A PROJECT REPORT
ON
SCREEN RECORDING WITH MULTIPLE OPTIONS

SUBMITTED IN PARTIAL FULFILLMENT OF THE
REQUIREMENT FOR THE DIPLOMA IN

COMPUTER ENGINEERING

BY

Roll no.	Name
60	GODSE VINAYAK H.
41	MULANI HAMEED K.
12	ROKADE MANASI M.
29	CHIMANE SUJATA N.

UNDER
THE GUIDENCE
Pro. Prajakta Mahajan

GOVERNMENT POLYTECHNIC, THANE
Phadke pada, Opp. Bharat Gears, Mumbra-shil road, Mumbra, thane 400612
Website: www.gpthane.org.in Email: gpthane@yahoo.co.in 022-25493329

GOVERNMENT POLYTECHNIC, THANE

Phadke pada, Opp. Bharat Gears, Mumbra-shil road, Mumbra, thane 400612

Website: www.gpthane.org.in Email: gpthane@yahoo.co.in 022-25493329

Certificate

This is to certify that following third year computer engineering student have successfully and satisfactorily completed their project, entitled "Screen Recorder with Multiple Options", in partial fulfillment of the requirement for the diploma in Computer Engineering for the academic year 2020-2021.

Roll no.	Name
60	GODSE VINAYAK H.
41	MULANI HAMEED K.
12	ROKADE MANASI M.
29	CHIMANE SUJATA N.

PROJECT GUIDE
(Pro.Prajakta Mahajan)

H.O.D
(Pro.Dipali P. Sapkal)

(External Examiner)

Principle
(Dr.Nitiket N. Mhala)

ACKNOWLEDGEMENT

We take it is an opportunity to thank all those who have directly and indirectly inspired, directed and assisted us towards successful completion of this project report.

We express our sincere thanks to the principal, **Dr. Prof. Mhala Sir** and the our guide **Mrs. Mahajan P. S. madam and Chate Vaishal madam** for having us allowed to submit this report as a part of our academic learning.

We express our sincere thanks to **Mrs. Mahajan. P. S. Madam,** for encouragement throughout the project report and guideline in designing and working out this project.

We are also grateful to the team of **"Screen Recording with multiple options"** their highly encouraging and co-operative attitude we express our sense of gratitude toward our friends and parents for their constant and moral support during project report.

Place: Government Polytechnic, Thane.

Date: / /2021

Yours sincerely, TY CO,

Godse Vinayak H.

Mulani Hameed K.

Rokade Manasi M.

Chimane Sujata N.

CERTIFICATE

This is to certify that as a part of the partial fulfillment of the fifth semester of diploma in computer engineering, following students

Godse Vinayak H.

Mulani Hameed K.

Rokade Manasi M.

Chimane Sujata N.

Has successfully completed the Micro Project "**Screen Recording with multiple options"** for the subject **"Capstone Project"** under guidance of **Mrs. Mahajan P. S. Madam** And submitted it to computer Engineering department of Government polytechnic, Thane.

Date: / /2021

Place: Government polytechnic, Thane.

Prof. Dipali P. Sapkal — **(**Head of Department)

Prof. P.S. Mahajan — (Project Guide)

Abstract:

With the rise in Online as a platform, many teachers, gamers, programmers are adopting the robust, secure and agile software to share, help and grow as a person and together. It's time to ditch skype and zoom, to handle the guilds the owners often get overwhelmed by how to handle the guilds, and this is where our project comes in. It helps the guild owners and developers both of them, help invite bots according to their needs and to give developers a platform to showcase their programming skills. Powered by latest trend in web technologies we made it possible for users and developers to finally take stand and join the discord revolution,

Screen Recording System is one of the most innovative ideas of this era. It will make a person's life easier in a dramatic way. If a person wishes to record the screen, that person just has to go to this system, select the recording option and start recording and click on stop recording when he wants to stop recording and click on download button and BAM!, the recorded video is started to downloading on user's system.

Contents

Chapter 1:

Introduction:

Background of Industry:

The “Screen Recording with multiple option” operation is focused on Recording Screen with different option like with mic, with camera, with internal audio etc. The entire purpose of Screen Recording with multiple option web app is to give the platform to users to record is desktop screen with multiple options. Quick and easy recording process user can there screen with camera as well user can record the screen with external voice (mic) and after recording complete user able to download that recording. App based on screen recording is one of the revolutions of modern world. Now days people wants to record their desktop screen with camera or with external audio (mic) so this web app helps that user to record the desktop screen without downloading any on his machine. Screen recording application helps you to record the video of your work with your sound for clarification. Also user can there educational lecture with mic through this app in easy and quiz way. With the technology evolving day by day Screen recording is a key part in present gamer streaming as well as in online education to record the game and lectures respectively.

Screen recording with multiple option is a web based application which enables users to record their own desktop screen for different perpous. The internet users are increasing rapidly the company has introduced screen recording system for getting the more traffic on their website. This application not only improves Users experiences but also its save the cost of video recording. Screen Recording System is one of the most innovative ideas of this era. It will make a person’s life easier in a dramatic way. If a person wishes to record their desktop screen, that person just has to go to this system, select the recording option, Record the screen stop the recording and download the recording and BAM!, the recording start downloading n their own system. It would be comfortable for the Users to have an Screen Recording system. It would be free for users, they can select screen recording application they want and record the screen. It will reduce the purchasing cost of the paid recording software. This Screen recording Web application will helps users in Recording Screen. This system will surely enhance the privacy of the user’s content.

Chapter 2:

Literature Survey:

We are developing Screen Recording with various options web application. This application is free for all also we will if user record the screen after recording complete he /she is able to download that recording by clicking on download button.

The project is to implement a Html , Css & JavaScript program for the Screen recording with various options. In this program the user is viable for capturing any task or window that is running on your desktop to recycle and use it for further edit and upload on social media. A screen recording software can be used while you are playing video games or when you are learning on a subject that may require a demonstration on it again soon.The main motive to create this project is , on the internet there are so many paid screen recording applications, and there are no various options to record screen , so we desided to develop a free screen recording application with various options.

Following options are in our screen recording project:-

1. **Screen recording**
2. **Screen recording with mic**
3. **Screen recording with internal audio**
3. **Screen recording with internal audio and camera**
4. **Screen recording with mic and camera**

Chapter 3:

Scope of the project:

1. Providing the screen recording platform to users.
2. Gets traffic to the website easily and quickly.
3. The scope of the project is the designing web interface and it will be used in future.
4. Helps users to recording screen with different options.
5. Helps users to download recorded screen.
6. Also save the time and cost of users.

Chapter 4:

Methodology:

Screen Recorder with Multiple options:

The project is to implement a Html, Css & JavaScript program for the Screen recording with various options. In this program the user is viable for capturing any task or window that is running on your desktop to recycle and use it for further edit and upload on social media. A screen recording software can be used while you are playing video games or when you are learning on a subject that may require a demonstration on it again soon.The main motive to create this project is , on the internet there are so many paid screen recording applications, and there are no various options to record screen, So we desided to develop a free screen recording application with various options.

MediaCapture Api:

The **Media Capture and Streams** API, often called the **Media Streams API** or **MediaStream API**, is an API related to WebRTC which provides support for streaming audio and video data. It provides the interfaces and methods for working with the streams and their constituent tracks, the constraints associated with data formats, the success and error callbacks when using the data asynchronously, and the events that are fired during the process.

