

VIDEO CHAT APP - DevOps

**What is our GOAL for this CLASS?**

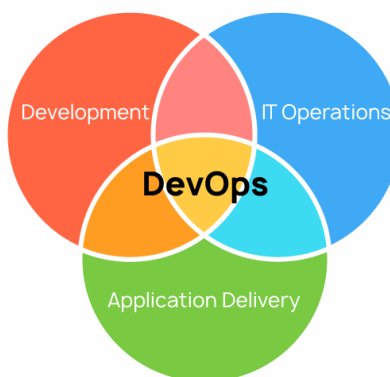
In this class we learnt about DevOps and how using DevOps we can host our video chat application on a remote device.

What did we ACHIEVE in the class TODAY?

- DevOPS
- Deploy web app on Heroku

Which CONCEPTS/ CODING BLOCKS did we cover today?

- Understanding about DevOps Engineering
- Creating a master github repository
- Introduction to Heroku!
- Deploying your first Web Application!



How did we DO the activities?



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. Today, we are going to host our video chat application on a remote device, or a remote server.

Activity:

1. The code that we completed in C-217, Let's clone it and then unzip the files from it!
1. Now let's create a new github repository. Push the code into the repository.
2. To create a personal token, we will first need to go to the settings.



Quick setup — if you've done this kind of thing before

 Set up in Desktop or **HTTPS** **SSH** `https://github.com/apoorvelous/video-chat-app.git` 

Get started by [creating a new file](#) or [uploading an existing file](#). We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).

...or create a new repository on the command line

```
echo "# video-chat-app" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/apoorvelous/video-chat-app.git
git push -u origin main
```

...or push an existing repository from the command line

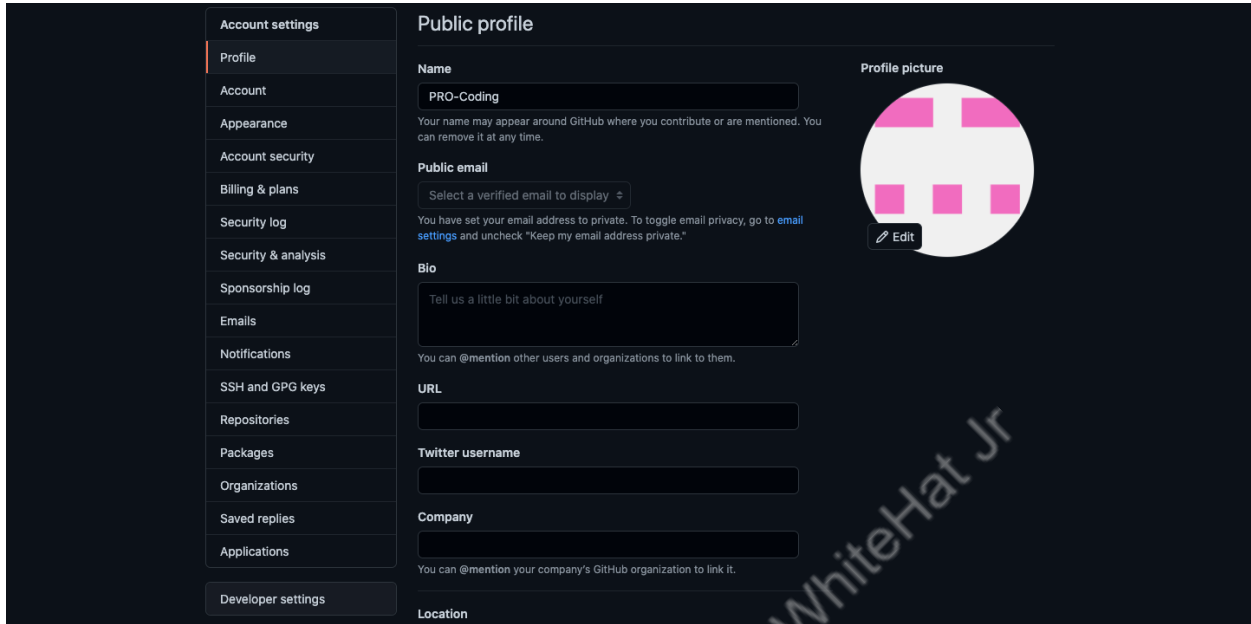
```
git remote add origin https://github.com/apoorvelous/video-chat-app.git
git branch -M main
git push -u origin main
```

...or import code from another repository

You can initialize this repository with code from a Subversion, Mercurial, or TFS project.

