


Topic	Video Chat App - DevOps	
Class Description	Student will learn about DevOps, i.e. Development and Operations, which means, they will be deploying an application on a remote Heroku Server!	
Class	C-218	
Class time	45 mins	
Goal	<ul style="list-style-type: none"> • Learning about PeerJS • Implementation of PeerJS and PeerJS server 	
Resources Required	<ul style="list-style-type: none"> • Teacher Resources: <ul style="list-style-type: none"> ○ Laptop with internet connectivity ○ Earphones with mic ○ Notebook and pen ○ Visual Studio Code • Student Resources: <ul style="list-style-type: none"> ○ Laptop with internet connectivity ○ Earphones with mic ○ Notebook and pen ○ Visual Studio Code 	
Class structure	Warm-Up Teacher - led Activity 1 Student - led Activity 1 Wrap-Up	10 mins 15 mins 15 mins 5 mins
WARM UP SESSION - 10mins		
Teacher Action		Student Action

Hey <student's name>. How are you? It's great to see you! Are you excited to learn something new today?	ESR: Hi, thanks, yes, I am excited about it!
Q&A Session	
Question	Answer
Why should web apps be deployed? A. To make it available for all B. To make it for two or three people C. To use it local D. None of the above	A
Why do we use github? A. To make code repositories and to maintain code versions B. To make pdf repositories C. To make folder repositories D. None of the above	A
TEACHER-LED ACTIVITY - 15mins	
Teacher Initiates Screen Share	
<p style="text-align: center;"><u>ACTIVITY</u></p> <ul style="list-style-type: none"> • Understanding about DevOps Engineering • Creating a master github repository 	
Teacher Action	Student Action
In the last class, we successfully implemented PeerJS, where we linked the functionality of our Rooms with Sockets, and also added users to a chatroom when the peers open the same page on the browser.	

<p>Do you have any doubts in the last class?</p> <p><i>Teacher clears the doubts, if any</i></p> <p>Great! We encountered a problem in the last class, which was that because PeerJS makes a lot of requests back and forth from client to server, it is impossible to test it on NGROK!</p> <p>Also, we were running the app locally, but the PeerJS server that we created wanted to use runs on port 443, which we don't have access to locally, since port 443 is HTTPS!</p> <p>With all these issues, how do you think we can solve our problem?</p> <p>One universal solution is to deploy our application on a remote server, so anyone can access it!</p> <p>Do you notice how you enter a website's name and the website opens up? Similarly, when we try to run our application through NGROK, it provides us with a URL that we can use to access the application from any device, right?</p> <p>Can you tell me what actually happens when we run the application from NGROK and how it opens up from any device using the URL?</p>	<p>ESR: Varied</p> <p>ESR: Varied!</p> <p>ESR: Yes</p> <p>ESR: The application is running on our device and people are using a tunnel through the URL to access the application on our device!</p>
---	--

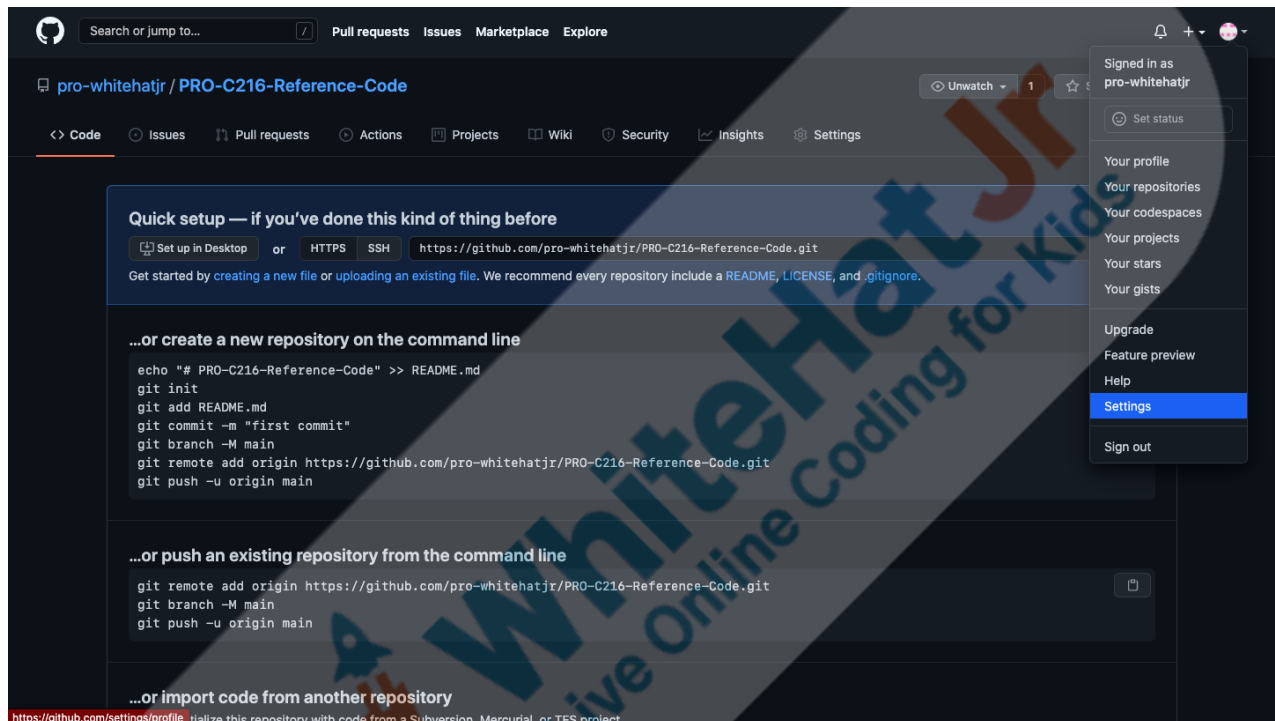
<p>That's awesome! Now, we often close our devices, or are away from our devices. We can't keep the application running on our device forever right? Also, NGROK only gives us a limit of 2 hours in which someone can access the website.</p> <p>What do you think a more permanent solution for this would be?</p> <p>A better solution for this would be to run our application on a remote device (or we can call it a remote server) which we don't have to monitor. This is a specialised field in Software Engineering, known as DevOps, which means Development and Operations.</p> <p>Today, we are going to host our video chat application on a remote device, or a remote server. We will learn about how DevOps actually work, and you will be deploying your very first application!</p> <p>Are you excited?</p> <p>Let's get started then!</p>	<p>ESR: Varied!</p> <p>ESR: Yes!</p>
<p>Let's understand a little bit about how DevOps work in big companies and for big softwares!</p> <p>There are situations where, at times, multiple developers are working on the same project/application at the same time!</p> <p>Now we know that to collaborate and maintain a single codebase, github can be used.</p> <p>Github can also be used to clone a remote repository into our devices.</p> <p>Similarly, let's say we have a machine allocated to us that is</p>	<p>ESR: Varied</p>

<p>at a different place! Can we clone our code into that machine too, if we get access to it?</p>	<p>ESR: Yes!</p>
<p>And what after we clone it? Can we run the application on that machine once we have it there?</p>	<p>ESR: Yes!</p>
<p>That's what DevOps is all about!</p>	
<p>How it works is that big softwares contains a single repository, to which multiple developers contribute! Now, as soon as the code updates on these repositories, a devops engineer pulls the latest update into the remote device on which they are running the application, and then they run the application again.</p>	
<p>This way, the application keeps getting updated.</p>	
<p>Now, it looks like tedious work to pull all the updates and latest code into a remote machine and run the application again and again! What devops engineers do in these cases is that they automate the process!</p>	
<p>This means that as soon as the code gets updated into the Github Repository, it also gets updated on the remote server, and the application restarts itself!</p>	
<p>Cool, isn't it?</p>	<p>ESR: Yes!</p>
<p>Okay! Now we understand the process, and a part of it we will understand as we implement it. The first thing that we should do is that we should create a master github repository, on which we will make our changes from now onwards!</p>	
<p><i>Teacher guides the student in creating a new Github repository. This is a student only activity in which the teacher is guiding. The teacher may or may not do this, as</i></p>	

