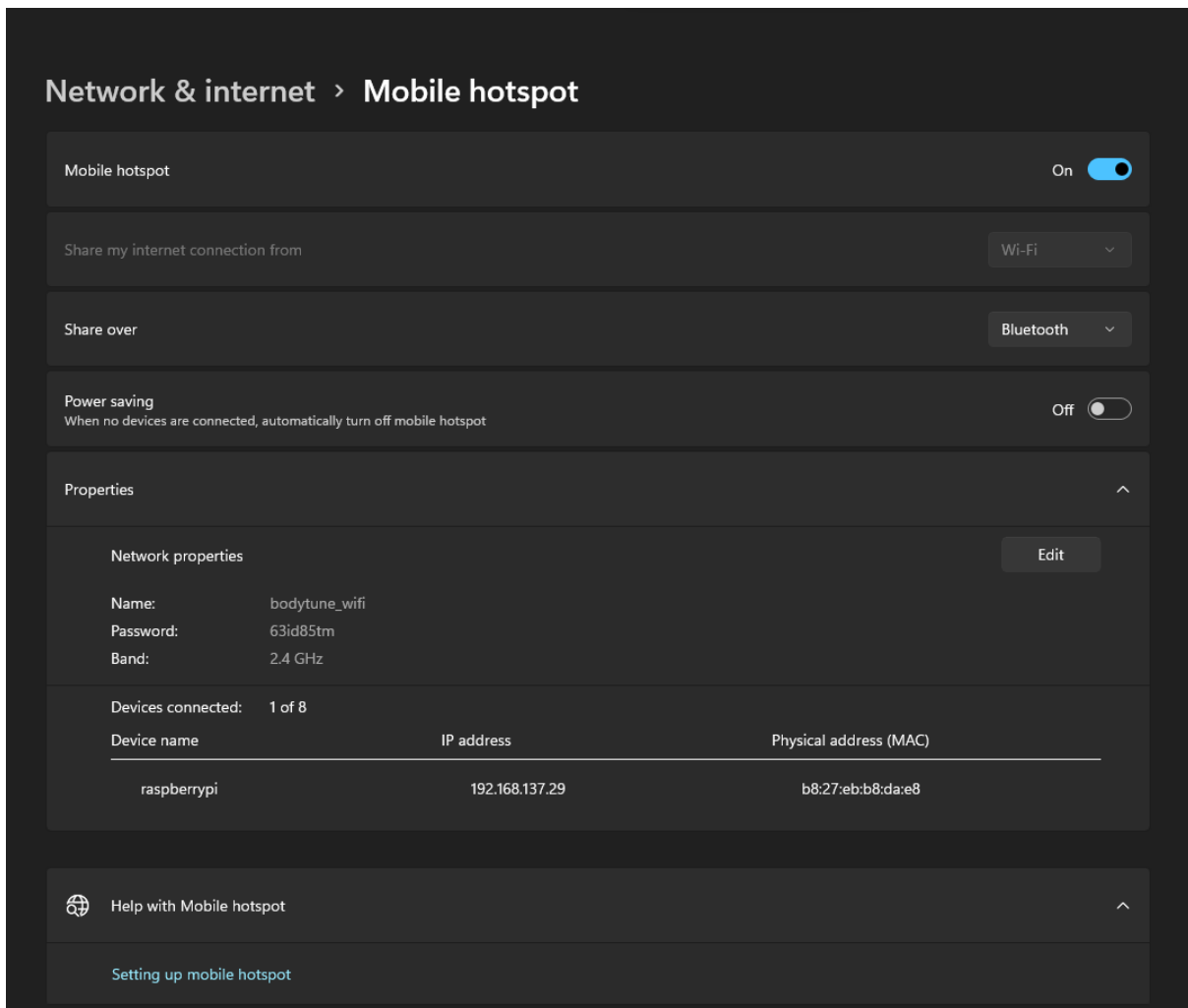
- windows10
- host567
- moritz's-device

Note: After renaming the device, please restart the device. Only after restarting the name will be changed

Setting up Mobile hotspot:

Open the "setting" and then go to "network and internet" and then go to "mobile hotspot". Click on the edit button to change the name and password of the hotspot.

- Hotspot Name: bodytune_wifi
- Password: 63id85tm
- Hotspot should be Turned ON



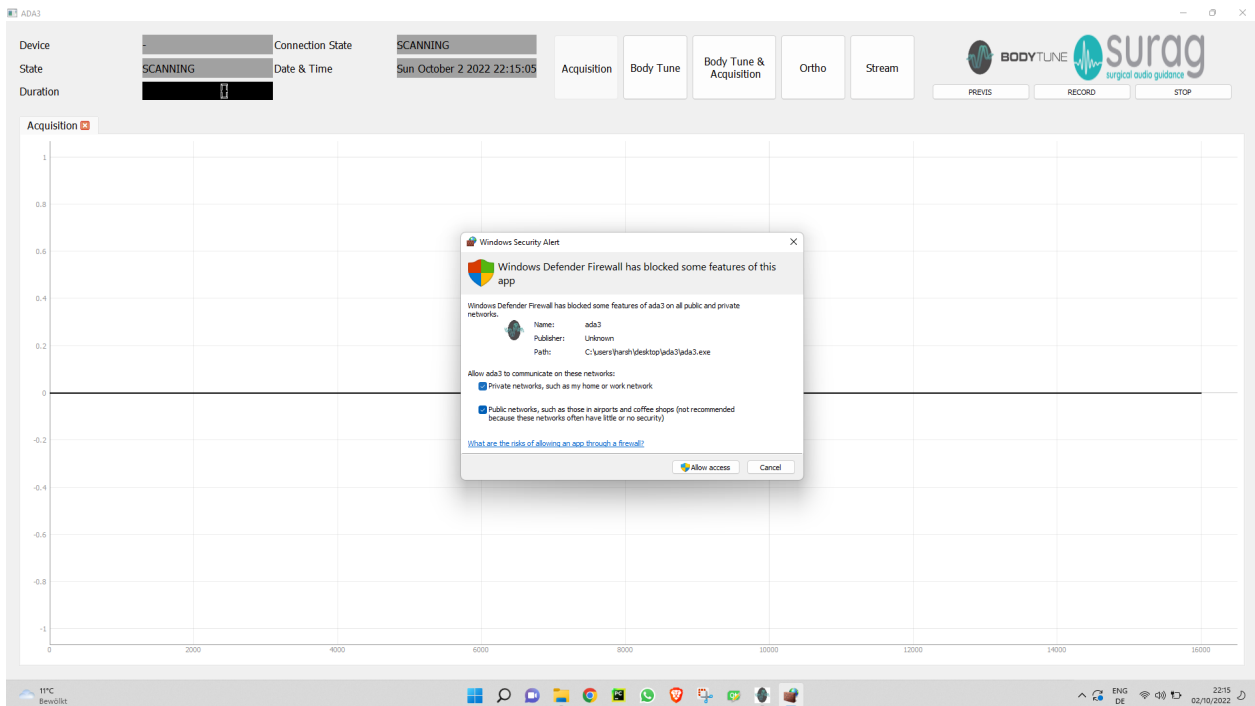
In order to connect the bodytune device to the desktop these requirements are to be met.

Finally, after setting up above requirements. Turn ON the bodytune device. The device should automatically connect to your desktop at this point.

Once the device is connected you can open the application shortcut which is created on the desktop.

-Note: If either the hotspot is not turned ON or the Raspberry Pi is not connected to the hotspot, the application won't open.

- Select both checkboxes and allow access to the application for windows defender firewall.



The connection state of the device is shown on the application.

Now, you can use the application to help the people by diagnosis and monitoring cerebrovascular diseases and make the world a better place.

Usage Guide:

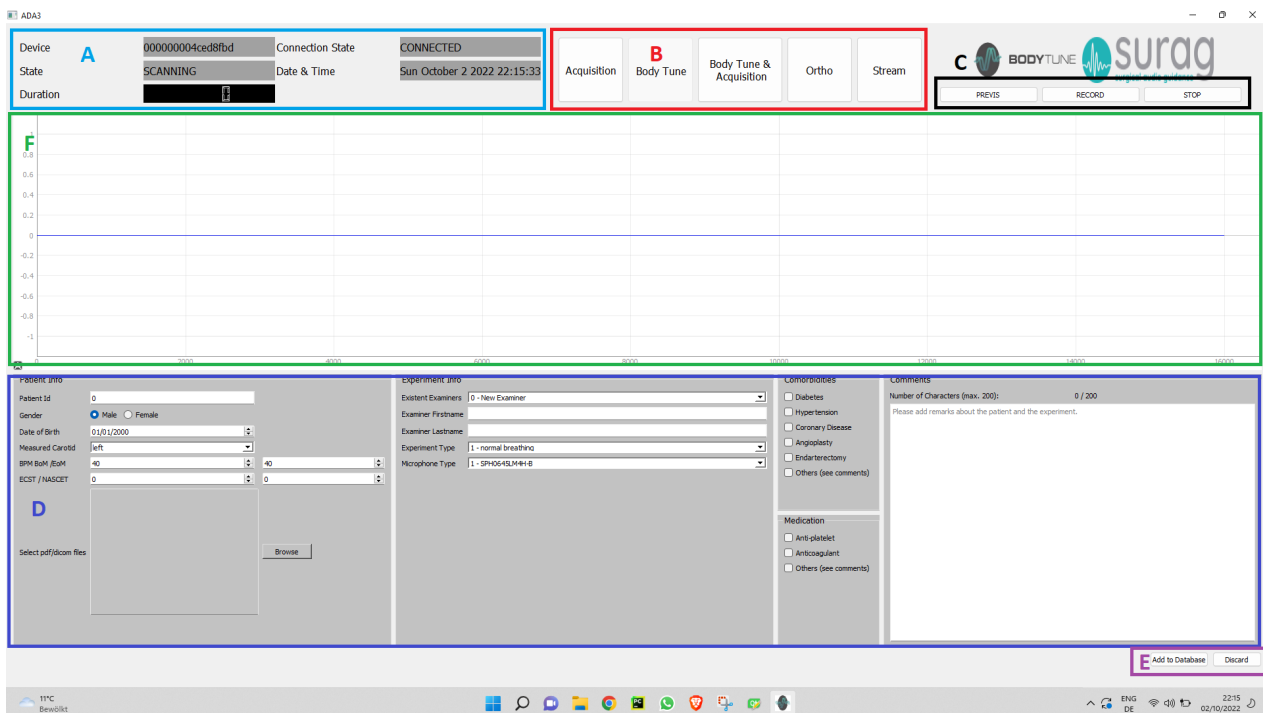
This guide will only work, if you installed the desktop application by following steps from Installation guide.