Supported Languages (Libraries):

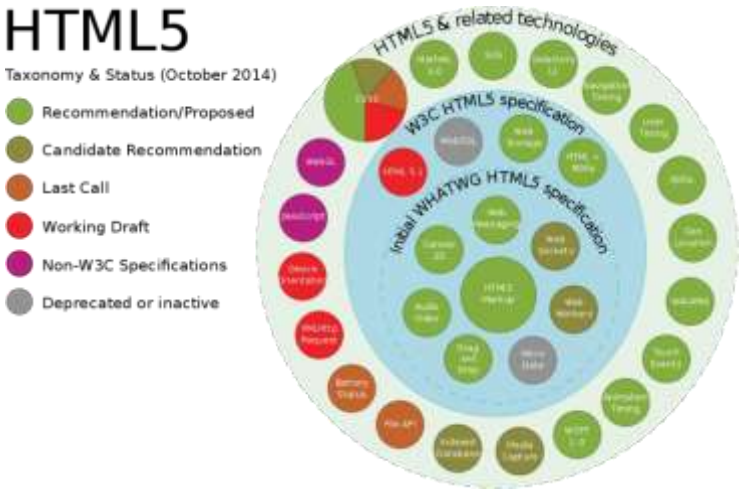
Most popular languages used in development:

- JavaScript
- PHP
- HTML
- CSS
- BootStrap

Html5:

HTML 5 introduces elements and attributes that reflect typical usage on modern websites. Some of them are semantic replacements for common uses of generic block ( ) and inline () elements, for example <nav> (website navigation block), <footer> (usually referring to bottom of web page or to last lines of HTML code), or <audio> and <video> instead of <object>. Some deprecated elements from HTML 4.01 have been dropped, including purely presentational elements such as <font> and <centre>, whose effects have long been superseded by the more capable Cascading Style Sheets. There is also a renewed emphasis on the importance of DOM scripting in Web behaviour.

The HTML 5 syntax is no longer based on SGML despite the similarity of its mark-up. It has, however, been designed to be backward-compatible with common parsing of older versions of HTML. It comes with a new introductory line that looks like an SGML document type declaration, <!DOCTYPE html>, which triggers the standards-compliant rendering mode. Since 5 January 2009, HTML 5 also includes Web Forms 2.0, a previously separate WHATWG (Web Hypertext Application Technology Working Group) specification



CSS3:

Cascading Style Sheets (CSS) is a language that is used to illustrate the look, style, and format of a document written in any mark-up language. In simple words, it is used to style and organize the layout of Web pages. CSS3 is the latest version of an earlier CSS version, CSS2.

A significant change in CSS3 in comparison to CSS2 is the introduction of modules. The benefit of this functionality is that it allows the specification to be finalized and accept faster, as segments are finalized and accepted in portions. Also, this allows the browser to support segments of the specification.

Some of the key modules of CSS3 are:

- Box model
- Image values and replaced content
- Text effects
- Selectors
- Backgrounds and borders
- Animations
- User interface (UI)
- Multiple column layouts

JavaScript:

JavaScript is a cross-platform, object-oriented scripting language used to make webpages interactive (e.g., having complex animations, clickable buttons, popup menus, etc.). There are also more advanced server side versions of JavaScript such as Node.js, which allow you to add more functionality to a website than downloading files (such as realtime collaboration between multiple computers). Inside a host environment (for example, a web browser), JavaScript can be connected to the objects of its environment to provide programmatic control over them.

JavaScript contains a standard library of objects, such as `Array`, `Date`, and `Math`, and a core set of language elements such as operators, control structures, and statements. Core JavaScript can be extended for a variety of purposes by supplementing it with additional objects; for example:

- *Client-side JavaScript* extends the core language by supplying objects to control a browser and its *Document Object Model* (DOM). For example, client-side extensions allow an application to place elements on an HTML form and respond to user events such as mouse clicks, form input, and page navigation.
- *Server-side JavaScript* extends the core language by supplying objects relevant to running JavaScript on a server. For example, server-side extensions allow an application

to communicate with a database, provide continuity of information from one invocation to another of the application, or perform file manipulations on a server.

This means that in the browser, JavaScript can change the way the webpage (DOM) looks. And, likewise, Node.js JavaScript on the server can respond to custom requests from code written in the browser.

PHP:

PHP started out as a small open source project that evolved as more and more people found out how useful it was. Rasmus Lerdorf unleashed the first version of PHP way back in 1994.

- PHP is a recursive acronym for "PHP: Hypertext Preprocessor".
- PHP is a server side scripting language that is embedded in HTML. It is used to manage dynamic content, databases, session tracking, even build entire e-commerce sites.
- It is integrated with a number of popular databases, including MySQL, PostgreSQL, Oracle, Sybase, Informix, and Microsoft SQL Server.
- PHP is pleasingly zippy in its execution, especially when compiled as an Apache module on the Unix side. The MySQL server, once started, executes even very complex queries with huge result sets in record-setting time.
- PHP supports a large number of major protocols such as POP3, IMAP, and LDAP. PHP4 added support for Java and distributed object architectures (COM and CORBA), making n-tier development a possibility for the first time.
- PHP Syntax is C-Like.

BOOTSTRAP:

Originally created by a designer and a developer at Twitter, Bootstrap has become one of the most popular front-end frameworks and open source projects in the world.

Bootstrap was created at Twitter in mid-2010 by @mdo and @fat. Prior to being an open-sourced framework, Bootstrap was known as *Twitter Blueprint*. A few months into development, Twitter held its first Hack Week and the project exploded as developers of all skill levels jumped in without any external guidance. It served as the style guide for internal tools development at the company for over a year before its public release, and continues to do so today.

Originally released on August 19, 2011, we've since had over twenty releases, including two major rewrites with v2 and v3. With Bootstrap 2, we added responsive functionality to the entire framework as an optional stylesheet. Building on that with Bootstrap 3, we rewrote the library once more to make it responsive by default with a mobile first approach.

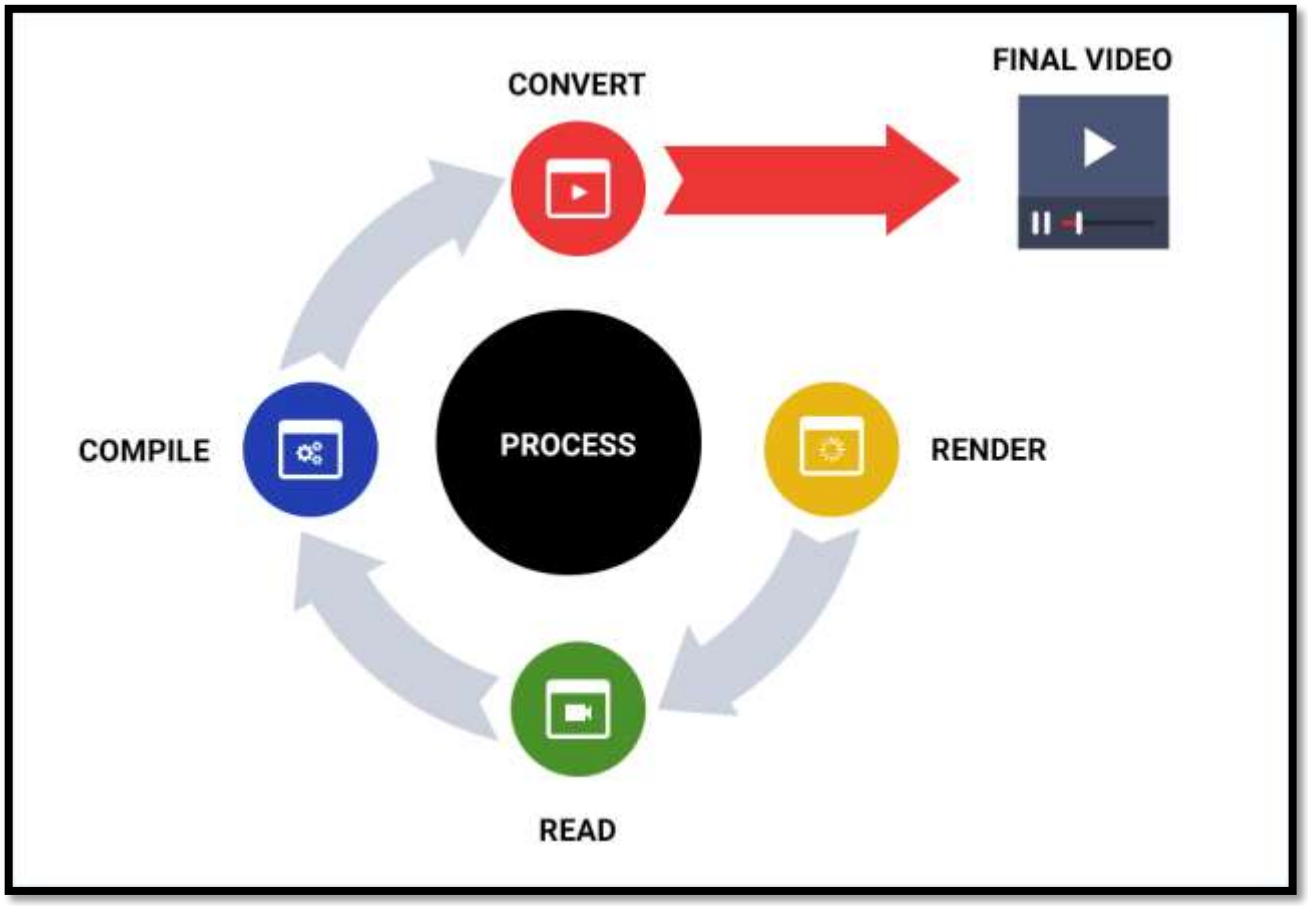
With Bootstrap 4, we once again rewrote the project to account for two key architectural changes: a migration to Sass and the move to CSS's flexbox. Our intention is to help in a small way to move the web development community forward by pushing for newer CSS properties, fewer dependencies, and new technologies across more modern browsers.