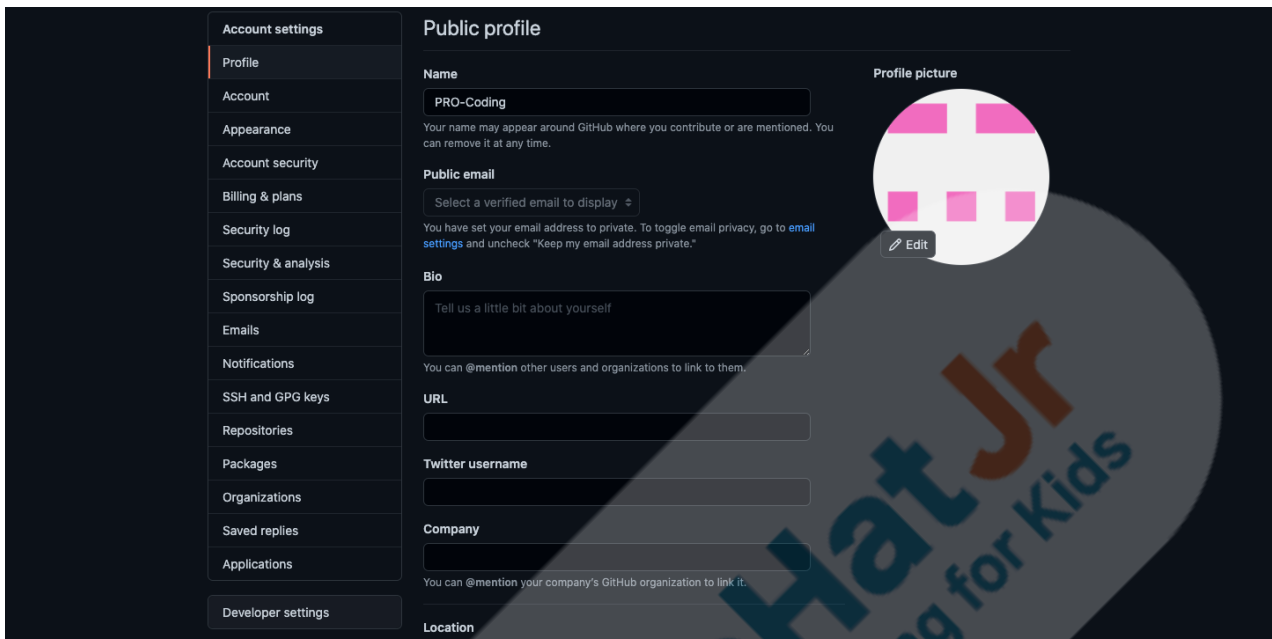
<p><i>per their wish.</i></p> <p><i>Note - The previous class code is available in Teacher Activity 1</i></p>	<p><i>Student opens Student Activity 1</i></p>
<p>Here, we have the code from the last class! Let's clone it and then unzip the files from it!</p> <p><i>Student clones the code from Student Activity 1 and unzips the code.</i></p>	
<p>Now let's create a new github repository!</p> <p><i>Teacher guides the student to create a new Github Repository for the video chat app</i></p>	<p><i>Student creates a new Github Repository for Video Chat App</i></p>
 <p>The screenshot shows the GitHub 'Quick setup' interface. It offers three methods to get started: 'Set up in Desktop', 'HTTPS', and 'SSH'. The 'HTTPS' method is selected, showing the URL 'https://github.com/apoorvelous/video-chat-app.git'. Below this, it provides instructions for creating a new repository on the command line, pushing an existing repository, and importing code from another repository. The command line instructions for creating a new repository are: <code>echo "# video-chat-app" >> README.md</code>, <code>git init</code>, <code>git add README.md</code>, <code>git commit -m "first commit"</code>, <code>git branch -M main</code>, <code>git remote add origin https://github.com/apoorvelous/video-chat-app.git</code>, and <code>git push -u origin main</code>. The instructions for pushing an existing repository are: <code>git remote add origin https://github.com/apoorvelous/video-chat-app.git</code>, <code>git branch -M main</code>, and <code>git push -u origin main</code>. The 'Import code' button is visible at the bottom.</p>	
<p>Alright, now we need to push our code into this repository!</p>	

*Github has changed the rules of pushing anything on github recently on **13th August 2021**. We need a personal token now to do so, and the steps mentioned below.*

To create a personal token, we will first need to go to the settings -

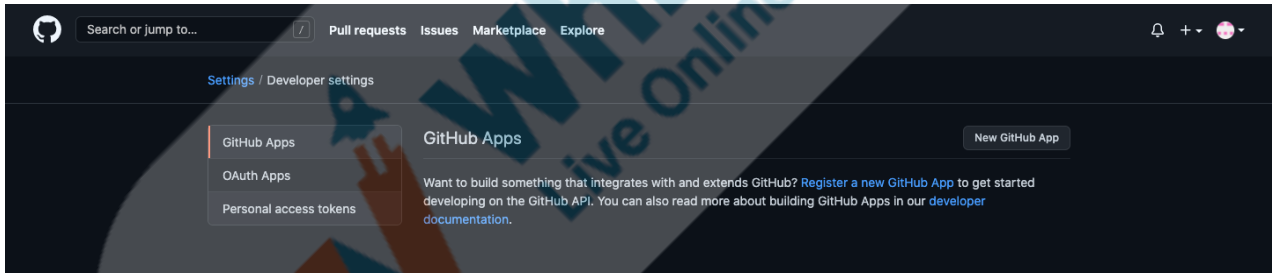


In Settings, go to Developer Settings



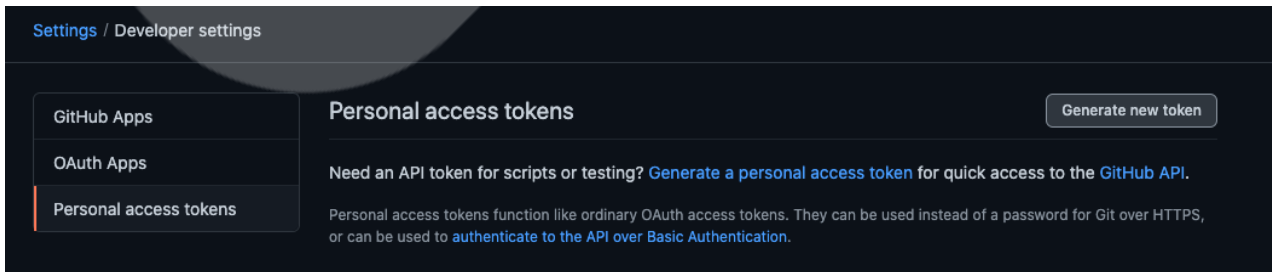
The screenshot shows the GitHub 'Public profile' settings page. On the left is a sidebar menu with options: Account settings, Profile, Account, Appearance, Account security, Billing & plans, Security log, Security & analysis, Sponsorship log, Emails, Notifications, SSH and GPG keys, Repositories, Packages, Organizations, Saved replies, Applications, and Developer settings. The main content area is titled 'Public profile' and includes fields for Name (set to 'PRO-Coding'), Public email (with a dropdown to 'Select a verified email to display'), Bio (a text area), URL, Twitter username, Company, and Location. A profile picture placeholder is shown with an 'Edit' button. A large watermark 'WhiteHat Jr Live Online Coding for Kids' is visible across the page.