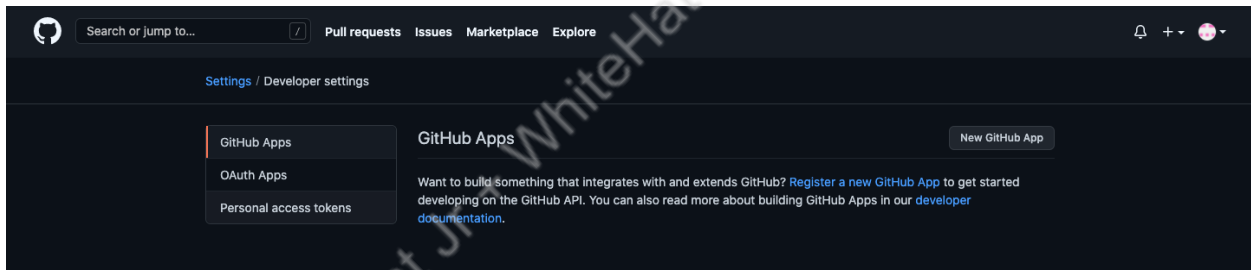
[Import code](#)

3. In settings:
 - Go to Developer Setting.



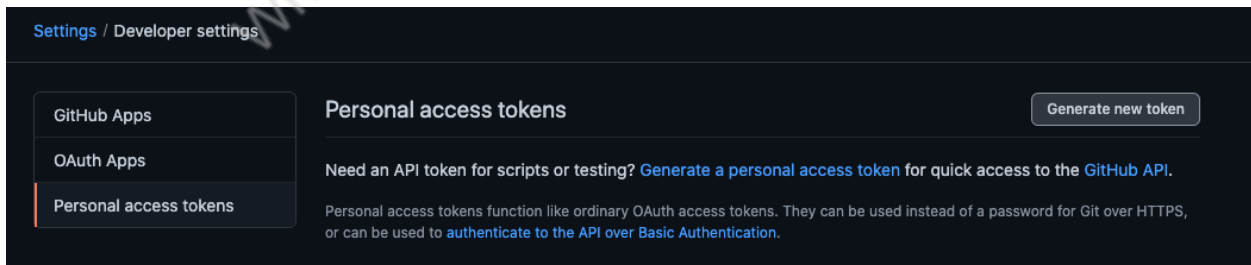
The screenshot shows the GitHub 'Public profile' settings page. On the left is a sidebar with a list of settings categories: Account settings, Profile, Account, Appearance, Account security, Billing & plans, Security log, Security & analysis, Sponsorship log, Emails, Notifications, SSH and GPG keys, Repositories, Packages, Organizations, Saved replies, Applications, and Developer settings. The 'Profile' category is selected. The main content area is titled 'Public profile' and contains several input fields: 'Name' (with the value 'PRO-Coding'), 'Public email' (with a dropdown to 'Select a verified email to display'), 'Bio' (with a text area containing 'Tell us a little bit about yourself'), 'URL', 'Twitter username', 'Company', and 'Location'. A 'Profile picture' section shows a circular placeholder with a grid pattern and an 'Edit' button. A diagonal watermark 'WhiteHat Jr' is visible across the right side of the page.