Chapter 5:

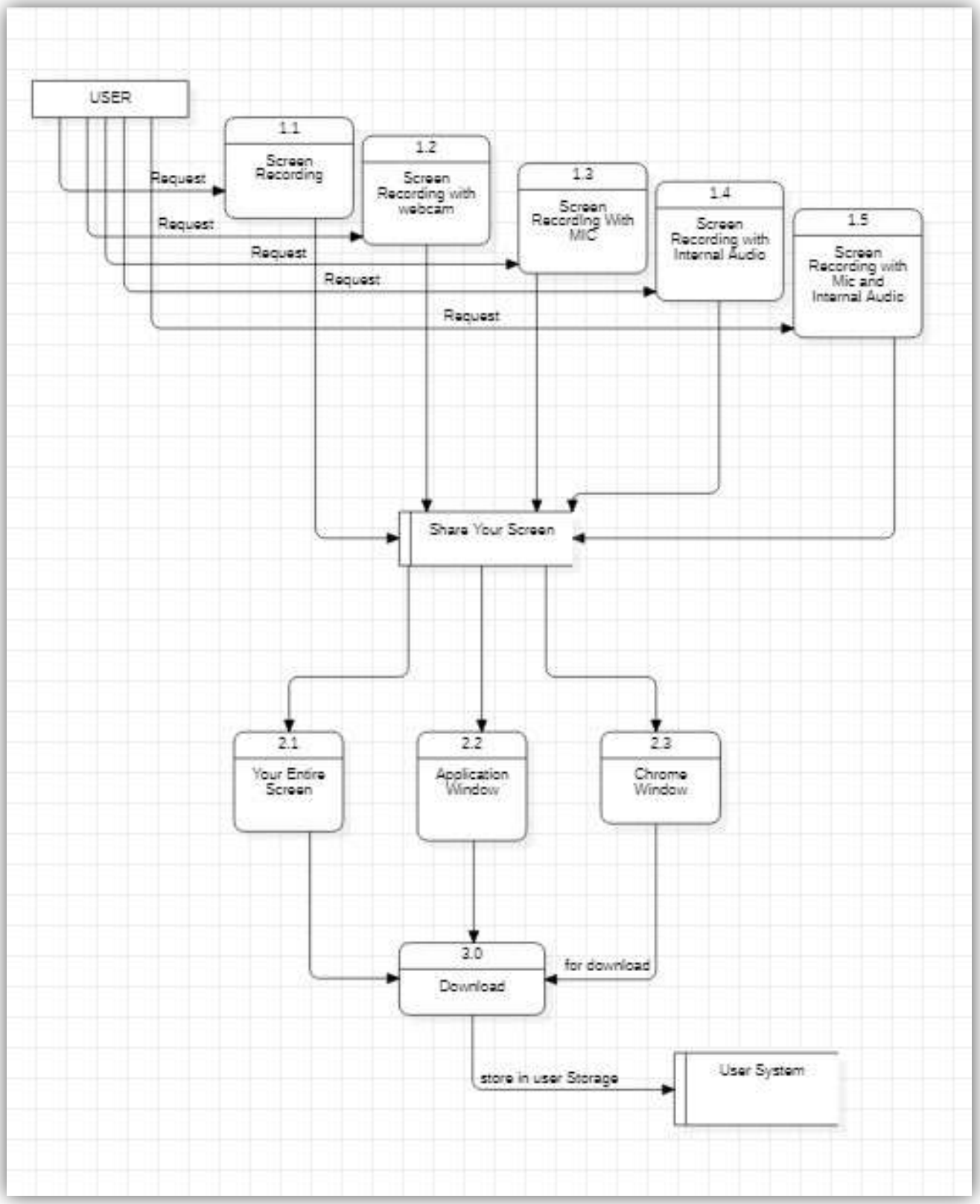
Details of design, working and processes

To make things easier, and remaining productive as possible, we decided to distribute our work between team members and they'd report after every sprint. Furthermore, **Vinayak Godse** and **Hameed Mulani** teamed up for Backend development. Now, after explaining how Screen Recording Software **works** overhead, we'll move to individual Functioning development.