Here, select the 3rd option, which is for **Personal Access Token** -



The screenshot shows the 'Settings / Developer settings' page in GitHub. On the left, a sidebar menu has 'GitHub Apps', 'OAuth Apps', and 'Personal access tokens'. The main content area is titled 'GitHub Apps' and contains a 'New GitHub App' button and introductory text about building GitHub Apps. A large watermark 'WhiteHat Jr Live Online Coding for Kids' is visible across the page.

Next, click on Generate Access Token -

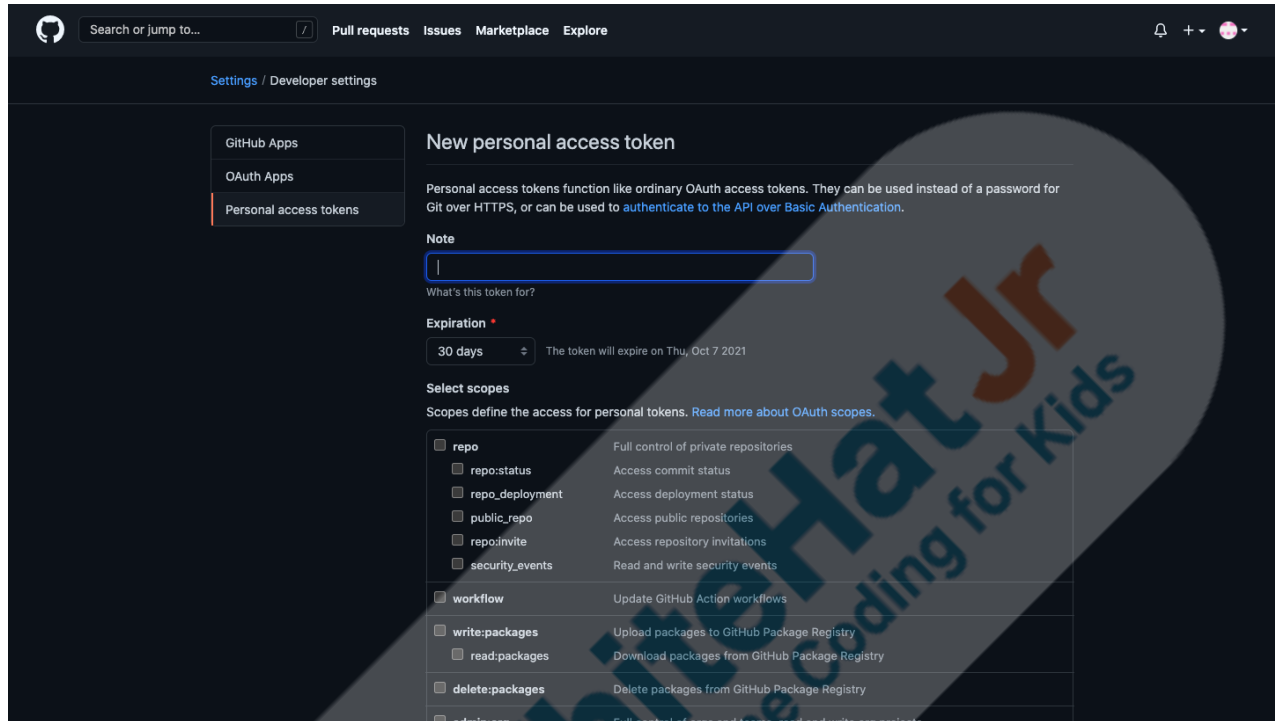


The screenshot shows the 'Settings / Developer settings' page with the 'Personal access tokens' section selected in the sidebar. The main content area is titled 'Personal access tokens' and features a 'Generate new token' button. Below the button, there is explanatory text about API tokens. A large watermark 'WhiteHat Jr Live Online Coding for Kids' is visible across the page.

© 2021 - WhiteHat Education Technology Private Limited.

Note: This document is the original copyright of WhiteHat Education Technology Private Limited.
Please don't share, download or copy this file without permission.

You will see the following screen -



Type anything in the **Note** section as per you like. It could be to update the video chat app!

Also, select the **repo** and **admin:repo_hook** from the checkbox options!

Settings / Developer settings

GitHub Apps

OAuth Apps

Personal access tokens

New personal access token

Personal access tokens function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to [authenticate to the API over Basic Authentication](#).

Note

Video Chat App Update

What's this token for?

Expiration *

30 days

The token will expire on Thu, Oct 7 2021

Select scopes

Scopes define the access for personal tokens. [Read more about OAuth scopes.](#)

☒ repo

Full control of private repositories

☒ repo:status

Access commit status

☒ repo_deployment

Access deployment status

☒ public_repo

Access public repositories

☒ repo:invite

Access repository invitations

☒ security_events

Read and write security events

And

☒ admin:repo_hook

Full control of repository hooks

☒ write:repo_hook

Write repository hooks

☒ read:repo_hook

Read repository hooks

Finally, click on the **generate token** button and Voila, the token is generated!

☐ admin:gpg_key

Full control of public user GPG keys (Developer Preview)

☐ write:gpg_key

Write public user GPG keys

☐ read:gpg_key

Read public user GPG keys

Generate token

Cancel

Copy the token that you see here! It will only be displayed once so be careful with it or the entire list of steps above would need to be repeated again.

Better yet, open a new tab and re-open the github repository that you just created!

Next, open a **cmd/terminal** window and navigate to the project that you just unzipped.

```
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/pro-whitehatjr/PRO-C216-Reference-Code.git
git push -u origin main
```

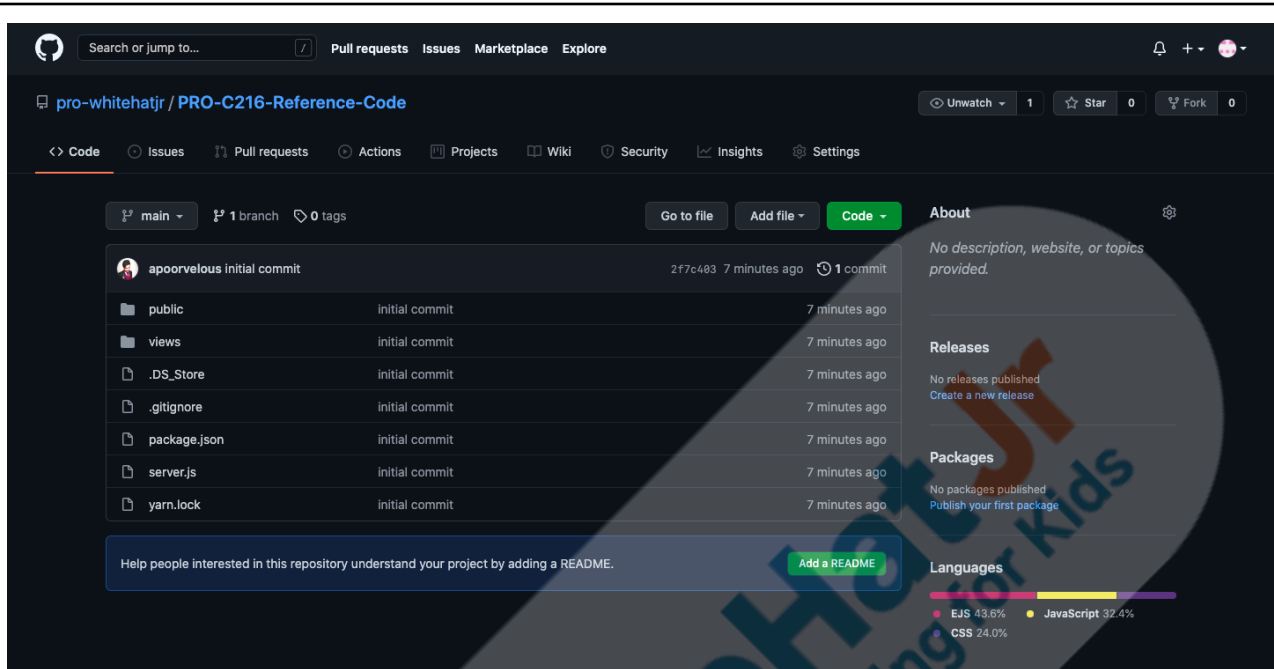
On running the last command, which is **git push -u origin main**, you **may** encounter an error!