Here you will see the steps on how to use the application.

Launch the ADA3 Desktop Application

After launching the application, a GUI opens up.

Below you have the layout of the GUI and it features highlighted and explained.



A: Control center

- ☐ STATE - The state the auscultation device is currently in. Can be IDLE, SCANNING, VISUALIZATION, RECORD and STOP.
- ☐ DURATION - The timer to keep track on the experiments.
- ☐ DATE & TIME - Current Real-Time Date & Time.
- ☐ CONNECTION STATE - Shows whether or not the auscultation device is connected. You can only receive data if the device is connected. If the auscultation device is connected to the laptop (verify in the mobile hotspot

settings), but this label shows disconnected, have a look at the [Troubleshooting](#) section.

- ❑ **DEVICE** - Shows the auscultation devices serial number, which is a unique identifier.

B: Tab widget – By clicking on the tabs you can change to the desired GUI based on the requirement and usage.

C: Control buttons - Here you have the buttons which performs the similar operation to that of the bodytune device buttons. This helps in performing the experiments from the application itself.

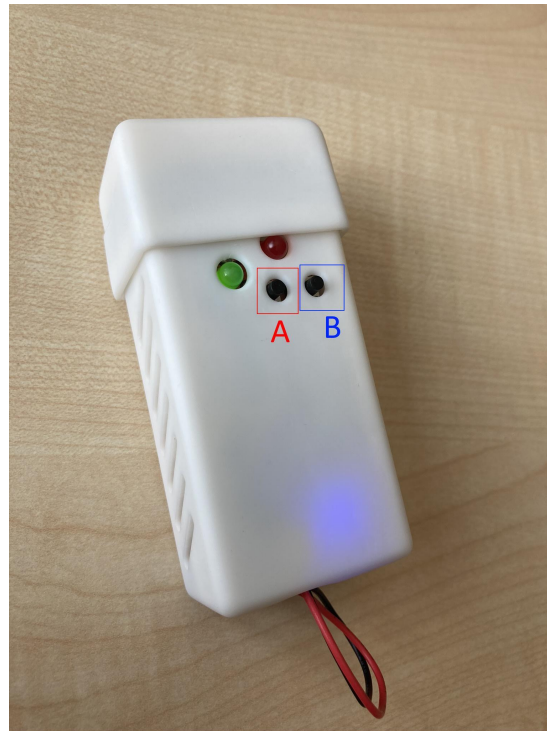
D: Annotation widget - Here you can add various information about the patient, the experiment, commodities and medication as well as further comments. And you can also add extra documents of format (pdf, dicom), which will be saved in the local database with the remaining files.

E: Save and Clear button - Here you will have the buttons to save the experiment with all the data and you can also clear everything without saving, to continue with another experiment.

F: Visualization widget - Visualizes one channel of the received audio signal in real-time. Effectively, at a sampling frequency of 16.000 Hz.

Measurement Process

The measurement can be done using the buttons on the Bodytune device or the Control buttons on the application.

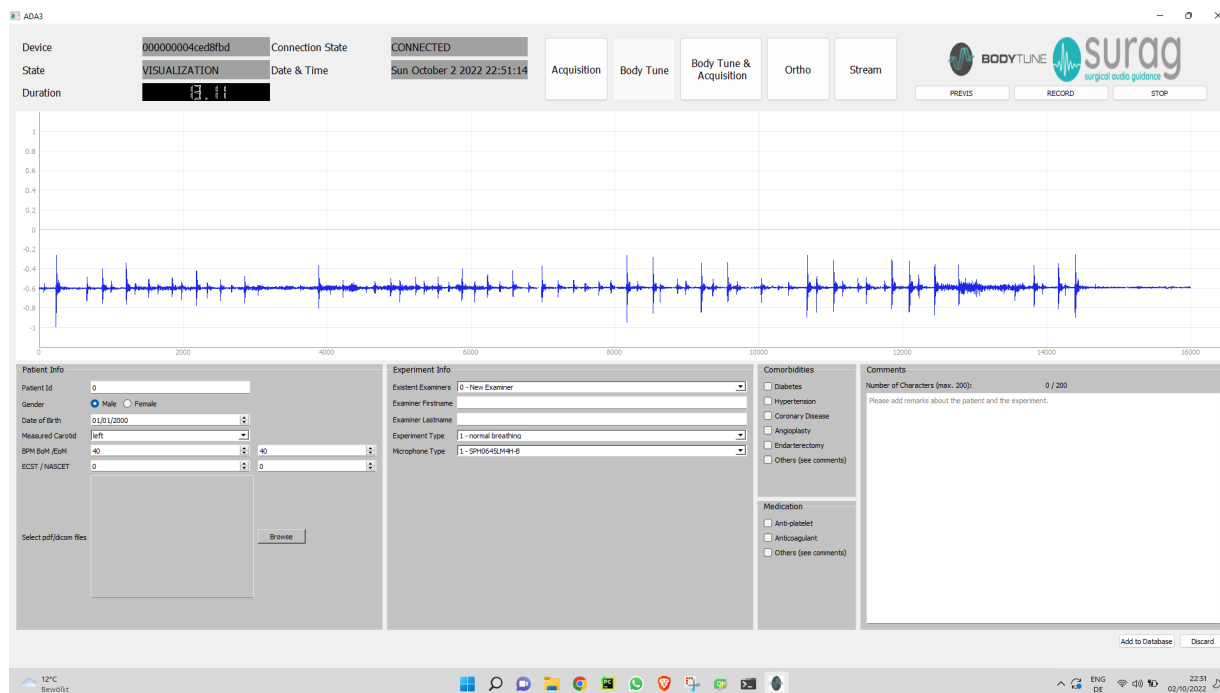


Below picture shows the basic idea on how the LED concept works, which shows the status of the bodytune device.

State	Button activity	Led1	Led2
Visualization	Button-1 press once	off	Blinking
Record	Button-1 press twice	off	On
Stop	Button 2 press	Blinking(2 times)	Blinking(2 times)
Idle	-	On	off
Shutdown	Button 2 pressed twice	Off	Off

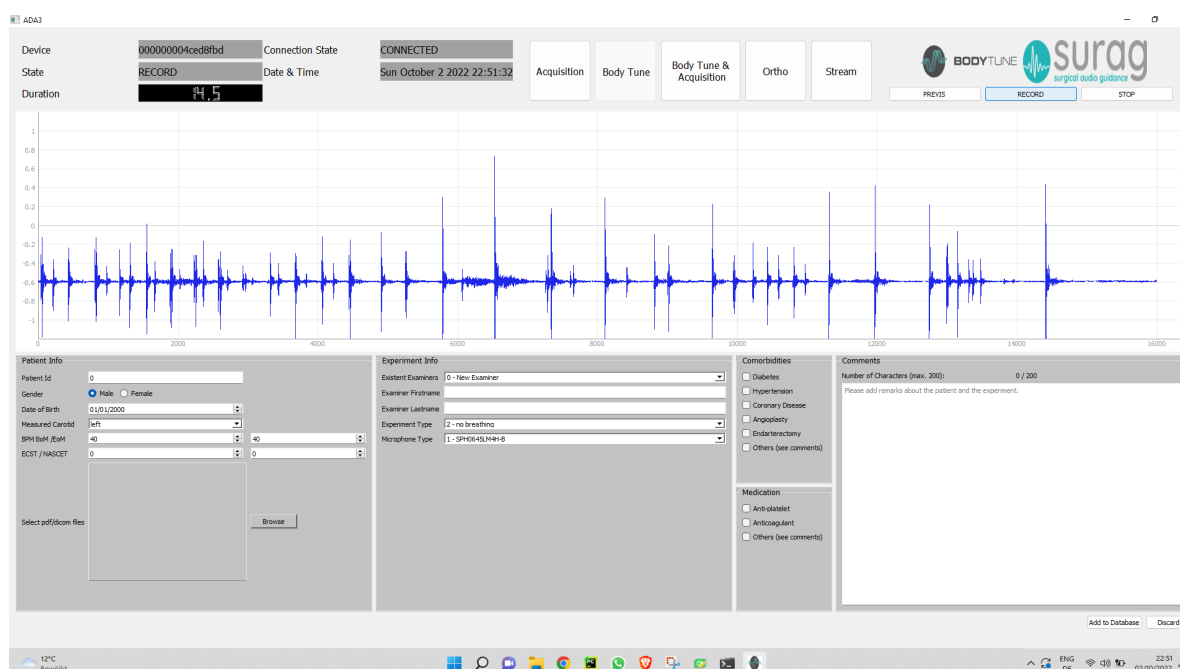
After pressing “Button A” on the auscultation device, the desktop app will start visualizing the received signal in **VISUALIZATION** mode, i.e., this signal is only visualized and cannot be stored.

You can also press “PREVIS” button on the top right corner of the application to start visualizing the received signal.



In **VISUALIZATION** mode, press again Button A to go to the **RECORD** mode. Now, the received signal will be recorded and later stored in **STOP** mode after adding annotation data.