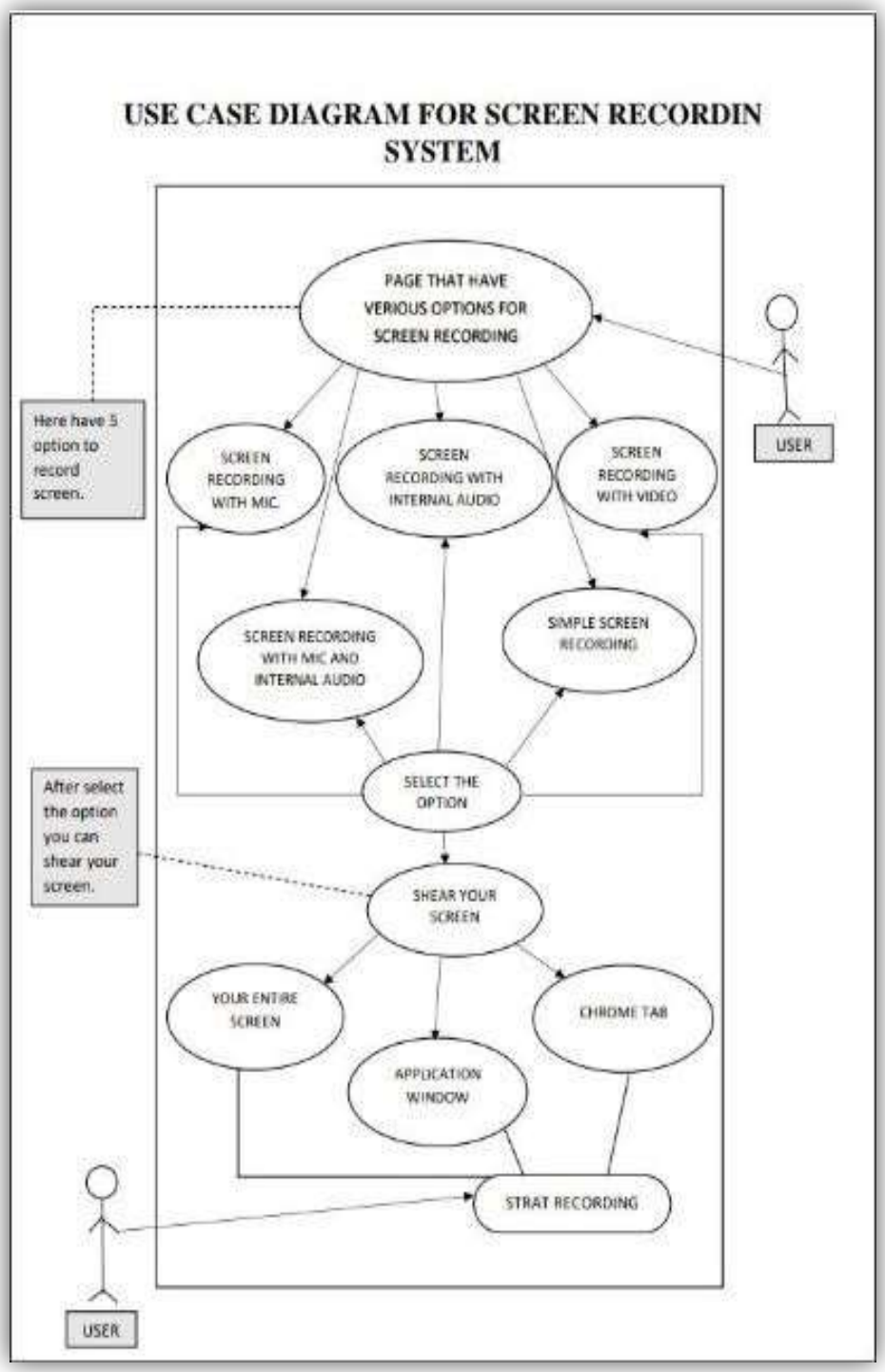
As we've discussed in earlier sections about website and overall information on **Capturing Screen**, Its time take it one step further and that is get an in-depth view on working behind the scene.

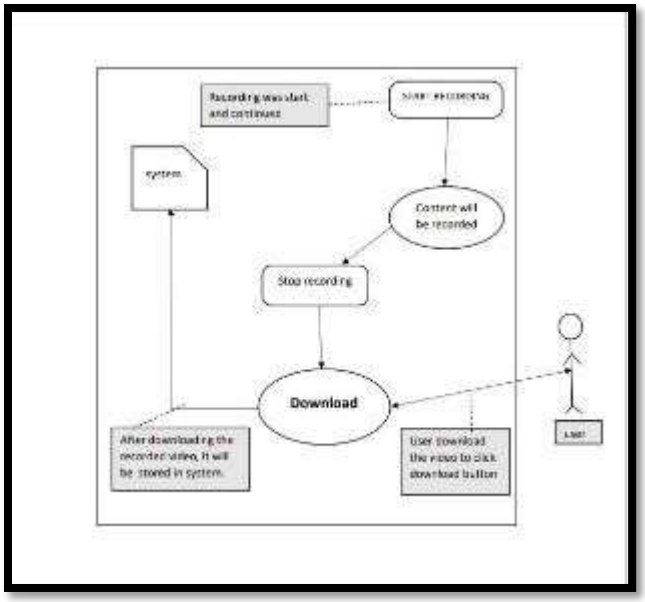


Data Flow **Diagram for Screen Recording**:

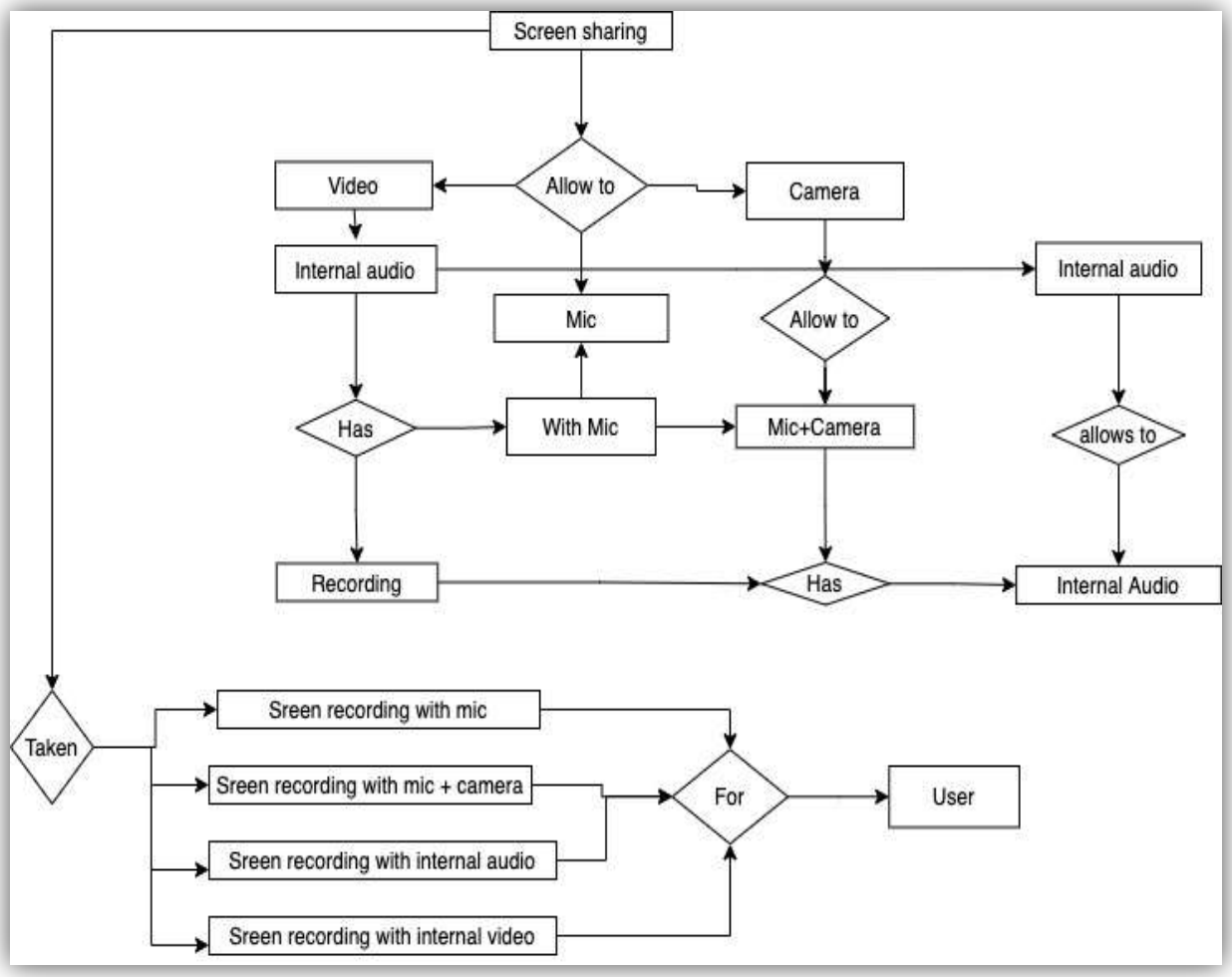


Use Case Diagram:




Content will be recorded
Stop recording
Download

ER Diagram:


Screen sharing
Video
Allow to
Camera
Internal audio
Internal audio
Mic
Allow to
Has
With Mic
Mic+Camera
allows to
Recording
Has
Internal Audio
Taken
Sreen recording with mic
Sreen recording with mic + camera
Sreen recording with internal audio
Sreen recording with internal video
For
User

Using the Screen Capture API

In this article, we will examine how to use the Screen Capture API and its getDisplayMedia() method to capture part or all of a screen for streaming, recording, or sharing during a WebRTC conference session.