Run the following commands -

```
git config --global user.name "Your Username"
git config --global user.password "Personal Access Token"
```

Here, do note that in the first command, **"Your Username"** would be the student's (or teacher's if the teacher is doing it) username and in the second command, **"Personal Access Token"** would be the personal access token that you just generated!

Once this is done, run the **git push -u origin main** command again, and you will see the code pushed into your repository!



Awesome! Now, our repository is all ready! We can now use it to deploy our code into our remote device (or our remote server!)

Teacher Stops Screen Share

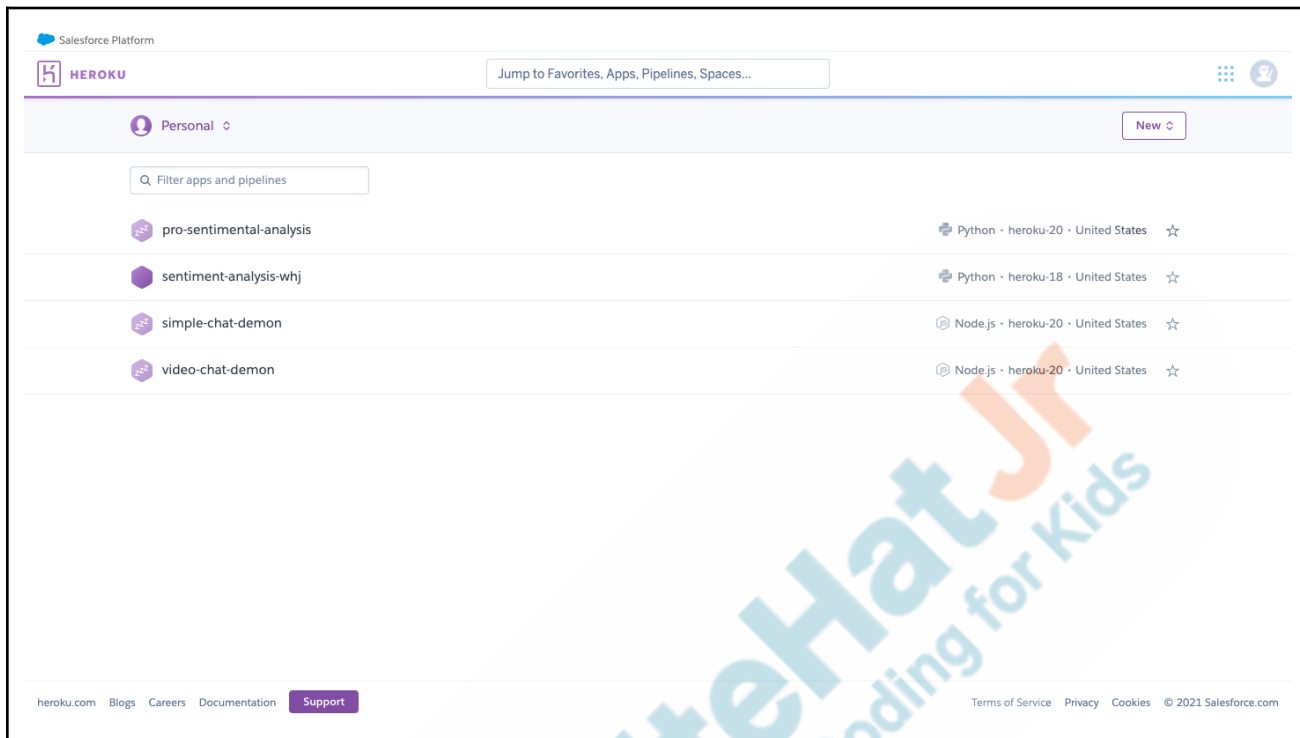
STUDENT-LED ACTIVITY - 15 mins

- Ask the student to press the ESC key to come back to the panel.
- Guide the student to start Screen Share.
- The teacher gets into Full Screen.

ACTIVITY

- Introduction to Heroku!
- Deploying your first Web Application!

Teacher Action	Student Action
<p>You must be wondering, where will we find this remote device?</p> <p>Well, there is an excellent platform just for this, known as Heroku. It is a platform that offers remote devices to host/deploy applications, and these remote devices are known as dyno.</p> <p>Let's start by signing up on Heroku!</p> <p><i>Teacher opens Teacher Activity 2 and guides the student to sign up</i></p>	<p><i>Student opens Student Activity 2 and signs up</i></p>
 <p>The screenshot shows the Heroku sign-up page. On the left, there are three sections: 'Free account' (Create apps, connect databases and add-on services, and collaborate on your apps, for free.), 'Your app platform' (A platform for apps, with app management & instant scaling, for development and production.), and 'Deploy now' (Go from code to running app in minutes. Deploy, scale, and deliver your app to the world.). On the right, there is a form with fields for: First name, Last name, Email address, Company name, Role (dropdown), Country (dropdown), and Primary development language (dropdown). A 'Sign up' button is at the bottom right.</p>	
<p>Now, let's login into Heroku and understand the dashboard.</p> <p>On successful login, you will see the following screen -</p>	



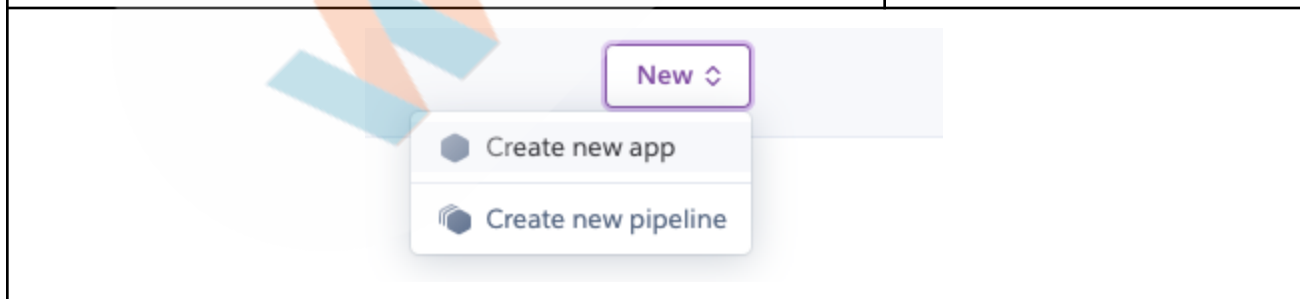
Here, for you, there will not be any projects as demonstrated in the screenshot, since your account is new.

Now, we can see a button on the top right, which says **New**.

Click on it, and then click on **Create a new app**

Teacher guides the student

Student follows the instructions



Next, enter a name for your app and click on the **Create App** button!