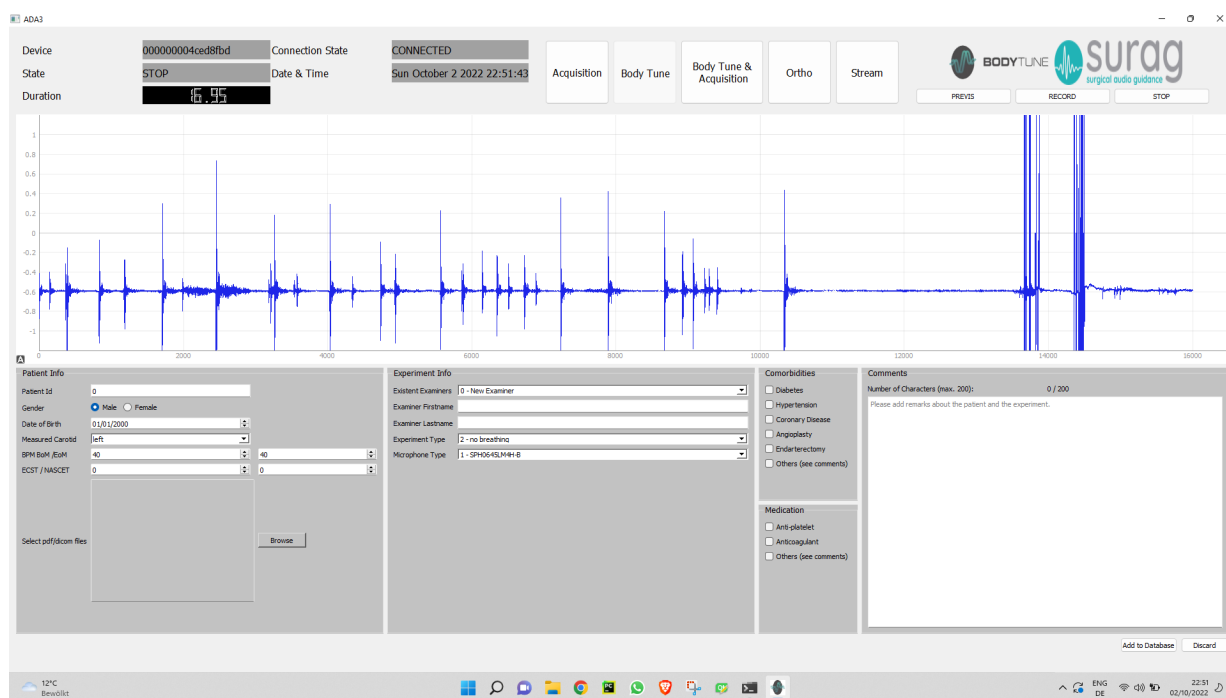
You can also press “RECORD” button on the top right corner of the application to start Recording the received signal



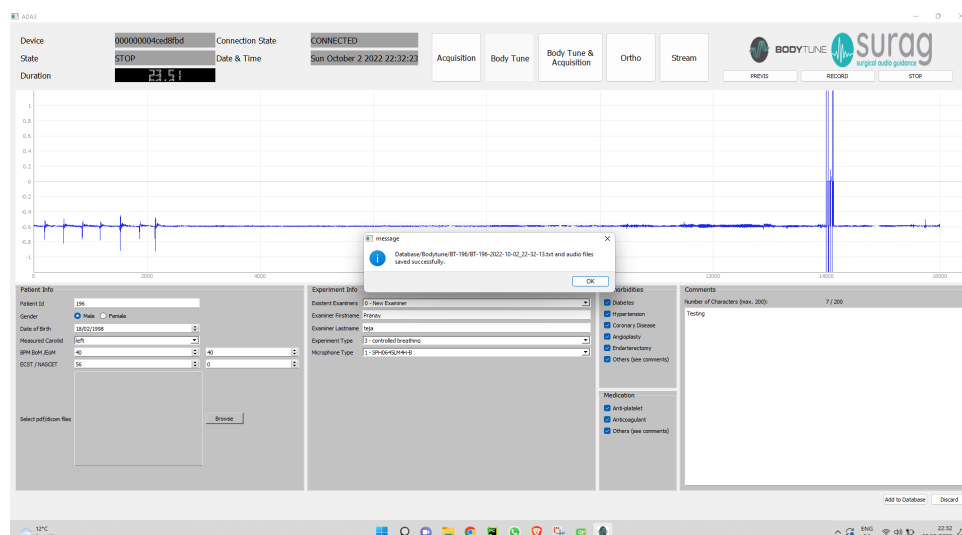
In **RECORD** mode, press Button B on the auscultation device to stop the recording. The desktop app's STATE label will show **STOP** and allow you to enter the annotation data.

You can also press “STOP” button on the top right corner of the application to Stop the recording

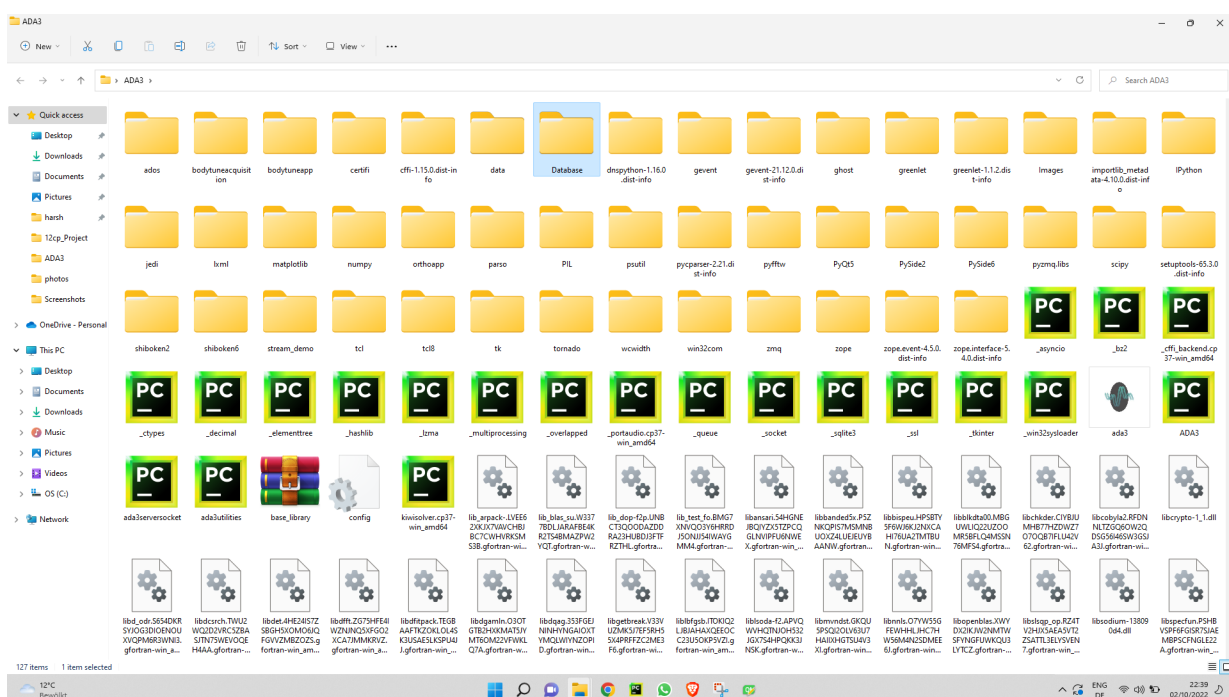
Note: Information in the annotation widget should be filled in order to store the data.



After adding annotation data in **STOP** mode, you can save the audio signal along with the annotation data by pressing Add to Database.



After the measurement is finished, navigate into your installation directory, e.g., C:/Users/Desktop/ADA3 to open the Database folder.



Files with respect to each GUI is saved under

- Body tune GUI - Bodytune
- Body tune and acquisition GUI - BodytuneAcquisition
- Ortho GUI – Ortho

Note: Inside “BodytuneAcquisition” folder a new folder is created with the examiner name inside which the patient folder is created

Patient details are further saved inside the folder with his ID/name of format BT-patient id/OR-name (ex:BT-964/OR-pranayteja)

- Inside the folder with the patient id, we save the patient information, examiner information and experiment information, medication and comorbidities in a text document of format (.txt)
- we save the audio data streamed as a wave file format (.wav).
- Additionally, files (PDF/DICOM) which are attached in the annotation widget are also saved in this folder.
- Every text document/wave file is created with a unique name.

Ex:BA-964-2022-09-25_10-10-26

- BA-964 - patient id
- 2022-09-25-Date of Measurement (YYYY-MM-DD)
- 10-10-26- Time of measurement (Hour-Minute-Sec)

The folder structure is implemented in such a way that every experiment from different GUI is stored in their respective folder, in which a folder is created with a unique patient id/name.

Troubleshooting:

Issues with the desktop and application

Desktop:

All the steps are explained based on a windows 11 OS and it is quite similar for a windows 10 OS.

- If the device is not connecting to the desktop.
 - check the desktop name which should be set as mentioned above
 - Check the name and password of the mobile hotspot
- Mobile hotspot: If the hotspot is not turning on. First check whether the Wi-Fi is turned ON or not. If it is not ON, turn it on and then turn on the hotspot and it should be working now

Desktop application:

- After the measurement is performed in the “Acquisition” GUI tab the application will not respond for some time (2-5 seconds) as it takes some time to post-process the audio signal and produce the result. So, be patient.
- If application crashes after opening.

It should be because of windows firewall or Antivirus

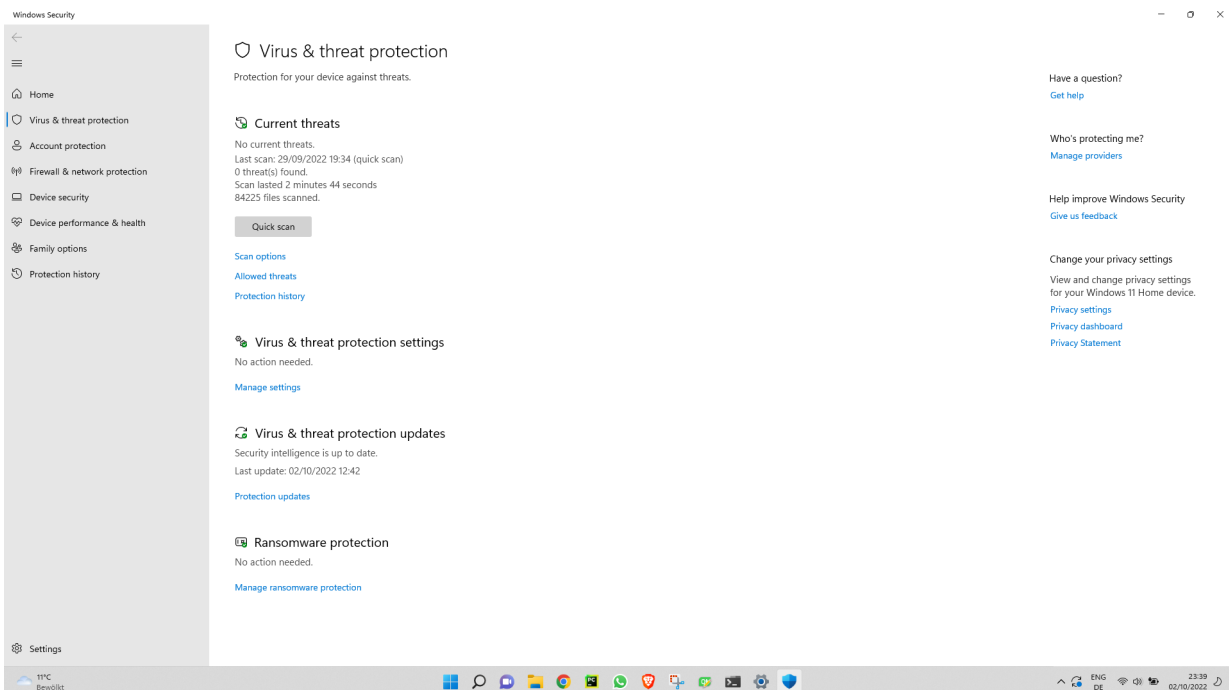
- Windows firewall
 - Go through the link to resolve this issue

Link:

https://app.gitbook.com/o/-LXP5F9G_r9JduBA3qNi/s/-LXP5F9JYKKH3ja1e0J3/bodytune-desktop-app-1/troubleshooting

- Antivirus:

- Open windows “Settings”
- Open “Privacy and security”
- Open “windows security”
- Open “Virus and threat protection”
- In current threats look for and open “protection history”
- Click on “threats blocked”
- You should see “Ada3.exe” in it
- There will be “actions” button in which you can select “ignore”
- The application will start working again



- If the state is not changing from the visualization to record or from record to stop even after pressing the buttons. Please restart bodytune device and also close the desktop application and open it again.