It may be useful to note that recent versions of the WebRTC adapter.js shim include implementations of getDisplayMedia() to enable screen sharing on browsers that support it but do not implement the current standard API. This works with at least Chrome, Edge, and Firefox.

Capturing screen contents

Capturing screen contents as a live MediaStream is initiated by calling navigator.mediaDevices.getDisplayMedia(), which returns a promise that resolves to a stream containing the live screen contents.

Starting screen capture: async/await style

```
async function startCapture(displayMediaOptions) {

  let captureStream = null;

  try {    captureStream = await

navigator.mediaDevices.getDisplayMedia(displayMediaOptions);

  } catch(err) {

    console.error("Error: " + err);

  }

  return captureStream;

}
```

You can write this code either using an asynchronous function and the await operator, as shown above, or using the Promise directly, as seen below.

Starting screen capture: *Promise* style

```
function startCapture(displayMediaOptions) {

 return navigator.mediaDevices.getDisplayMedia(displayMediaOptions)

    .catch(err => { console.error("Error:" + err); return null; });

}
```

Either way, the user agent responds by presenting a user interface that prompts the user to choose the screen area to share. Both of these implementations of startCapture() return the MediaStream containing the captured display imagery.

For the purposes of the Screen Capture API, a **display surface** is any content object that can be selected by the API for sharing purposes. Sharing surfaces include the contents of a browser tab, a complete window, all of the windows of an application combined into a single surface, and a monitor (or group of monitors combined together into one surface).

There are two types of display surface. A **visible display surface** is a surface which is entirely visible on the screen, such as the front most window or tab, or the entire screen.

A **logical display surface** is one which is in part or completely obscured, either by being overlapped by another object to some extent, or by being entirely hidden or offscreen. How these are handled by the Screen Capture API varies. Generally, the browser will provide an image which obscures the hidden portion of the logical display surface in some way, such as by blurring or replacing with a color or pattern. This is done for security reasons, as the content that cannot be seen by the user may contain data which they do not want to share.

A user agent might allow the capture of the entire content of an obscured window after gaining permission from the user to do so. In this case, the user agent may include the obscured content, either by getting the current contents of the hidden portion of the window or by presenting the most-recently-visible contents if the current contents are not available.

Options and constraints

The constraints object passed into getDisplayMedia() is a DisplayMediaStreamConstraints object which is used to configuring the resulting stream.

Unlike most uses of constraints in media APIs, here it's solely used to define the stream configuration, and *not* to filter the available choices.

See Properties of shared screen tracks for details about additional constraints for configuring a screen-capture stream that are added to MediaTrackConstraints, MediaTrackSupportedConstraints, and MediaTrackSettings).

None of the constraints are applied in any way until after the content to capture has been selected. The contraints alter what you see in the resulting stream.

For example, if you specify a width constraint for the video, it's applied by scaling the video after the user selects the area to share. It doesn't establish a restriction on the size of the source itself.

Constraints *never* cause changes to the list of sources available for capture by the Screen Sharing API. This ensures that web applications can't force the user to share specific content by restricting the source list until only one item is left.

While display capture is in effect, the machine which is sharing screen contents will display some form of indicator so the user is aware that sharing is taking place.

For privacy and security reasons, screen sharing sources are not enumerable using enumerateDevices(). Related to this, the devicechange event is never sent when there are changes to the sources available for getDisplayMedia().

Capturing shared audio

getDisplayMedia() is most commonly used to capture video of a user's screen (or parts thereof). However, user agents may allow the capture of audio along with the video content. The source of this audio might be the selected window, the entire computer's audio system, or the user's microphone (or a combination of all of the above).

Before starting a project that will require sharing of audio, be sure to check the Browser compatibility in MediaDevices.getDisplayMedia() to see if the browsers you wish compaibility with have support for audio in captured screen streams.

To request that the screen be shared with included audio, the options passed into getDisplayMedia() might look like this:

```
const gdmOptions = {

  video: true,

  audio: true

}
```

This allows the user total freedom to select whatever they want, within the limits of what the user agent supports. This could be refined further by specifying additional information for each of audio and video:

```
const gdmOptions = {

  video: {

    cursor: "always"

  },

  audio: {

    echoCancellation: true,

    noiseSuppression: true,

    sampleRate: 44100}

}
```

In this example the cursor will always be visible in the capture, and the audio track should ideally have noise suppression and echo cancellation features enabled, as well as an ideal audio sample rate of 44.1kHz.

Capturing audio is always optional, and even when web content requests a stream with both audio and video, the returned MediaStream may still have only one video track, with no audio.

Starting display capture

The startCapture() method, below, starts the capture of a MediaStream whose contents are taken from a user-selected area of the screen. startCapture() is called when the "Start Capture" button is clicked.

```
async function startCapture() {

  logElem.innerHTML = "";



  try {

    videoElem.srcObject = await
navigator.mediaDevices.getDisplayMedia(displayMediaOptions);

    dumpOptionsInfo();

  } catch(err) {

    console.error("Error: " + err);

  }

}
```

After clearing the contents of the log in order to get rid of any leftover text from the previous attempt to connect, startCapture() calls getDisplayMedia(), passing into it the constraints object defined by displayMediaOptions. Using await, the following line of code does not get executed until after the promise returned by getDisplayMedia() resolves. Upon resolution, the promise returns a MediaStream, which will stream the contents of the screen, window, or other region selected by the user.

The stream is connected to the <video> element by storing the returned MediaStream into the element's srcObject.

The dumpOptionsInfo() function—which we will look at in a moment—dumps information about the stream to the log box for educational purposes.

If any of that fails, the catch() clause outputs an error message to the log box.

Stopping display capture

The stopCapture() method is called when the "Stop Capture" button is clicked. It stops the stream by getting its track list using MediaStream.getTracks(), then calling each track's stop() method. Once that's done, srcObject is set to null to make sure it's understood by anyone interested that there's no stream connected.

```
function stopCapture(evt) {

  let tracks = videoElem.srcObject.getTracks();



  tracks.forEach(track => track.stop());

  videoElem.srcObject = null;

}
```

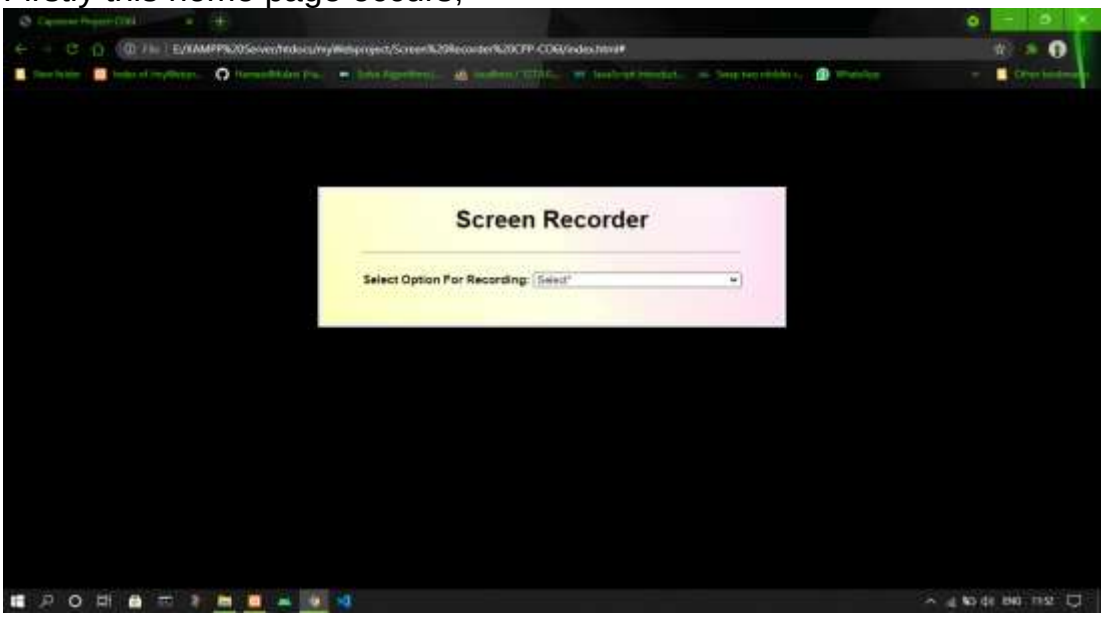
The track list is obtained by calling getVideoTracks() on the capture'd screen's MediaStream. The settings currently in effect are obtained using getSettings() and the established constraints are gotten with getConstraints()