Teacher guides the student

Student follows the instructions

Create New App

App name

✓

video-chat-app-216 is available

Choose a region

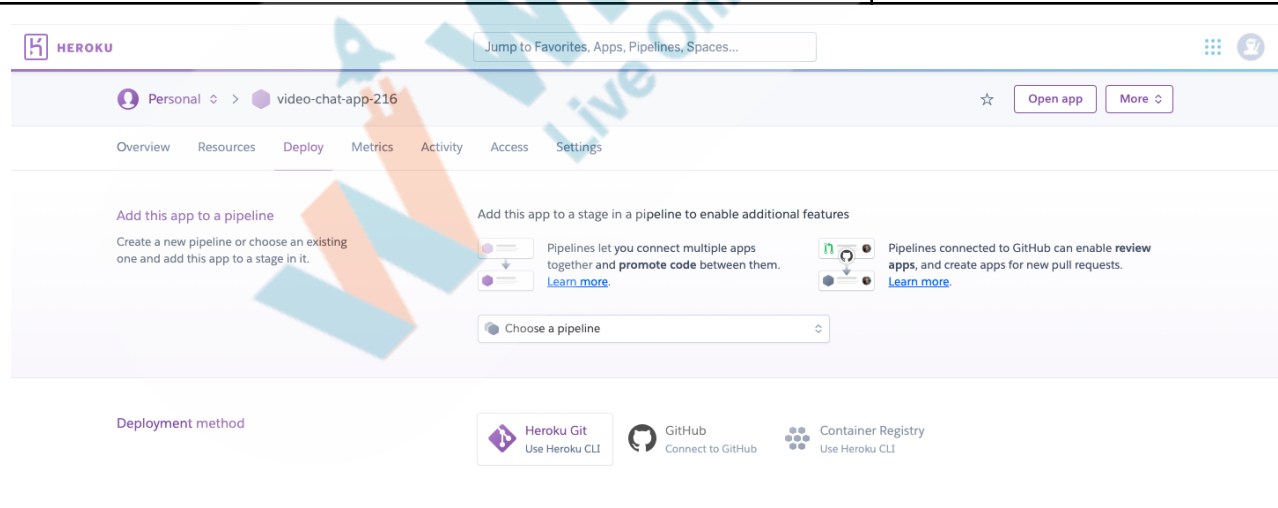
United States

⌵

Add to pipeline...




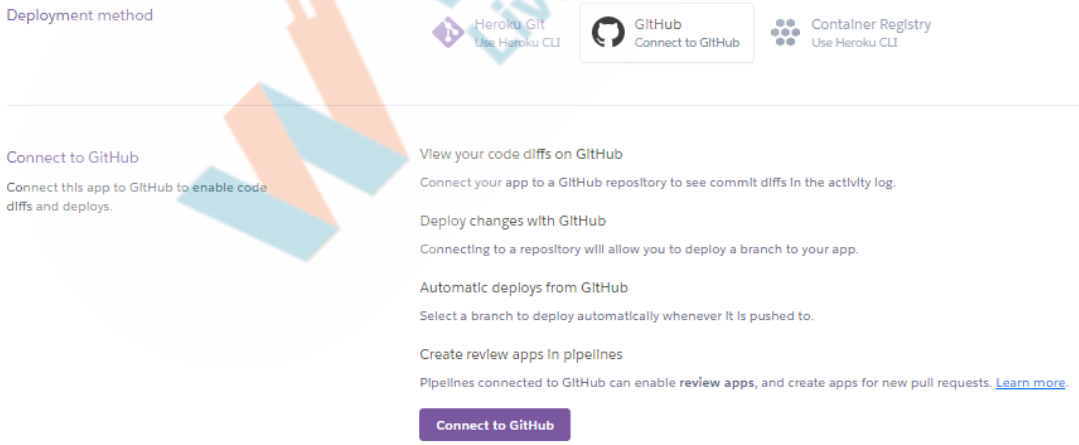
Create app

Once done, we will be prompted to the following screen -



The screenshot shows the Heroku dashboard for an app named 'video-chat-app-216'. The top navigation bar includes 'HEROKU', a search bar, and user profile icons. Below the navigation bar, there's a breadcrumb trail: 'Personal > video-chat-app-216'. The main content area has tabs for 'Overview', 'Resources', 'Deploy', 'Metrics', 'Activity', 'Access', and 'Settings'. The 'Deploy' tab is active, showing options to 'Add this app to a pipeline' and 'Add this app to a stage in a pipeline to enable additional features'. There are also links for 'Heroku Git', 'GitHub', and 'Container Registry' under the 'Deployment method' section.

Here, you will notice a section that says **Deployment methods** -

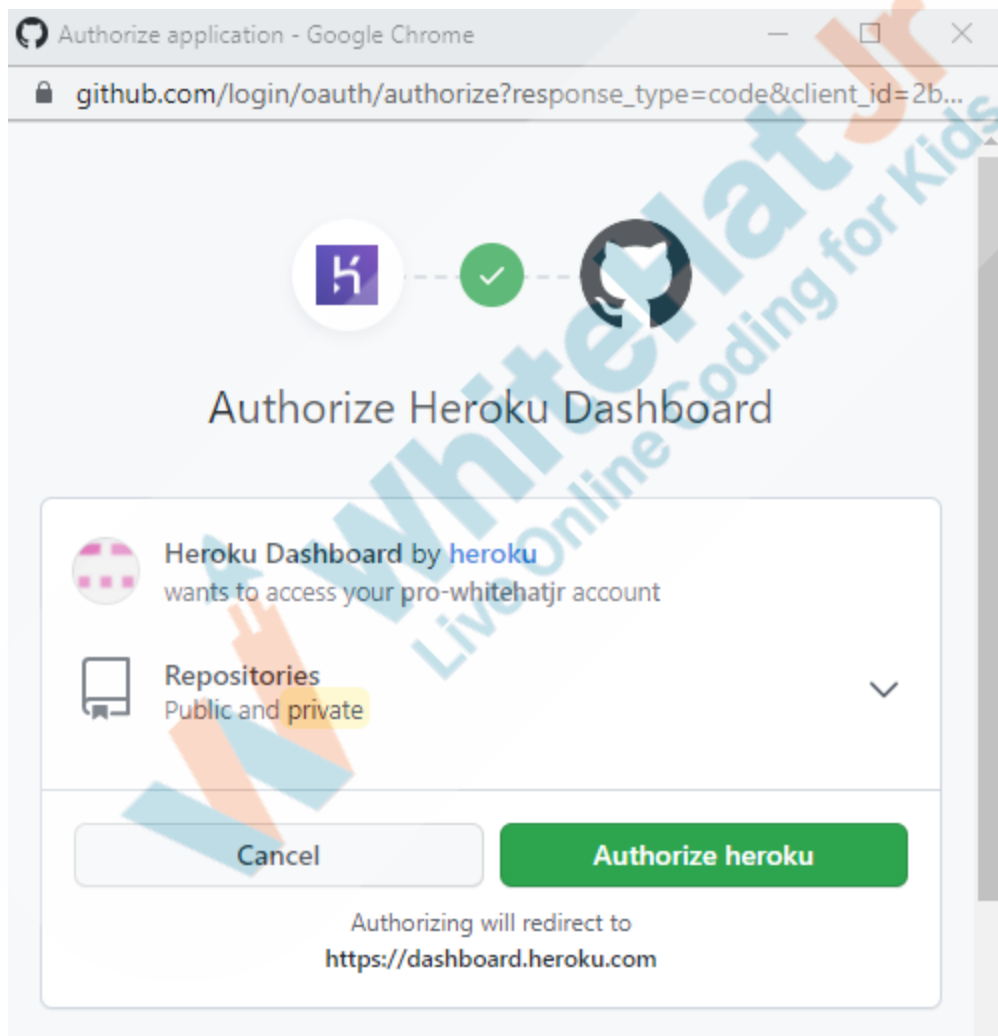
<p>Deployment method</p> <div>  Heroku Git Use Heroku CLI  GitHub Connect to GitHub  Container Registry Use Heroku CLI </div>	
<p>Here, the two of the most famous methods used to deploy on Heroku are -</p> <ol style="list-style-type: none"> 1. Heroku Git - Default Method 2. Github - Widely used Method <p>Heroku Git method is a rather complicated one, where you have to download and install a tool called heroku which works on terminal/cmd.</p> <p>In this method, Heroku provides you with a git repository in which you have to push your code for it to work.</p> <p>For us, we already have created a Github repository, so why not use it directly? Right.</p> <p>Let's select the Github method from this prompt and see the options we get -</p>	
 <p>The screenshot shows the Heroku deployment interface. At the top, there are three options: 'Heroku Git' (selected), 'GitHub', and 'Container Registry'. Below this, the 'Connect to GitHub' section is expanded, showing instructions on how to connect the app to a GitHub repository, including options for viewing code diffs, deploying changes, and creating review apps. A 'Connect to GitHub' button is visible at the bottom of this section.</p>	
<p>Here, we see an option below which says -</p>	