ADA3 report

REQUIREMENTS

Project Goal

1. An application which provides access to all versions of the GUI's
2. Application installer for Windows Operating System
3. Local storage of data in a single location
4. Two-way communication between the application and Body tune
5. LED concept in the Body tune Device

Mandatory Changes:

- Examiner name - The Examiner Name added once can be selected from Existing Examiners to reduce the hassle.
- Experiment running time - The experiment type in the Bodytune GUI changes after every 12 seconds.
- Adding Docs - The examiner can add additional documents required for the patient's analysis in the annotation widget before storing the data in local storage. Both pdf and Dicom formats are supported.
- Common Control Centre for all GUIs - Every GUI has a common Control Centre consisting of state, Connection state, Device Name, Date & Time, Duration, Tabs and Control Buttons.
- Getting Device ID from Configuration File - Every time when a device is connected the application searches for the Device ID from Config file and displays it in the control center or if a new device is connected it automatically adds it in the config file.
- Ortho GUI Scaling - The ortho screen is resized to fill the entire screen for a better user experience.
- Folder creation - The folders are created to save the data and audio signal

To know more about folder creation structure, go to Page 41.

- **New Annotations:**
 - Experiment Info - A whole new Experiment info section is added in the Bodytune and Acquisition's Annotation widget.
 - Patient id - The patient id is changed from int values to alphanumeric values.
 - Gender button - A radio button for the gender is added in the Bodytune and Acquisition annotations widget.

Failed Deliverables:

- Adapt the version to Bluetooth
 - The bit size in the RPI is very less, so data transferred through Bluetooth functionality is delayed
- Application for Linux and macOS
 - macOS: Cannot create hotspot simultaneously with Wi-Fi
 - Linux (Ubuntu 20.04): Program runs successfully for Linux, but the Body tune device is unable to connect to hotspot.

Future developments:

- Timer for experiments
 - Countdown timer where we can enter the number of seconds, we want to carry out each experiment
- Visualization of pre-recorded audio clips
 - Able to visualize already recorded audio clips

Setup a new Auscultation Device:

Go to this link to set a new different version raspberry pi

https://app.gitbook.com/o/-LXP5F9G_r9JduBA3qNi/s/-LXP5F9JYKKH3ja1e0J3/set-up-a-new-raspberry-pi-for-the-acquisition-of-audio-signals

- Setting up a new Raspberry Pi Zero W

Requirements:

You will need a raspberry pi, an SD-Card and a laptop along with Win32 disk imager software.

RPI image file:

Image file: A bootable or no bootable structure that contains files

The RPI image file consists raspberry pi OS along with all the programs which are necessary for the working of the bodytune device.

A raspberry pi image file is available at this link.

Link:https://www.dropbox.com/s/wftg8moogby5fkt/rpi_img_20Oct2020.zip?dl=0

Please download it. Ignore the warnings given by windows as it cannot verify the source of file you download.

Win32 Disk Imager software:

Win32disk imager program is designed to write a raw disk image to a removable disk. To download the software, go to this link

<https://sourceforge.net/projects/win32diskimager/>

- Format the SD card
- Select the RPI image file from the folder it is downloaded
- Select the device that is in the SD card drive
- Press write button

Once the operation is finished insert the SD card in the Raspberry pi.

Desktop application development:

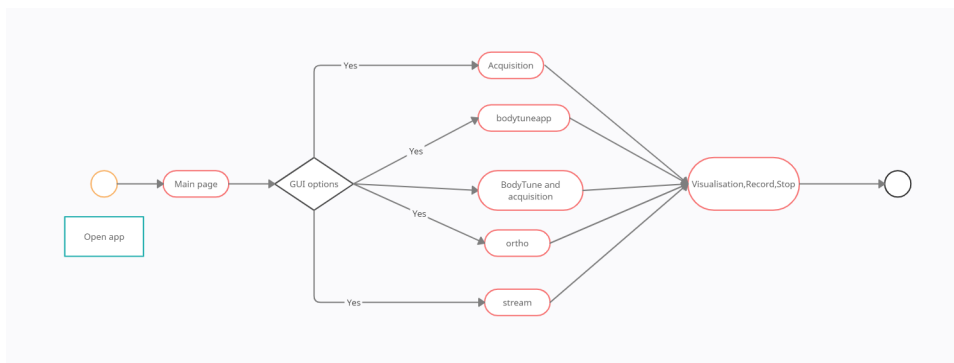
Here we will discuss about why and how we developed the graphical user interface (GUI) and made that into a successful fully functional desktop application for Bodytune and the tools used.

The desktop application was designed to provide a real-time visualization of the acquired audio signal, to add patient related data such as demographic information or comorbidities and to organize and retrieve acquired signals.

But we have 5 different versions of user interfaces with slightly different functionalities. This means five different applications if we want to use them. Which is not recommended.

So, we were tasked to develop an application which integrates all the different GUI and their functionalities and adding few more utilities to enhance the user experience.

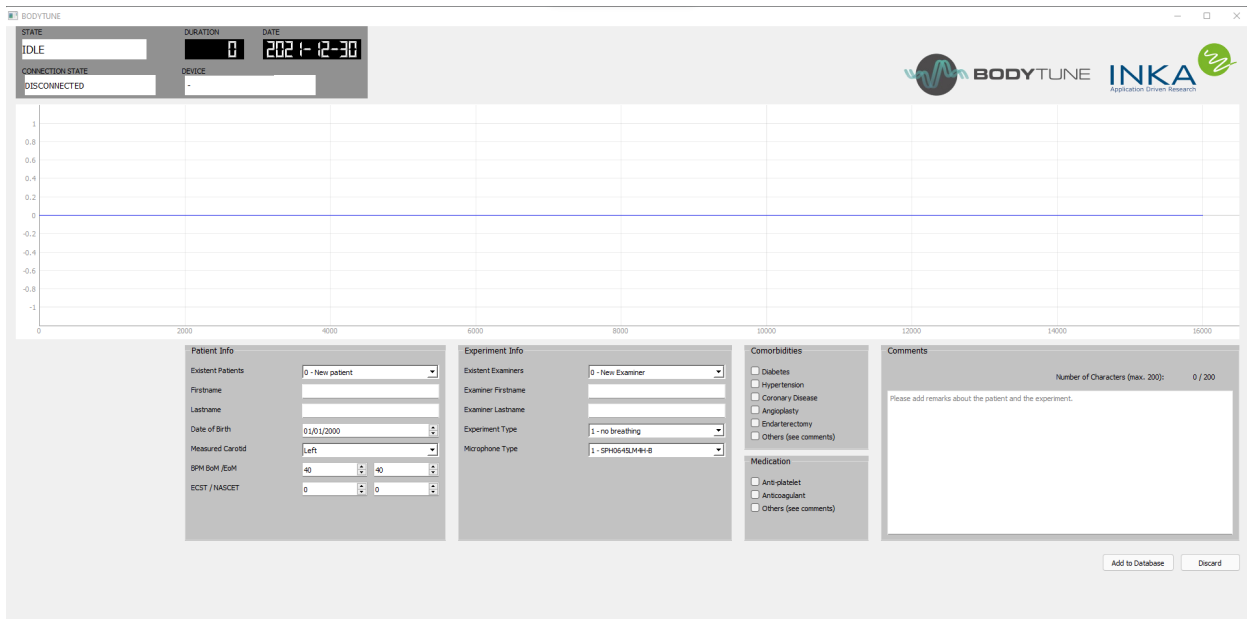
Activity diagram: Below diagram shows how the application is intended to work



Initial versions of the GUI's and functionalities:

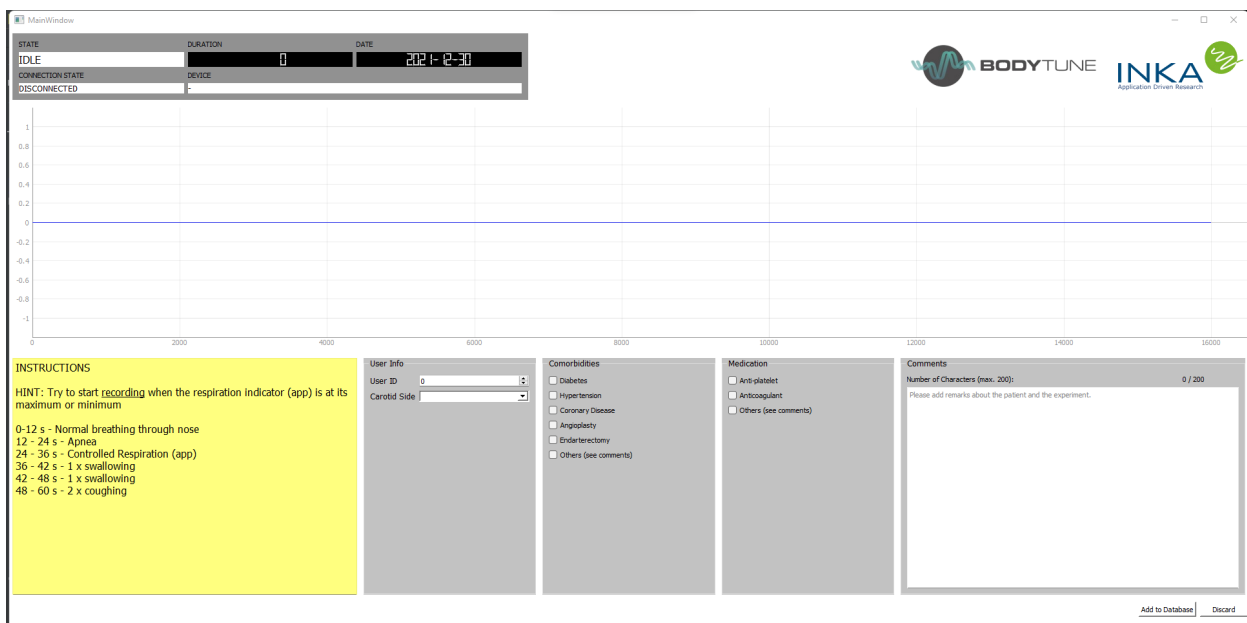
BODYTUNE GUI

- real-time visualization and storage of audio data
- basic annotation of audio data