Chapter 6:

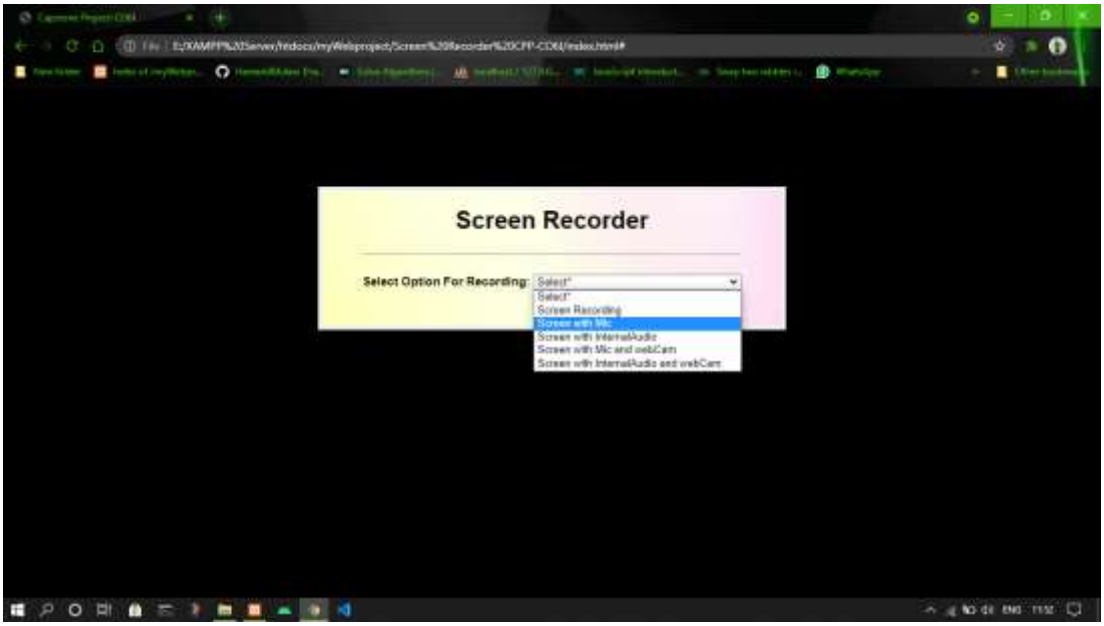
Results and applications

Home page Results:

Firstly this home page occurs,



The user want to select recording option from select box.



Select box to **select** option,

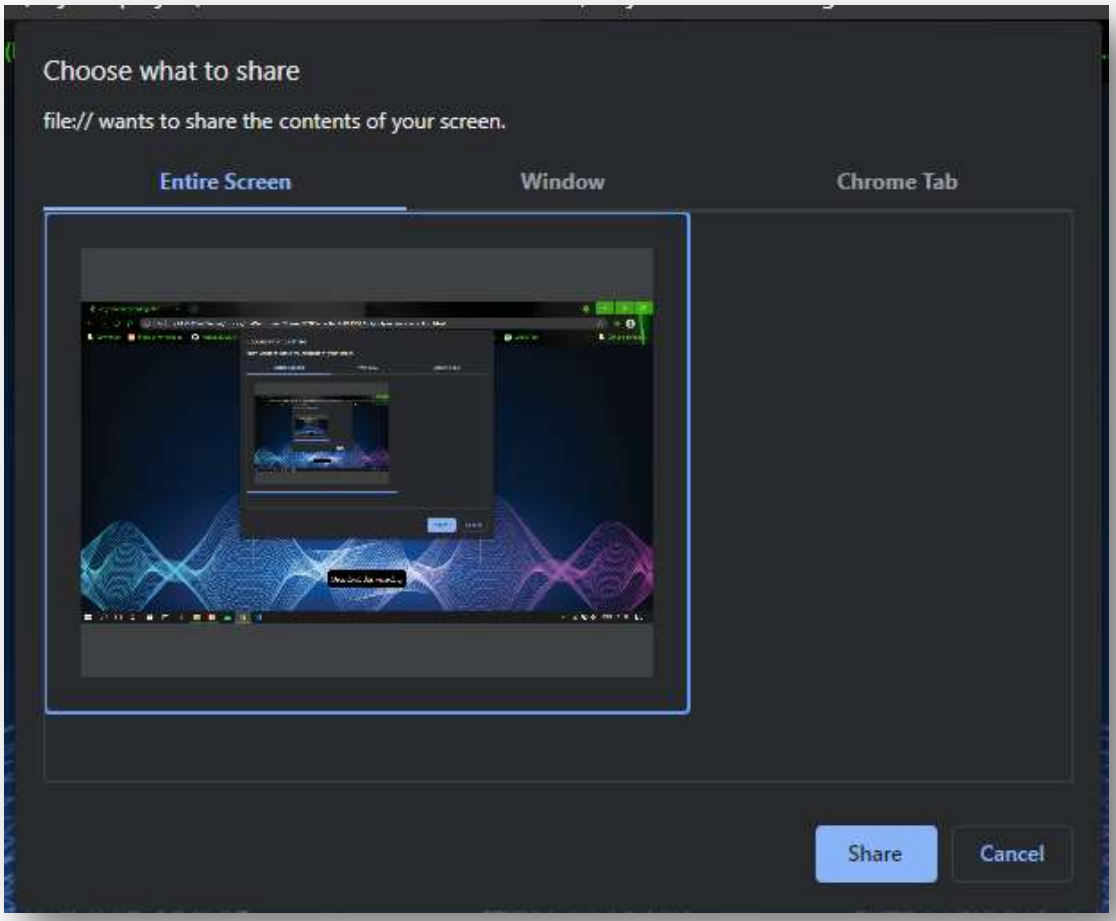
Above options in our screen recording project are:-