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



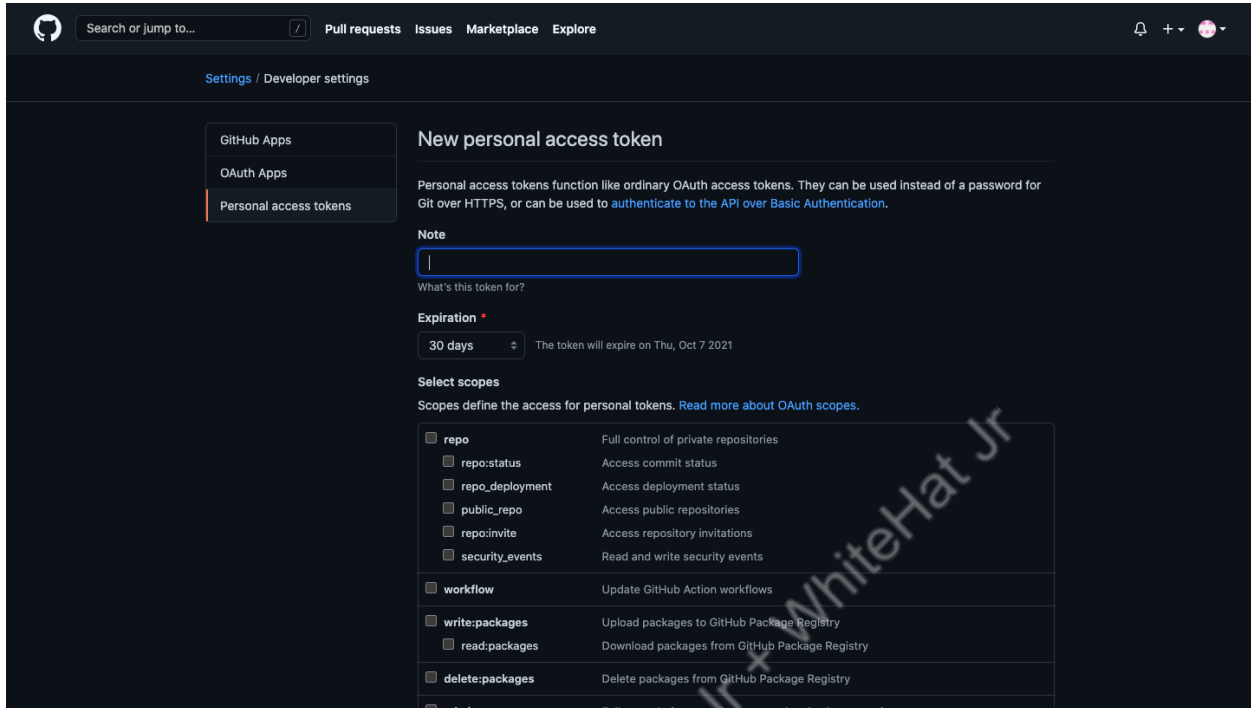
The screenshot shows the GitHub 'Settings / Developer settings' page. On the left is a sidebar with three options: 'GitHub Apps', 'OAuth Apps', and 'Personal access tokens'. The 'Personal access tokens' option is selected. The main content area is titled 'GitHub Apps' and contains a 'New GitHub App' button. Below the button is a paragraph of text: 'Want to build something that integrates with and extends GitHub? Register a new GitHub App to get started developing on the GitHub API. You can also read more about building GitHub Apps in our developer documentation.' A diagonal watermark 'WhiteHat Jr' is visible across the page.

- Next, click on Generate Access Token.



The screenshot shows the GitHub 'Settings / Developer settings' page, specifically the 'Personal access tokens' section. On the left is a sidebar with three options: 'GitHub Apps', 'OAuth Apps', and 'Personal access tokens'. The 'Personal access tokens' option is selected. The main content area is titled 'Personal access tokens' and contains a 'Generate new token' button. Below the button is a paragraph of text: 'Need an API token for scripts or testing? Generate a personal access token for quick access to the GitHub API. 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.' A diagonal watermark 'WhiteHat Jr' is visible across the page.