Connect to Github

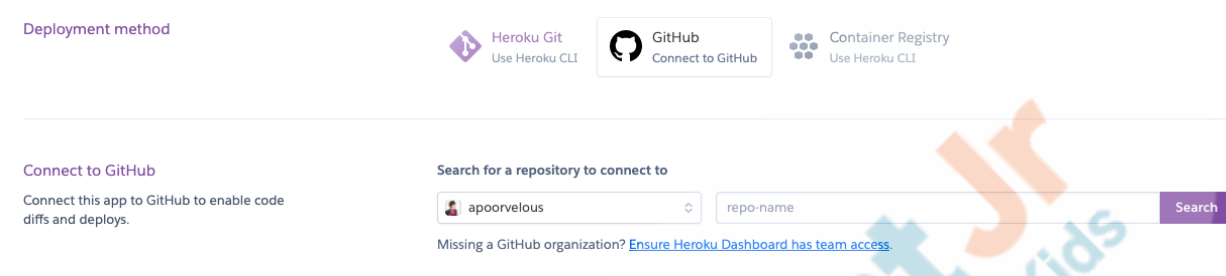
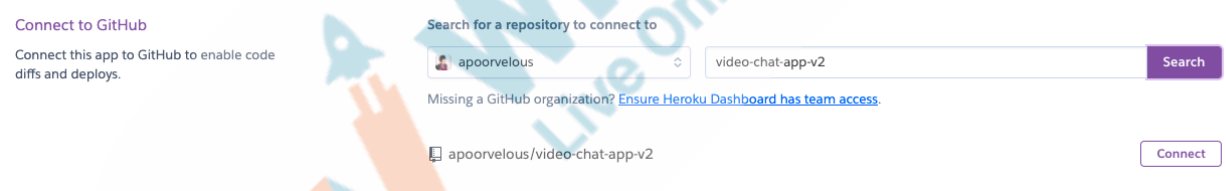
Let's click on it and connect our Heroku with Github

Teacher guides the student

Student follows the instructions



A dialog box opens up, where you can select the Github account which you want to authorise with Heroku, and click

<p>on Authorize heroku</p> <p><i>Teacher guides the student</i></p>	<p><i>Student follows the instructions</i></p>
	
<p>Once done, we can see that Heroku asks us for a Repo name that we want to connect with our app.</p> <p>Let's enter the name of the repository that we just created and click on search -</p> <p><i>Teacher guides the student</i></p>	<p><i>Student follows the instructions</i></p>
	
<p>Once you search, you will see that Heroku has located the repository, and we can click on connect to connect that particular repo to the Heroku app we created.</p> <p>Let's do that -</p> <p><i>Teacher guides the student</i></p>	<p><i>Student follows the instructions</i></p>

<p>Automatic deploys</p> <p>Enables a chosen branch to be automatically deployed to this app.</p>	<div style="border: 1px solid #add8e6; padding: 5px; margin-bottom: 10px;"> <p> You can now change your main deploy branch from "master" to "main" for both manual and automatic deploys, please follow the instructions here.</p> </div> <p>Enable automatic deploys from GitHub</p> <p>Every push to the branch you specify here will deploy a new version of this app. Deploys happen automatically: be sure that this branch is always in a deployable state and any tests have passed before you push. Learn more.</p> <p>Choose a branch to deploy</p> <div style="border: 1px solid #ccc; padding: 2px; display: inline-block;">  main </div> <p><input type="checkbox"/> Wait for CI to pass before deploy</p> <p><small>Only enable this option if you have a Continuous Integration service configured on your repo.</small></p> <p>Enable Automatic Deploys</p>
<p>Manual deploy</p> <p>Deploy the current state of a branch to this app.</p>	<p>Deploy a GitHub branch</p> <p>This will deploy the current state of the branch you specify below. Learn more.</p> <p>Choose a branch to deploy</p> <div style="border: 1px solid #ccc; padding: 2px; display: inline-block;">  main </div> <p>Deploy Branch</p>

Now that the app is connected, we can see that the options are different now.

On scrolling down to the bottom, we can see a **Deploy Branch** button in the last section which says **Manual deploy**.

We can select the branch that we want to deploy. For us, it will be the default **main** branch.

Let's click on it so our **app** gets deployed -

Teacher guides the student

Student follows the instructions

Manual deploy

Deploy the current state of a branch to this app.

Deploy a GitHub branch

This will deploy the current state of the branch you specify below. [Learn more](#).

Choose a branch to deploy

Deploy Branch

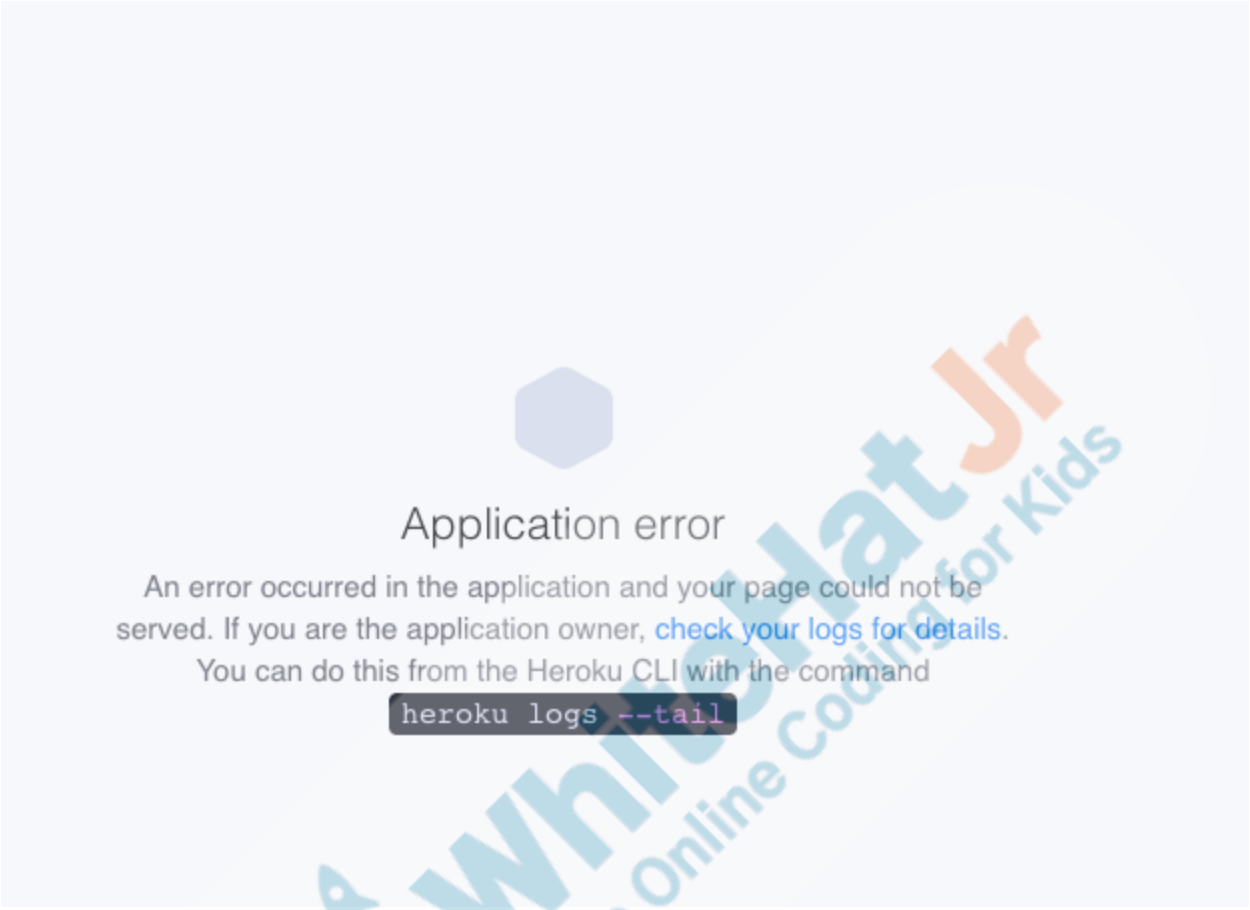
Receive code from GitHub	✓
Build main 21f5f000	✓
Release phase	✓
Deploy to Heroku	✓

Your app was successfully deployed.