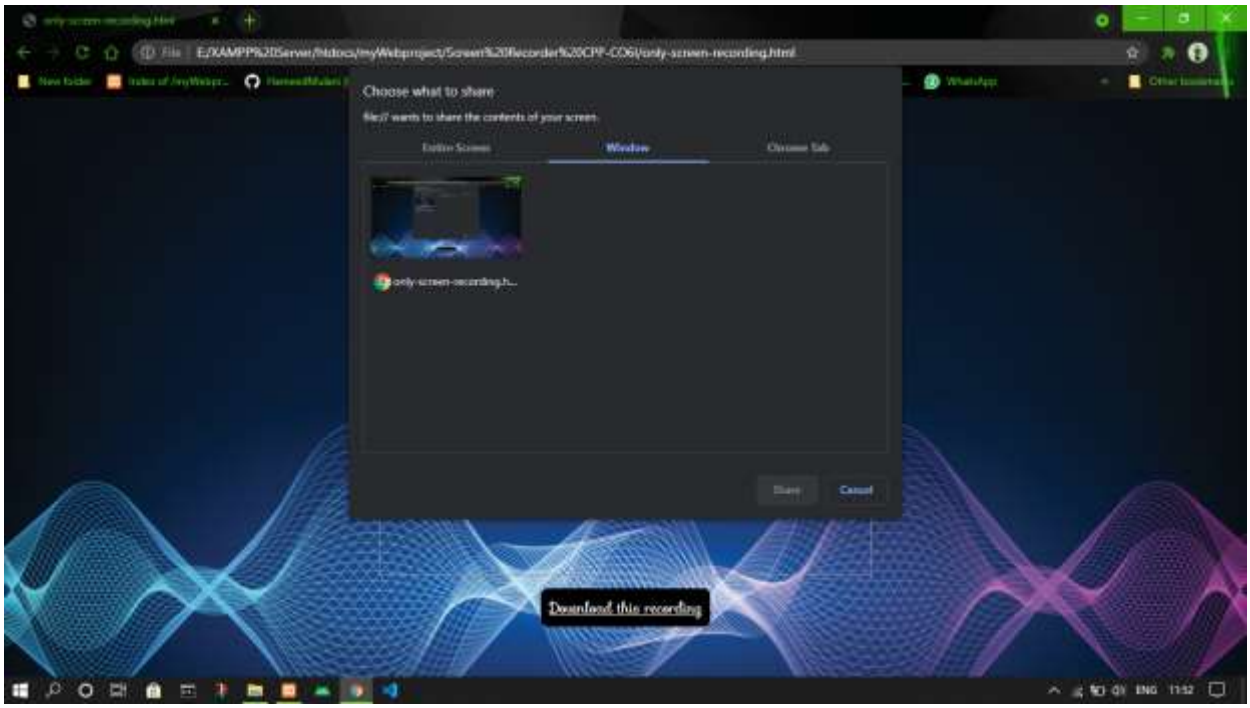
1. ***Screen recording.***
2. ***Screen recording with mike***
3. ***Screen recording with internal audio.***
4. ***Screen recording with mike and camera.***
5. ***Screen recording with internal audio and camera.***

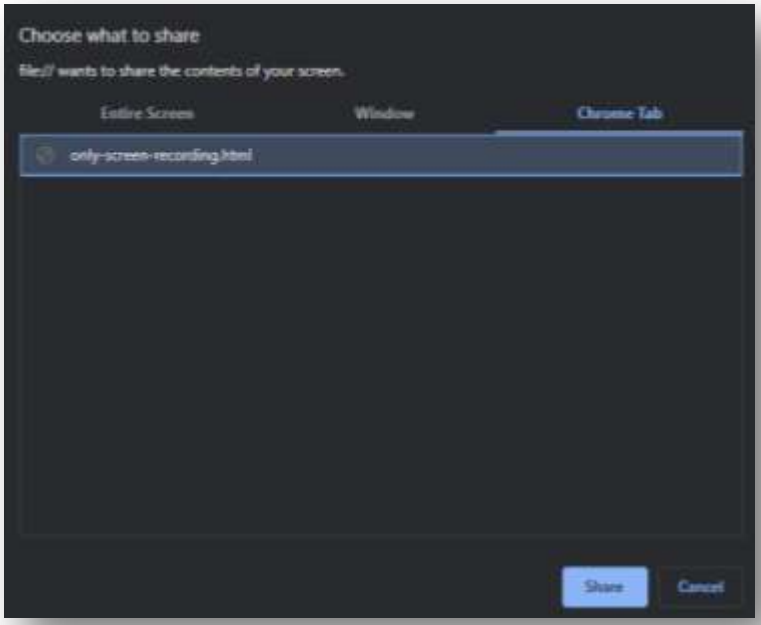
1) Screen recording Output:

Click on **Start Recording Button** to start recording

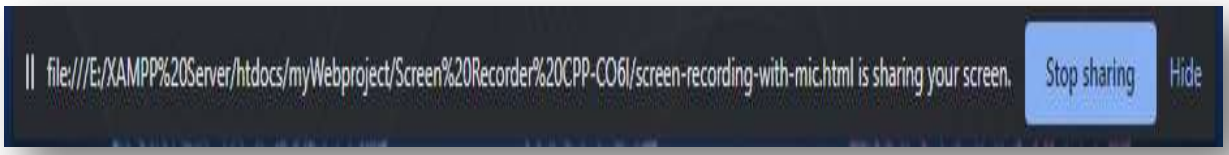
Then pop ups **Sharing Window,** we want to select any one of them.





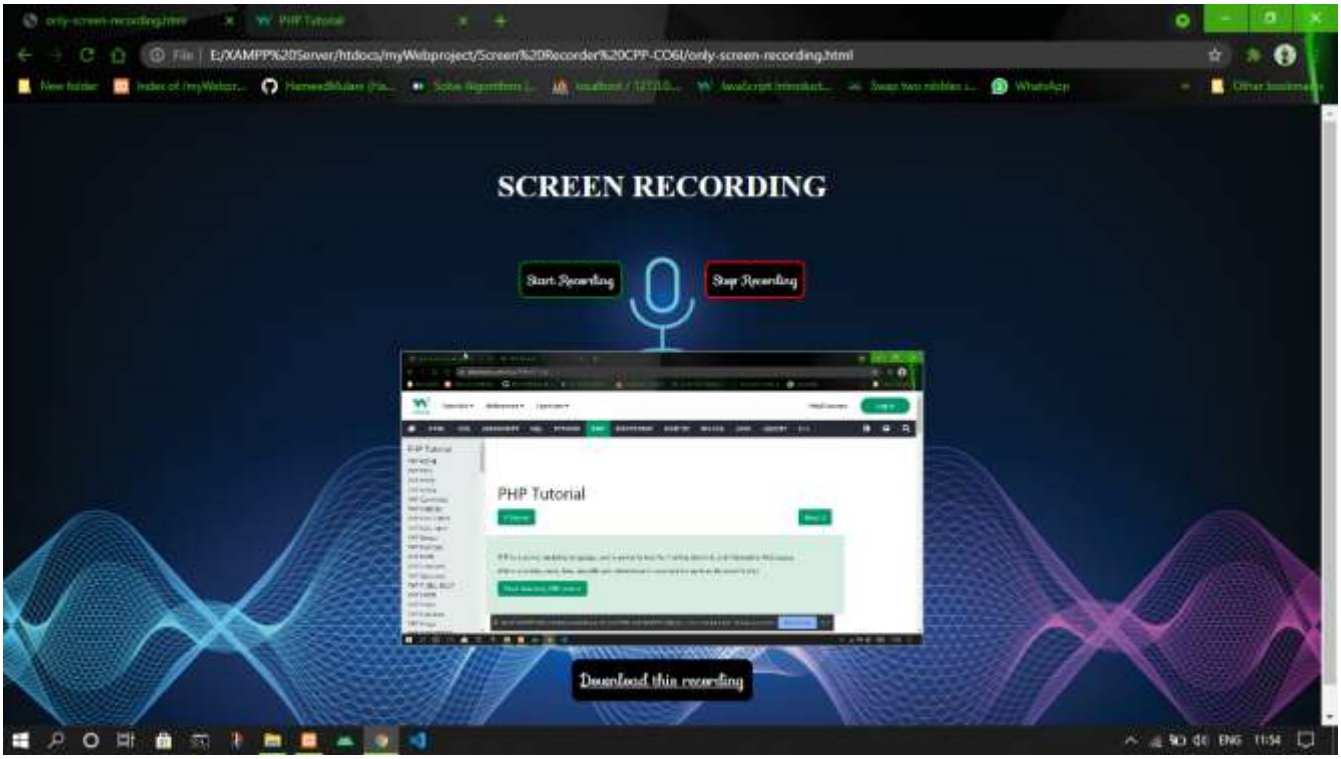


Click on ***Share*** Button, Now Recording will Started.