BODYTUNE ACQUISITION GUI

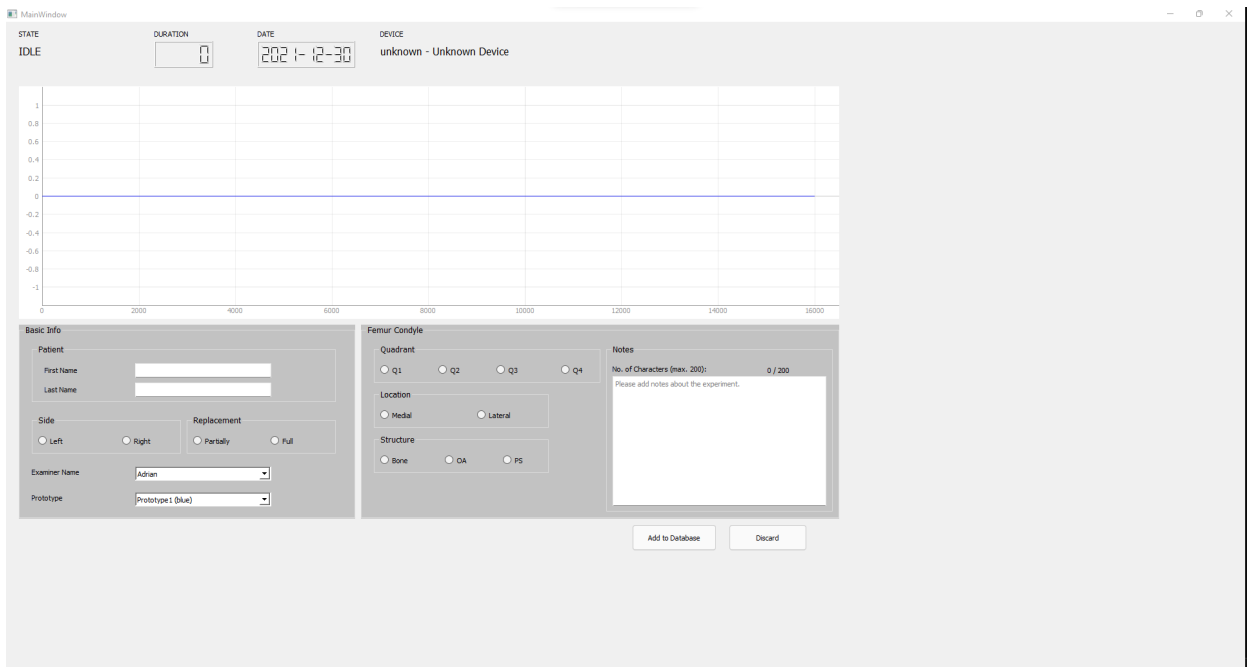
- real-time visualization and storage of audio data
- annotation of audio data for acquiring data for the project
- No experiment tabs or examiner details



ORTHO GUI

- real-time visualization and storage of audio data

- annotation of audio data for acquiring data in an orthopedic setting



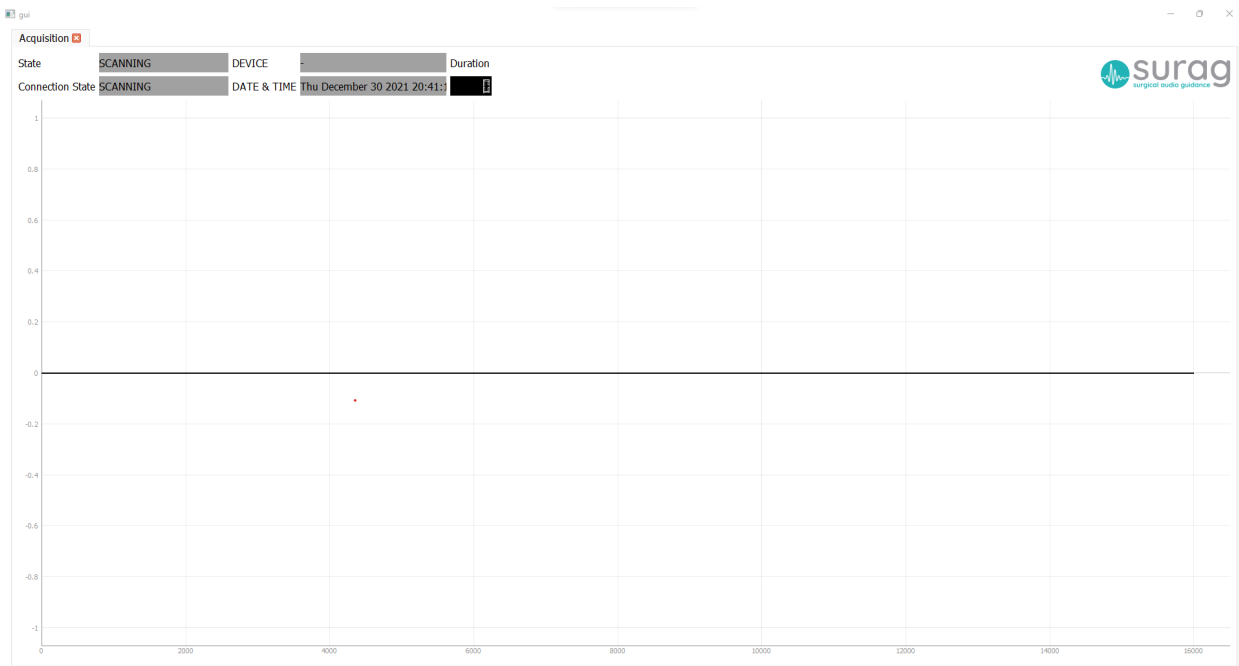
Stream DEMO

- real-time visualization of audio data
- playing back audio data in real-time



ADOS

- real-time visualization of audio data
- playing back audio data in real-time
- post-processing of audio data and display of the results



Development of a new integrated GUI:

Setting up the development environment:

Python:

Version 3.7.6

Download link: <https://www.python.org/downloads/release/python-376/>

The application was written and developed using python language and it is libraries

Note: While installing python select the “Add python3.7 to path”. If not, python will not be added to your system path.

To check whether python is added to the system path:

In command prompt enter: `python --version` it should give the version of the python you downloaded

To find where the python is installed in command prompt enter: `where python`

Pip:

Version: `pip 22.2.2`

Install pip: `pip install pip`

Pip is the package installer for Python. It is used to install packages from the Python Package Index and other indexes.

Pip is installed via command prompt.

When python 3.7.6 is installed older version of the pip is downloaded. So, it has to be upgraded.

Upgrade pip command: `pip3 install --upgrade pip`

UI design:

We used Qt Designer to design our user interface.

Qt Designer:

Qt Designer is the Qt tool for designing and building graphical user interfaces (GUIs) with Qt Widgets. It can compose and customize the windows or dialogs in a what-you-see-is-what-you-get (WYSIWYG) manner and test them using different styles and resolutions.

After completion of the UI interface, using PYQT we converted the UI to a python code which was further integrated into the implementation.

A standalone Qt designer can be downloaded from this link

<https://build-system.fman.io/qt-designer-download>

Qt designer can also be downloaded using pip

PYQT:

Version: 5.15.4

Installation: PyCharm/pip can use the requirement file to download the pyqt version necessary for the project.

PyQt5 is a comprehensive set of Python bindings for Qt v5. It is implemented as more than 35 extension modules and enables Python to be used as an alternative application development language to C++ on all supported platforms.

Integrated development environment:

An integrated development environment (IDE) is a software application that provides comprehensive facilities to computer programmers for software development.

IDE used: PyCharm

Download link: <https://www.jetbrains.com/pycharm/>

PyCharm IDE community version is used to develop and test the interface of the application code.

Setting PyCharm:

Interpreter: To set the interpreter press “Ctrl+Alt+S”

System interpreter: This is the place where the python is set in the system. This folder would contain many libraries and packages. So, it is suggested to have a local virtual environment which only has the libraries which are necessary for the present project.

Virtual Environment: To set the “Virtualenv” select add interpreter in the python interpreter and choose a location inside your project to set a local virtual environment which is specific to this particular project.

The PyCharm reads the “requirement.txt” file which consists all the necessary libraries and packages and installs them to the virtual environment (if virtual env is selected)

-Note: If some packages failed to install, they have to be manually installed. Using the tab “Python Packages” on the bottom left corner of the PyCharm interface it can be done.

-Note: Even then if they failed to install then they have to be installed using pip. Make sure the packages are saved into the “virtual environment” folder instead of the system interpreter folder.

Command to install the packages: `pip install <package-name>`

Pip also can be used to install the requirement.txt file

Command `pip install -r requirements.txt`

Note: packages which causes issues and need to be specially installed using pip “**package names**”

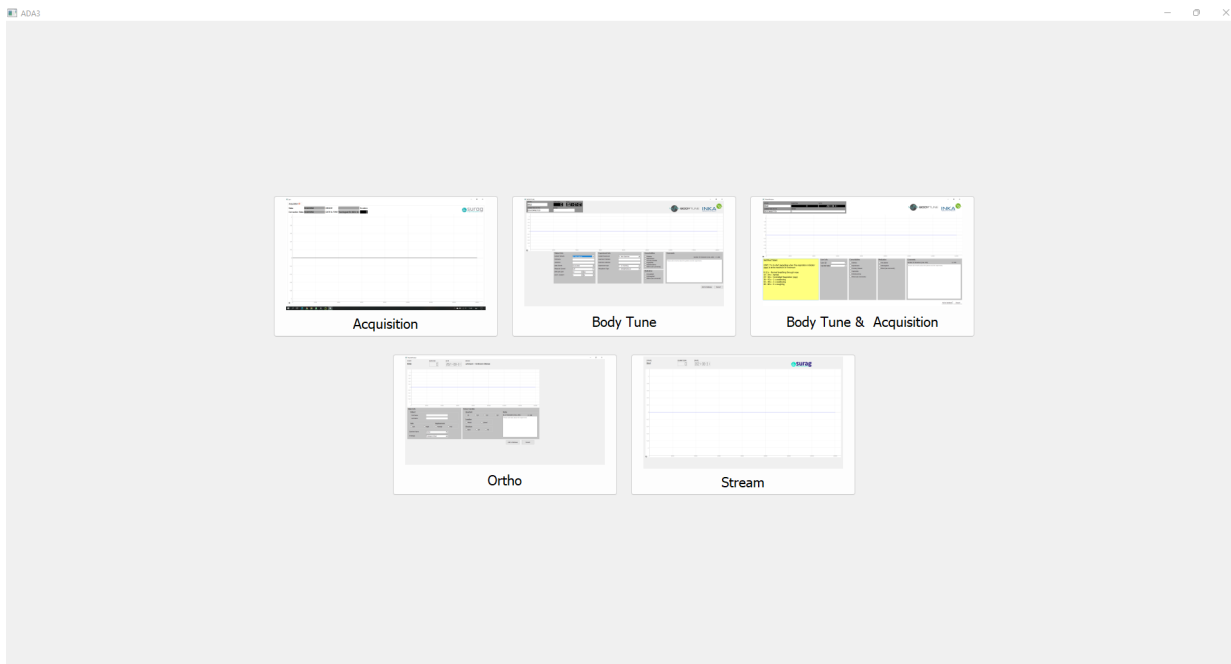
Using Qt designer, pyqt5 and PyCharm a new GUI is developed which integrates all the functionalities of the above mentioned 5 different GUI’s.