- You will see the following screen -



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

What's this token for?

Expiration

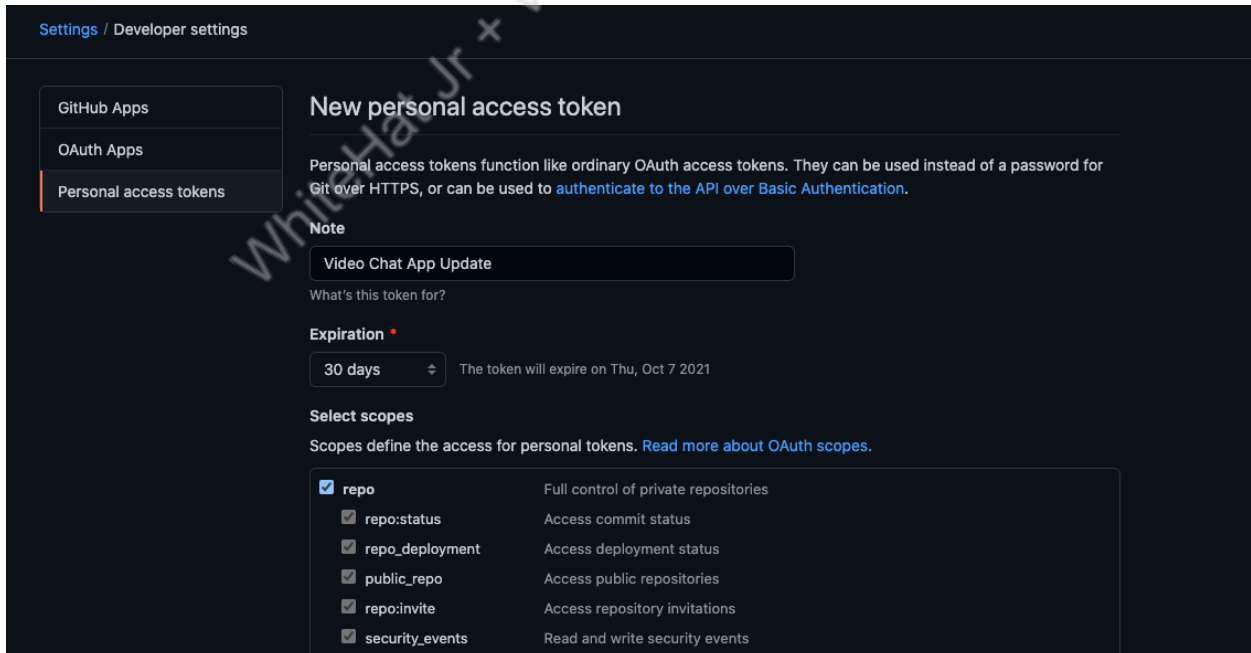
30 days The token will expire on Thu, Oct 7 2021

Select scopes

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

<input checked="" type="checkbox"/> repo	Full control of private repositories
<input checked="" type="checkbox"/> repo:status	Access commit status
<input checked="" type="checkbox"/> repo_deployment	Access deployment status
<input checked="" type="checkbox"/> public_repo	Access public repositories
<input checked="" type="checkbox"/> repo:invite	Access repository invitations
<input checked="" type="checkbox"/> security_events	Read and write security events
<input type="checkbox"/> workflow	Update GitHub Action workflows
<input type="checkbox"/> write:packages	Upload packages to GitHub Package Registry
<input type="checkbox"/> read:packages	Download packages from GitHub Package Registry
<input type="checkbox"/> delete:packages	Delete packages from GitHub Package Registry
<input type="checkbox"/> admin:repo_hook	Full control of repos and teams: read and write repo projects

- 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.](#)