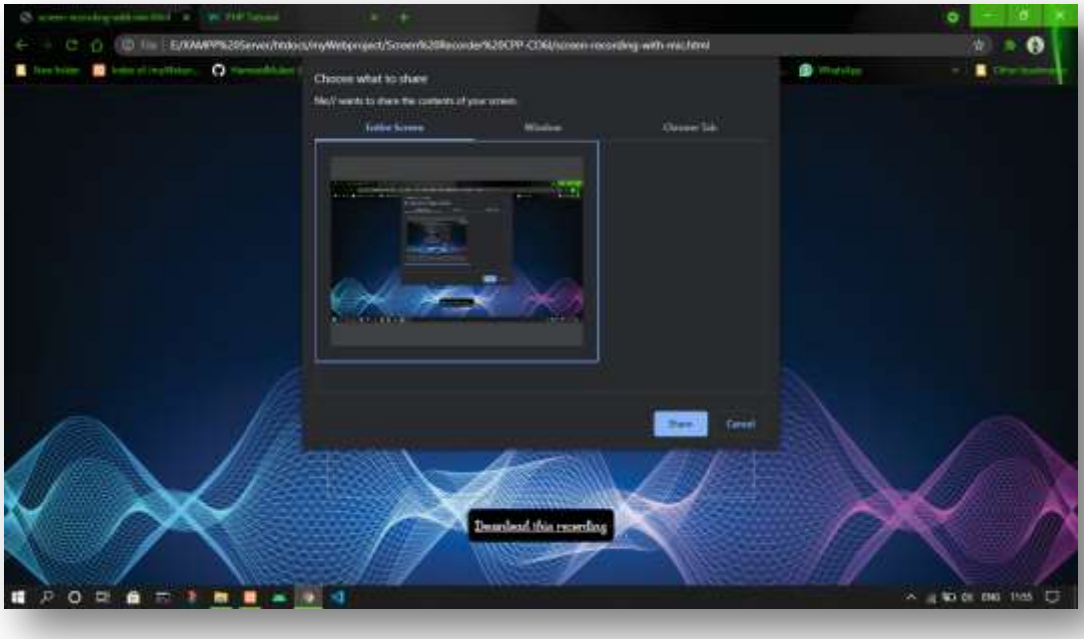
For stopping the current recording window, *click on* ***Stop Recording*** Button.

After Clicking on ***Stop Recording*** Button, the recorded Video will occurs on Video Tag.



For downloading the recorded video, ***Click on Download*** Button for download Recorded file.

2. Screen Recording With Mike.

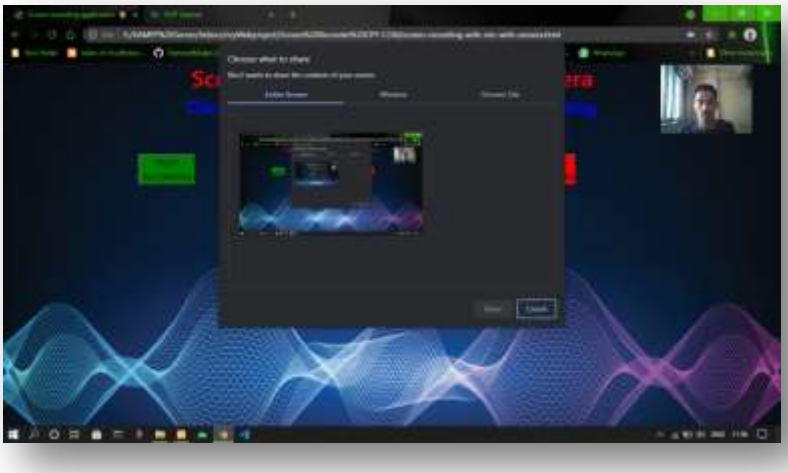



SCREEN RECORDING
Download this recording

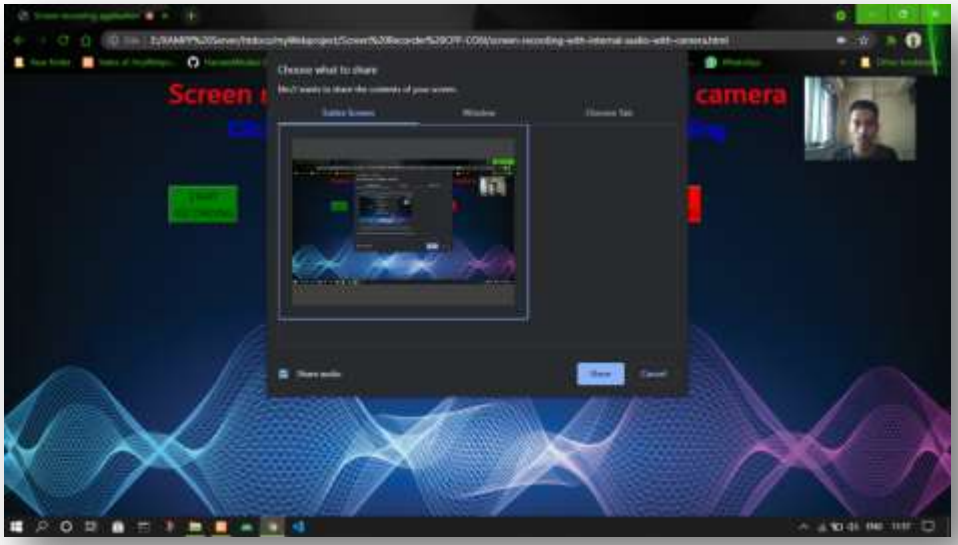

SCREEN RECORDING
Download this recording

3. Screen recording with internal Audio.

4. Screen recording with Mike and Web Camera.



5. Screen recording with internal audio and Web Camera.




Screen recording with internal audio with camera

Chapter 7:

Conclusions and future scope:-

Screen recording is a powerful way to share any information that you have on your screen with your viewers. Before you make your first screencast, these nine essential tips will help you create a quality screen recording that you'll want to share.

1. You can record anything on your screen.
2. We can plan what we are going to say..
3. Length can affect effectiveness.
4. A matter of audio quality.
5. Can register the right size.
6. If you use a webcam, you can make sure it looks good.
7. Your mouse can see the cursor.
8. If we can do this, we can cut out mistakes and unnecessary content.
9. You can find out where you are going to share your video.

Depending on your personal needs and the operating system you are using, you may be able to achieve your goals through a free screen recorder. And if you plan to use screen recording in different ways, you can use different options.
Hopefully one of these screen recording programs will meet your needs, whether you're creating YouTube tutorials or just sharing gaming success with your peers.

Chapter 8:

References and bibliography

- https://javascript.info/(Official discord.js library)
- https://www.w3schools.com/js/
- https://webrtc.org/ (Screen Capture API)
- https://webrtc.org/getting-started/media-devices
- https://webrtc.org/getting-started/overview/
- https://webrtc.org/support/
- https://www.tutorialspoint.com/webrtc/
- https://www.tutorialspoint.com/webrtc/webrtc_overview.htms
- https://webrtc.github.io/samples/
- https://www.youtube.com/channel/UCbwXnUipZsLfUckBPsC7Jog
- https://stackoverflow.com/
- https://vincentgarreau.com/particles.js/ (particles.js)
- https://use.fontawesome.com/releases/v5.4.1/css/all.css/
- https://github.com/webrtc
- https://www.youtube.com/watch?v=K6L38xk2rkk&t=84s/
- https://www.php.net/manual/en/manual.php/(Php Manual)
- https://www.w3schools.com/Php/
- https://www.php.net/(php)
- https://getbootstrap.com/ (Bootstrap official)