Advanced App for Audio Acquisition (ADA3):

ADA3 is an application which consists of a new and improved GUI which provides access to all the different GUI versions.

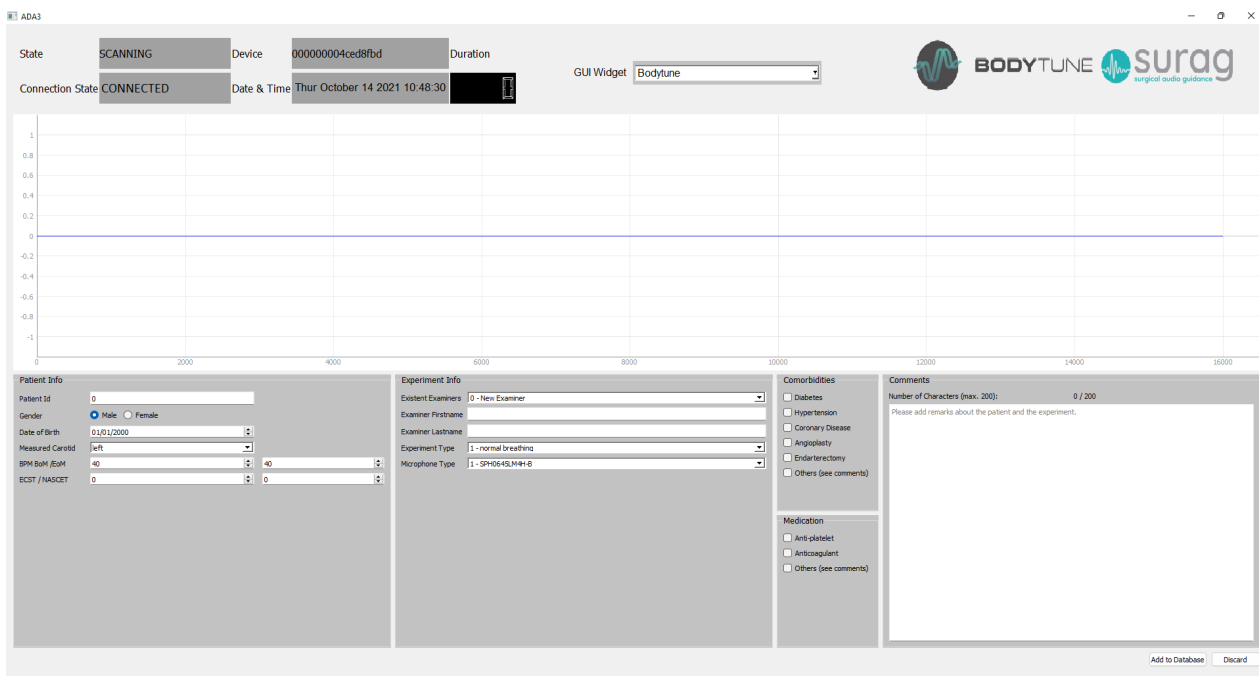
ADA3 GUI development versions:

Revision model 1:



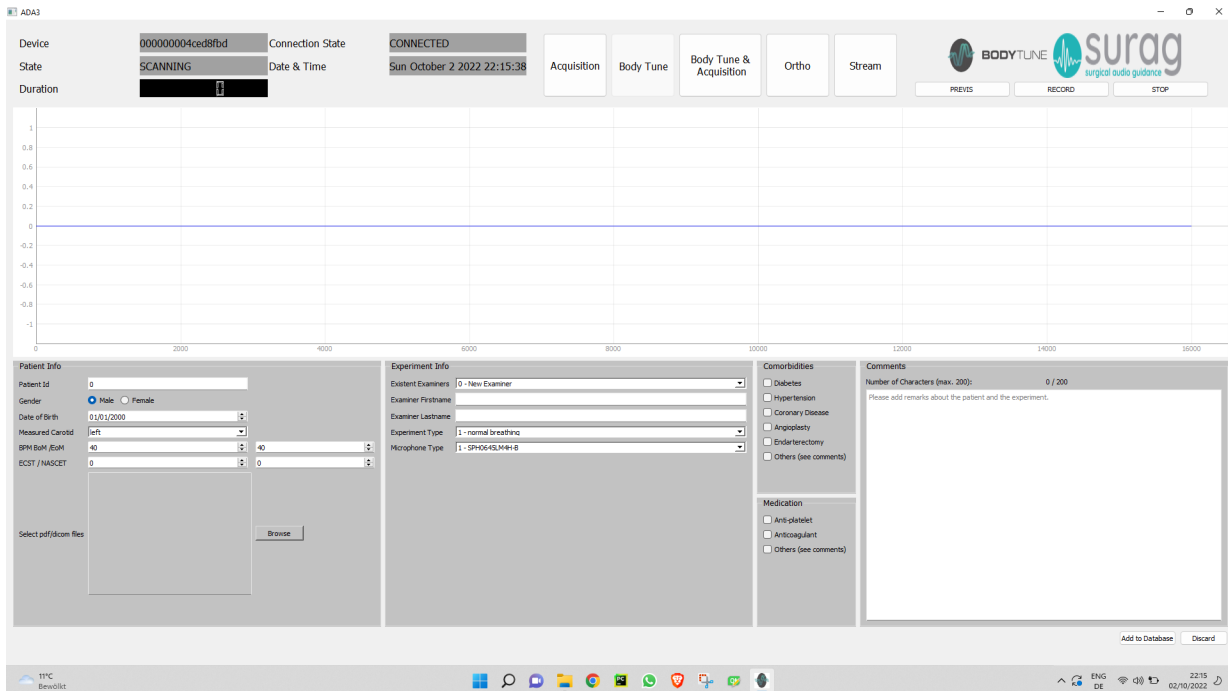
In this version to navigate between GUI's tab's (as shown in the image) has to be clicked to open a GUI. But to open a new GUI already opened GUI has to be closed, which is not ideal for user

Revision model 2:



In this version there is a common control center for all GUI's. The GUI's can be selected from a combo box. But by using the combo box seamless travel between GUI is hindered.

Finalized revision:



- A GUI which has a common control center where we can see all the details about the state of the device, connection state, Duration of the experiment, Date and time of experiment
- Using the GUI tab's, switch between GUI's is very easy
- After switching to GUI, all the features with respect that particular GUI can be viewed
- Can add the annotation details of the patient, examiner, experiment, medication, comorbidities etc.
- Can visualize the audio data from the body tune device
- Add new files
- Button to save the information given in a particular GUI
- Button to clear the data entered
- Buttons to control the Bodytune device

Apart from the above changes, few more updates which are made in this version are detailed in the requirement section under mandatory changes please go through them.

Finally, a GUI is developed which provides access to all the different versions of the GUI's and also provided with new additional functionalities.

The developed version of the GUI can only be used by a user when they can download and install all the tools which were mentioned in the development section of the GUI and it is not user friendly. So, we need to create an application installer which install all the packages necessary for the proper working of the application.

MSI Installer:

Installer files are used by Microsoft's Windows Operating System for installing software. It is a package with installation information for a particular program. The package includes details about the files to be installed as well as the specific locations to where the file is to be installed.

In creating an installer, we followed two steps:

1. Creating an executable file
2. Creating an installer from the executable

Creating an executable file:

An executable file (EXE file) is a computer file that contains an encoded sequence of instructions that the system can execute directly when the user clicks the file icon.

Tool's used: Auto PY to EXE, PyInstaller

Auto PY to EXE:

Version: **2.23.1**

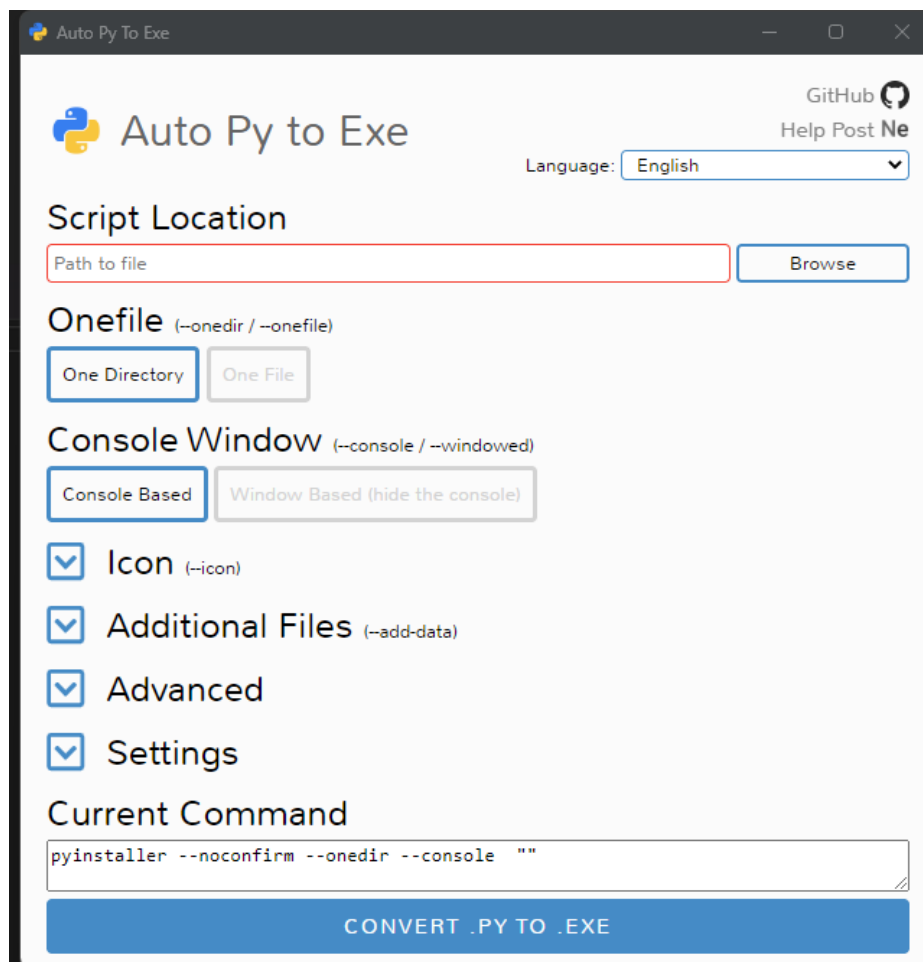
Download: pip install auto-py-to-exe

A .Py to .exe converter using a simple graphical interface and Pyinstaller in Python

This tool provides a GUI to select all necessary files, folders, dependencies and icons which are necessary to create a new executable.