[View](#)

Once deployed, we can click on the **View** button to see if it is deployed or not.

Let's do that -

 <p>Application error</p> <p>An error occurred in the application and your page could not be served. If you are the application owner, check your logs for details.</p> <p>You can do this from the Heroku CLI with the command</p> <pre>heroku logs --tail</pre>	
Oh no! It shows an Application Error! We can check the logs by clicking on the <i>check you logs for details</i> link.	

```

2021-09-07T12:08:36.514312+00:00 app[web.1]: > video-chat-app@1.0.0 start /app
2021-09-07T12:08:36.514312+00:00 app[web.1]: > node server.js
2021-09-07T12:08:36.514312+00:00 app[web.1]:
2021-09-07T12:09:33.558035+00:00 heroku[web.1]: Error R10 (Boot timeout) -> Web process failed to bind
to $PORT within 60 seconds of launch
2021-09-07T12:09:33.606748+00:00 heroku[web.1]: Stopping process with SIGKILL
2021-09-07T12:09:33.688757+00:00 heroku[web.1]: Process exited with status 137
2021-09-07T12:09:34.042752+00:00 heroku[web.1]: State changed from starting to crashed
2021-09-07T12:09:34.071754+00:00 heroku[web.1]: State changed from crashed to starting
2021-09-07T12:09:36.097478+00:00 heroku[web.1]: Starting process with command `npm start`
2021-09-07T12:09:38.106544+00:00 app[web.1]:
2021-09-07T12:09:38.106556+00:00 app[web.1]: > video-chat-app@1.0.0 start /app
2021-09-07T12:09:38.106557+00:00 app[web.1]: > node server.js
2021-09-07T12:09:38.106557+00:00 app[web.1]:
2021-09-07T12:10:36.325565+00:00 heroku[web.1]: Error R10 (Boot timeout) -> Web process failed to bind
to $PORT within 60 seconds of launch
2021-09-07T12:10:36.490875+00:00 heroku[web.1]: Stopping process with SIGKILL
2021-09-07T12:10:36.600111+00:00 heroku[web.1]: Process exited with status 137
2021-09-07T12:10:36.654706+00:00 heroku[web.1]: State changed from starting to crashed
2021-09-07T12:13:25.813781+00:00 heroku[router]: at=error code=H10 desc="App crashed" method=GET path=
"/" host=video-chat-app-216.herokuapp.com request_id=8f3e6bd2-b2c9-4f13-b852-770b2fa3f710 fwd="205.254.
164.98" dyno= connect= service= status=503 bytes= protocol=https
2021-09-07T12:13:26.400591+00:00 heroku[router]: at=error code=H10 desc="App crashed" method=GET path=
"/favicon.ico" host=video-chat-app-216.herokuapp.com request_id=1c6b54b9-ddc9-4c2b-82a1-1e968449d631 f
wd="205.254.164.98" dyno= connect= service= status=503 bytes= protocol=https

```

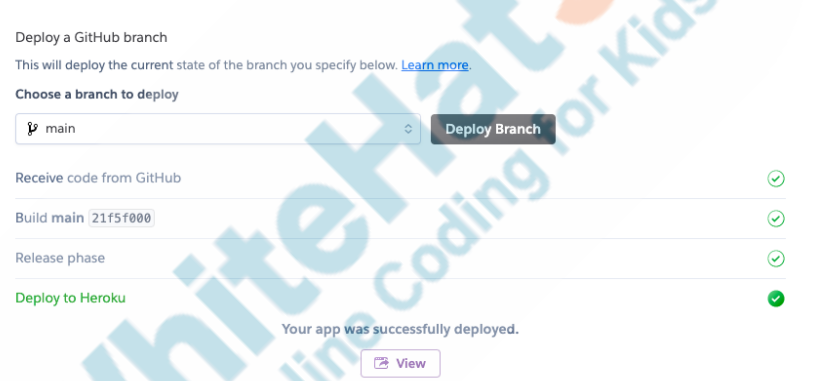
Here, one of the logs tells us that it was not able to bind with the port within 60 seconds! The reason behind this is that we run the app locally on port 3030, but Heroku uses a different port! To tackle this, let's make a small change into our **server.js** file -

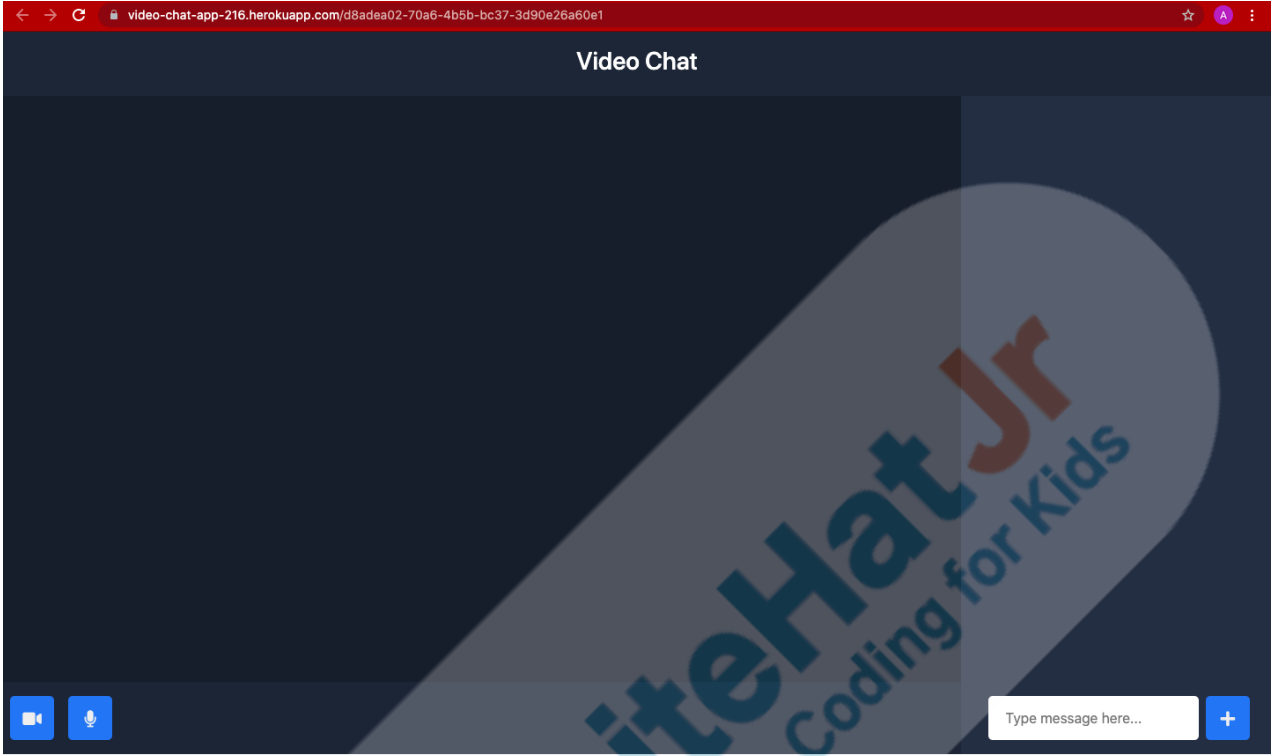
Teacher guides the student to make the change