We made a folder based executable as it is easier to load and easy to debug for any issues.

This tool also helps you in debugging the issues while the executable is run by the system.



Link provides reference to how Auto-Py-to-Exe GUI looks after adding all the files and folders

Link: <https://www.dropbox.com/s/ekxd3ivbis70tbn/Auto%20Py%20To%20Exe.pdf?dl=0>

Auto Py to exe documentation and help link:

https://nitratine.net/blog/post/issues-when-using-auto-py-to-exe/?utm_source=auto_py_to_exe&utm_medium=application_link&utm_campaign=auto_py_to_exe_help&utm_content=bottom

PyInstaller:

Version: **5.4.1**

Installation: pip install pyinstaller

PyInstaller bundles a Python application and all its dependencies into a single package. The user can run the packaged app without installing a Python interpreter or any modules.

PyInstaller reads a Python script. It analyzes the code to discover every other module and library the script needs in order to execute. Then it collects copies of all those files – including the active Python interpreter! – and puts them with the script in a single folder, or optionally in a single executable file

Auto Py to Exe converts the python file using pyinstaller. The Json file below shows the background implementation.

Link: https://www.dropbox.com/s/arjo7oxcg9gnreb/installer_folder.json?dl=0

Creating an installer from the executable:

Tool used: Inno setup

Download link: <https://jrsoftware.org/isdl.php>

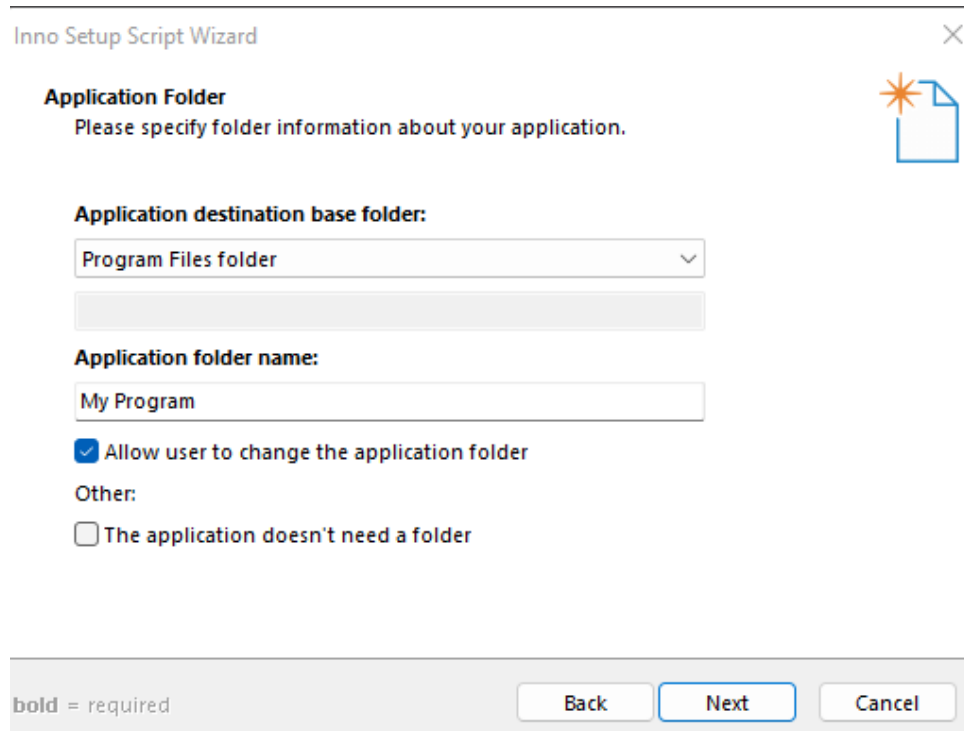
Inno setup:

Version: 6.2.1

Inno setup is an installer for windows.

Can be used to make executables of different formats

The folder-based executable is now converted into a single file-based executable along with additional information on how to unpack the executable and where to install the application and few other details like application name, version, desktop shortcut, license file, icon etc.



The screenshot shows the 'Inno Setup Script Wizard' window, specifically the 'Application Folder' step. The window has a title bar with the text 'Inno Setup Script Wizard' and a close button. Below the title bar, the text 'Application Folder' is followed by the instruction 'Please specify folder information about your application.' To the right of this text is an icon of a document with a star. The main area contains three sections: 'Application destination base folder:' with a dropdown menu showing 'Program Files folder' and a disabled text box below it; 'Application folder name:' with a text box containing 'My Program'; and a checkbox labeled 'Allow user to change the application folder' which is checked. Below this is the text 'Other:' followed by a checkbox labeled 'The application doesn't need a folder' which is unchecked. At the bottom, there is a legend 'bold = required' and three buttons: 'Back', 'Next' (which is highlighted with a blue border), and 'Cancel'.

Inno Setup Script Wizard

Application Folder
Please specify folder information about your application.

Application destination base folder:
Program Files folder

Application folder name:
My Program

☒ Allow user to change the application folder

Other:
☐ The application doesn't need a folder

bold = required

Back Next Cancel

Using Auto-Py-to-Exe and Inno setup an installer file was made which can be installed in a 64-bit operating system (windows 10/11), x64-based processor.

Now, the user can install the application in the system and after installation just by clicking on application icon can launch and start using the GUI and its features.

Dual Communication:

The auscultation device and the desktop application exchange control commands and real-time audio data via a local Wi-Fi network which is hosted by the device running the desktop application.

The communication between the two devices happens using Transmission Control Protocol (TCP) and socket communication.

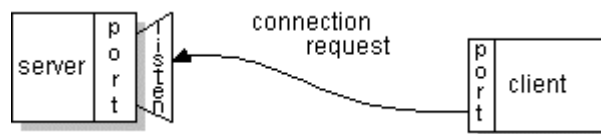
Transmission Control Protocol (TCP) is a communication standard that enables application programs and computing devices to exchange messages over a network. TCP guarantees the integrity of the data being communicated between a server and a client over a network.

Socket communication:

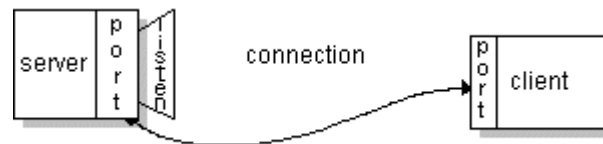
A socket is one endpoint (IP address and a port number) of a two-way communication link between two programs (server and client) running on the network.

<https://docs.oracle.com/javase/tutorial/networking/sockets/definition.html#:~:text=A%20socket%20is%20one%20endpoint,address%20and%200a%20port%20number.>

A server runs on a specific computer and has a socket that is bound to a specific port number. The server just waits, listening to the socket for a client to make a connection request.



A client knows the hostname of the machine on which the server is running and the port number on which the server is listening. To make a connection request, the client tries to engage with the server on the server's machine and port. The client also needs to identify itself to the server so it binds to a local port number (assigned by the system) that it will use during this connection.



Socket programming overview

<https://docs.python.org/3/howto/sockets.html>

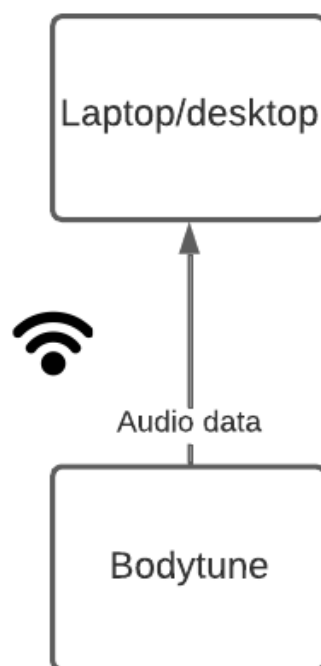
<http://thezanshow.com/electronics-tutorials/raspberry-pi/tutorial-27-28-29>

Initial setting:

In the initial version of the bodytune, when the device enters visualization state the socket starts receiving data from the auscultation device and a real-time visualization of the acquired audio signal starts in the application.

For changing the states between visualization, record and stop buttons on the bodytune are used.

The desktop application provides the visualization of the sound signal.



Dual communication setting:

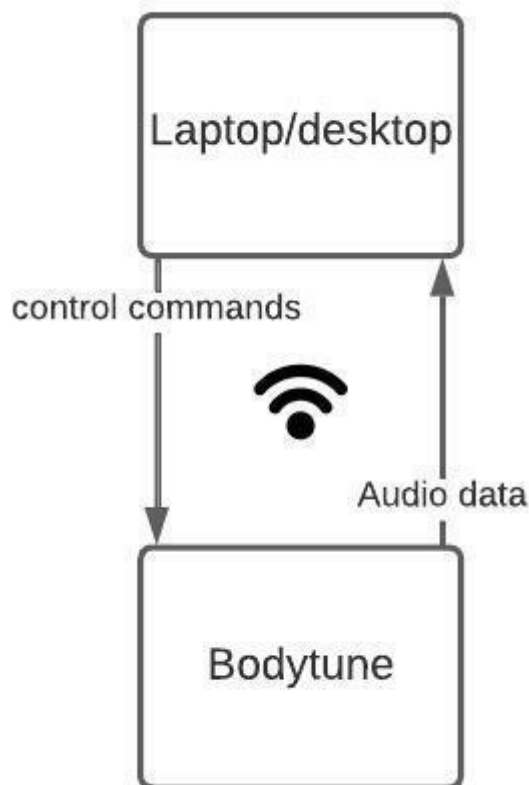
Bodytune device is developed for a user to use it all my himself and during such examination a user may not be able to access the buttons or sometimes presses the wrong button. To resolve this issue control buttons were introduced in the application.