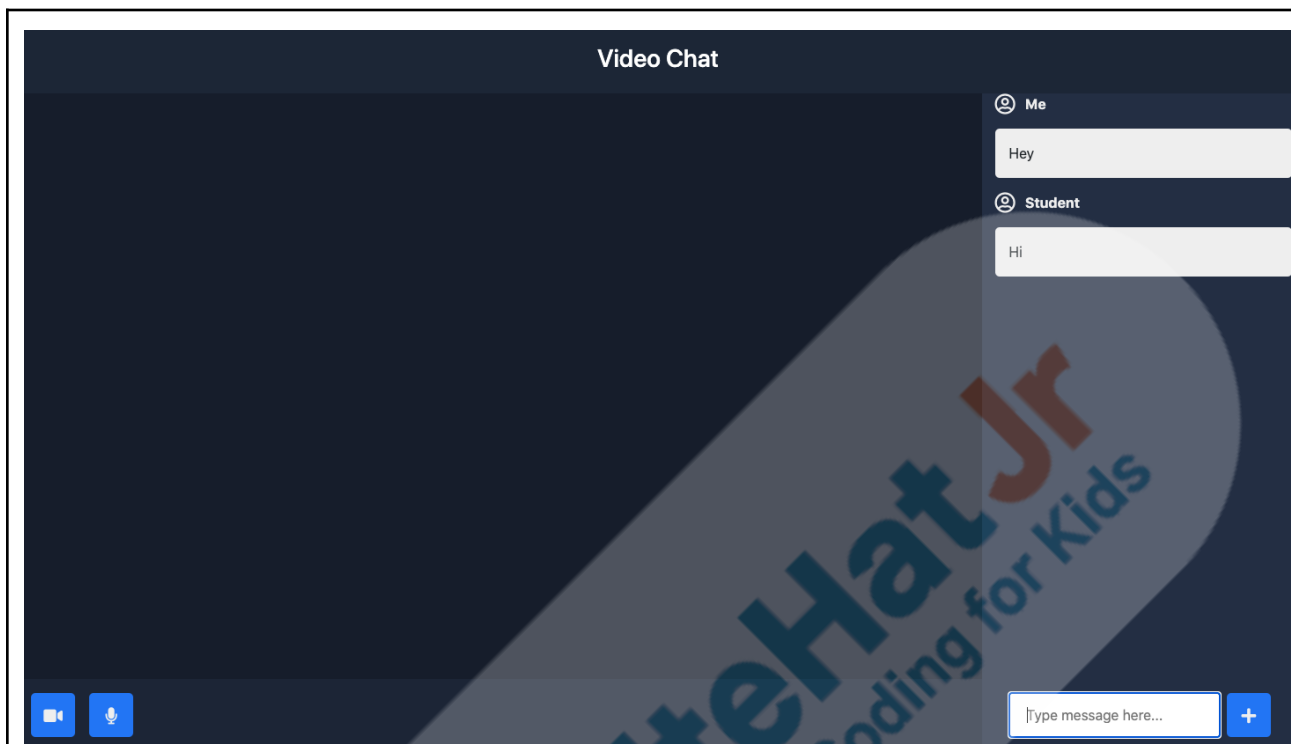
Student makes the change

```
server.listen(process.env.PORT || 3030);
```

Instead of **server.listen(3030)**, we would need to do **server.listen(process.env.PORT || 3030)**. This will let the app see if there is a port server is offering. If not, it will use 3030 by default. This way, it will run both locally and on Heroku.

<p>Now, let's update this small change on Github with the following commands -</p> <p>git add -A git commit -m "small change" git push</p> <p><i>Teacher helps the student in running the command</i></p> <p>Now that it is updated on Github, let's click on Deploy Branch again on the Heroku Dashboard page.</p>	<p><i>Student runs the command</i></p>
<p>Manual deploy</p> <p>Deploy the current state of a branch to this app.</p>	 <p>The screenshot shows the Heroku 'Manual deploy' interface. It includes a section 'Deploy a GitHub branch' with a dropdown menu set to 'main' and a 'Deploy Branch' button. Below this, a progress bar shows four steps: 'Receive code from GitHub', 'Build main', 'Release phase', and 'Deploy to Heroku', all marked with green checkmarks. A message at the bottom states 'Your app was successfully deployed.' with a 'View' button.</p>
<p>Awesome! Let's open the app again to see if it works this time!</p> <p><i>Teacher guides the student in opening the app</i></p>	<p><i>Student opens the app</i></p>

	
<p>Awesome! Just awesome! Our app works on the browser now!</p> <p>Let's test it!</p> <p><i>Teacher takes the link from the student and enters it in her browser to enter the same room. Teacher and student tries to chat</i></p>	<p><i>Student and teacher tries to chat!</i></p>



Awesome, simply Awesome isn't it? You just deployed your very first application and learnt about DevOps Engineering! Kudos to you!


Now, we will complete our video chat functionality in the coming classes, and you can go on to have video chat with your friends and family on an application that you built!



Teacher Guides Student to Stop Screen Share

WRAP UP SESSION - 5 Mins

Quiz time - Click on in-class quiz

Question	Answer
<p>What is cmd?</p> <p>A. Command Prompt B. Command Terminal</p>	D

C. Command D. All of the above	
Why do we use git add -A? A. Add Files to the repository B. Delete the file from the repository C. Change files to the repository D. None of the above	
What does git commit -m "small change" mean? A. Small change and commit B. Commit C. Small Change D. None of the above	
End the quiz panel	
FEEDBACK <ul style="list-style-type: none"> • Appreciate the students for their efforts in the class. • Ask the student to make notes for the reflection journal along with the code they wrote in today's class. 	
Teacher Action	Student Action
You get Hats off for your excellent work! In the next class, we will be getting into WebRTC!	<i>Make sure you have given at least 2 Hats Off during the class for:</i> <div> Creatively Solved Activities  +10 </div>

	<div>Great Question  +10</div> <div>Strong Concentration  +10</div>
<p>Project Discussion</p> <p>In this class, we learnt how we can deploy an app on Heroku. In this project, you'll be deploying the forum chat app to Heroku.</p> <p>John's app is up and running! He has been trying to deploy it on Heroku so that everyone can start using it, but he needs help from an expert. Help him deploy the app on Heroku.</p>	
<p>Teacher Clicks</p>	<div>✕ End Class</div>
ADDITIONAL ACTIVITIES	
<p>Additional Activities</p> <p><i>Encourage the student to write reflection notes in their reflection journal using markdown.</i></p> <p>Use these as guiding questions:</p> <ul style="list-style-type: none"> • What happened today? <ul style="list-style-type: none"> ○ Describe what happened. ○ The code I wrote. • How did I feel after the class? 	<p><i>The student uses the markdown editor to write her/his reflections in the reflection journal.</i></p>

- What have I learned about programming and developing games?
- What aspects of the class helped me? What did I find difficult?

ACTIVITY LINKS		
Activity Name	Description	Link
Teacher Activity 1	Previous Class Code	https://github.com/pro-whitehatjr/PRO-C217-ReferenceCode
Teacher Activity 2	Heroku Signup	https://signup.heroku.com/
Teacher Activity 3	Reference Code	https://github.com/pro-whitehatjr/PRO-C218-ReferenceCode
Student Activity 1	Previous Class Code	https://github.com/pro-whitehatjr/PRO-C217-ReferenceCode
Student Activity 2	Heroku Signup	https://signup.heroku.com/