Controlling bodytune device from the application: The application GUI consists of buttons (see **image**) when pressed sends the control commands (Visualization, Record, Stop) to the bodytune through a socket.



Real-time visualization:

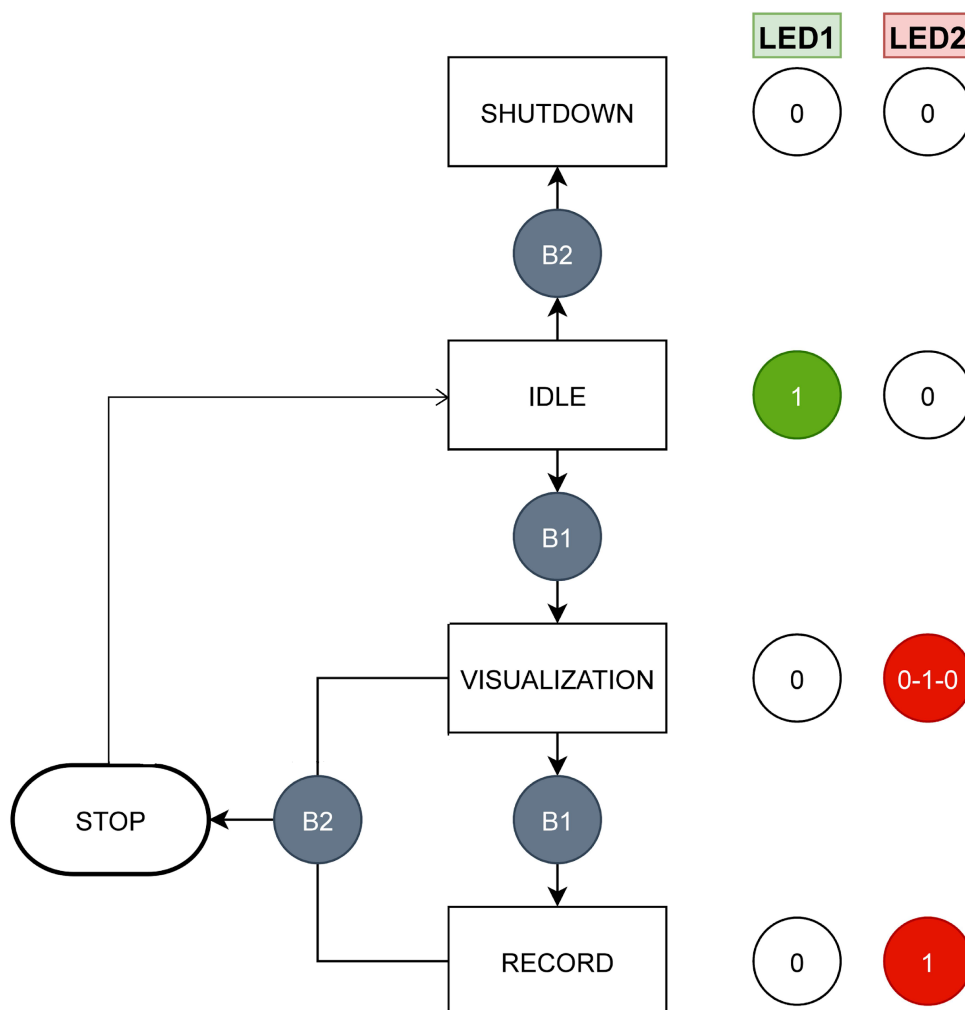
After receiving the commands from the application, the device then changes the states accordingly and streams the acquired audio data through a different socket to the application for real time visualization.



LED Implementation:

We will now see the implementation of LED concept in the Bodytune Device to know the status of the device just by seeing the LEDs.

State	Button activity	Led1	Led2
Visualization	Button-1 press once	off	Blinking
Record	Button-1 press twice	off	On
Stop	Button 2 press	Blinking(2 times)	Blinking(2 times)
Idle	-	On	off
Shutdown	Button 2 pressed twice	Off	Off



The bodytune stream audio data via Wi-Fi to a computer. The functionality of the bodytune is controlled via two buttons, and the two Led's provide feedback about its current state. Figure 1 provides an overview about the button activity and the respective LED feedback.

- When in idle state LED1 is ON and LED2 is OFF.
- If the Button B1 is pressed once LED2 starts blinking indicating the Visualization state.
- If the Button B1 is pressed again LED2 is ON without blinking indicating the Record state.
- If the Button B2 is pressed once both LED1 and LED2 blinks twice
- After the stop state the bodytune device go into idle state which is indicated by LED1 ON
- Finally, if Button B2 is double clicked both LEDs turn off indicating the device shutdown.

Database:

The audio streamed from the bodytune device has to be saved along with the details that are entered in the annotation widget of the application GUI like patient information, examiner information and experiment information, medication and comorbidities.

Initially, there was no common database to save the annotation details and the audio data from all the different GUI's. We created a new database system which stores the data locally in a file format.

When we press the “add to database” button in any of the GUI's a folder by the name “Database” is created in the application installation folder (if there is no such folder already present in the installation directory).

Subsequently, new folders are created inside the “Database” directory corresponding to each GUI.

Bodytune GUI: When “add to database” is pressed in Bodytune GUI a directory by the name “Bodytune” is created inside the “Database” folder and inside the “Bodytune” folder a new folder is created each time with the patient-id. Inside the directory with the patient id, we save the patient information, examiner information and experiment information, medication and comorbidities in a text document format (.txt) and we save the audio data streamed as a wave file format (.wav). Additionally, files (PDF/DICOM) which are attached in the annotation widget are also saved in this folder.

Ortho GUI: Similarly, to bodytune GUI, folders are created inside the “Database” directory. When “add to database” is pressed “Ortho” folder is created inside the Database folder furthermore new folder with respect to new patient with the patients (first name and last name) are created inside the “Ortho” folder in which we save the text file(.txt), wave file (.wav) and additional files (PDF/DICOM).

Bodytune and Acquisition GUI: Similarly, to bodytune GUI, folders are created inside the “Database” directory. When “add to database” is pressed “BodytuneAcquisition” folder is created inside the Database folder. Furthermore, new folders are created with respect to each new examiner name, inside which a new folder is created with respect to each patient id. In these folders the text file(.txt), wave file (.wav) and additional files (PDF/DICOM) are saved.

Picture below shows the folder structure in the “Database” folder

```
C:\Users\harsh\Desktop\ADA3\Database>tree /f
Folder PATH listing for volume OS
Volume serial number is 86C4-1288
C:..
├──Bodytune
│   └──BT-196
│       ADA3 Project abstract.pdf
│       BT-196-2022-10-02_22-32-13.txt
│       TMP-196-2022-10-02_22-32-13.wav
├──BodytuneAcquisition
│   └──harsha
│       └──BA-198
│           ADA3 Project abstract.pdf
│           BA-198-2022-10-02_22-35-23.txt
│           TMP-198-2022-10-02_22-35-23.wav
└──Ortho
    └──OR-harshavardhan
        ADA3 Project abstract.pdf
        OR-harshavardhan-2022-10-02_22-37-15.txt
        TMP-harshavardhan-2022-10-02_22-37-15.wav

C:\Users\harsh\Desktop\ADA3\Database>
```

- Every text document, wave file is created with a unique name.

Ex:BA-198-2022-10-02_22-35-23

Where,

BA-198 - patient id

2022-10-02-Date of Measurement (YYYY-MM-DD)

22-35-23- Time of measurement (Hour-Minute-Sec)

Annotation data which is being saved and their type

- Patient details
 - o ID: Alphanumeric
 - o First name: string
 - o Last name: string
 - o Gender: string (Male/female)
 - o date of birth: Date (01/01/2000)
 - o Measured Carotid: string (Left/Right)
 - o BPM BoM/Eom: int
 - o ECST/NASCET: int
 - o Replacement: string (Partially/full)
- Examiner details
 - o First name: String
 - o Last name: String
 - o Existent examiner: String
- Measurement information:
 - o Experiment type: String
 - o Microphone type: Alphanumeric
- Comorbidities:
 - o diabetes: Boolean
 - o hypertension: Boolean
 - o Coronary Disease: Boolean
 - o Angioplasty: Boolean
 - o Endarterectomy: Boolean
 - o Others: Boolean
- Medication:
 - o Anti-platelet: Boolean
 - o Anticoagulant: Boolean
 - o Others: Boolean
- Comments: String

Application Testing Cases:

Here we will see the testing of our application we developed against different test cases we made keeping in mind the requirements of the client and user usage methods. The testing cases are done to make sure we developed an application that is fully functional and bug free. We are focused on delivering an application which is safe and user friendly.

These test cases are based on some of the problems we faced while developing the application and we want the users to not face any of these issues at any instance.

Testing Cases:

Test Case 1: When control buttons “stop” pressed after “previz” led to application crash

Result: Now the application follows a sequential step where after previz you are only able to record and then move to stop.

Status: Pass

Test Case 2: Changing the GUI tabs from one tab to another while the record is going on.

Result: It should not switch between tabs while on record mode.

Status: Pass

Test Case 3: Pressing add to database button while in record is leading to application crash.

Result: Dialog prompt notifying the user to stop the recording first.

Status: Pass

Test Case 4: The patient id should be more than 2 numbers and the files are to be saved with that patient id.

Result: The patient id is to be alpha numeric value and the file is saving based on the patient id.

Status: Pass

Test Case 5: Test if the experiments are auto starting without any input.(Bodytune GUI)

Result: The experiments should only start from normal breathing after input and should change to a new experiment every 12 Secs automatically.

Status: Pass

Test Case 6: Test the experiment running time for visualization/record.

Result: The experiments will run for 180 Secs on visualization mode and 60 Secs on record mode and will stop automatically.

Status: Pass

Test Case 7: Test if the LEDs on the bodytune device show the status of the device.

Result: The LEDs on the device should show the Previs, Record and Stop mode status.

Status: Pass

Test Case 8: Test if the dual communication between application and bodytune device are working irrespective of which program starts first.

Result: It is connecting without any problem.

Status: Pass