<input checked="" type="checkbox"/> repo	Full control of private repositories
<input checked="" type="checkbox"/> repo:status	Access commit status
<input checked="" type="checkbox"/> repo_deployment	Access deployment status
<input checked="" type="checkbox"/> public_repo	Access public repositories
<input checked="" type="checkbox"/> repo:invite	Access repository invitations
<input checked="" type="checkbox"/> security_events	Read and write security events
<input type="checkbox"/> workflow	Update GitHub Action workflows
<input type="checkbox"/> write:packages	Upload packages to GitHub Package Registry
<input type="checkbox"/> read:packages	Download packages from GitHub Package Registry
<input type="checkbox"/> delete:packages	Delete packages from GitHub Package Registry
<input type="checkbox"/> admin:repo_hook	Full control of repos and teams: read and write repo projects

<input checked="" type="checkbox"/> admin:repo_hook	Full control of repository hooks
<input checked="" type="checkbox"/> write:repo_hook	Write repository hooks
<input checked="" type="checkbox"/> read:repo_hook	Read repository hooks

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

<input type="checkbox"/> admin:pgp_key	Full control of public user GPG keys (Developer Preview)
<input type="checkbox"/> write:pgp_key	Write public user GPG keys
<input type="checkbox"/> read:pgp_key	Read public user GPG keys

Generate token Cancel

- 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
```

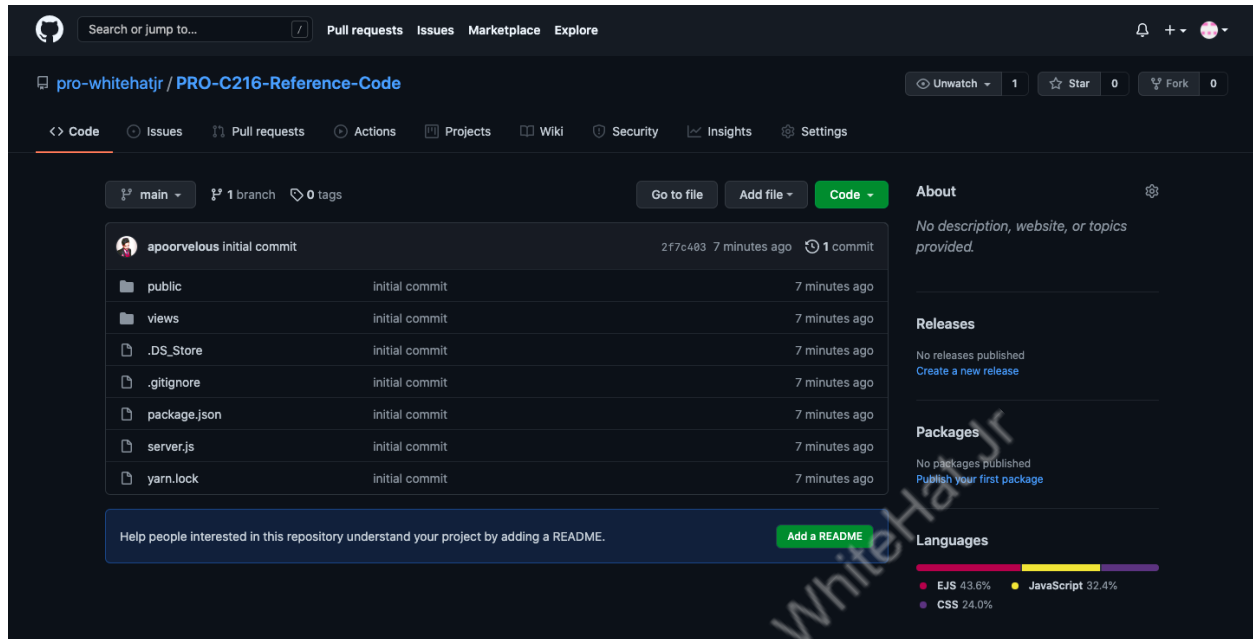
- On running the last command, which is **git push -u origin main**, you *may* encounter an error!

Run the following commands -

git config --global user.name "Your Username"

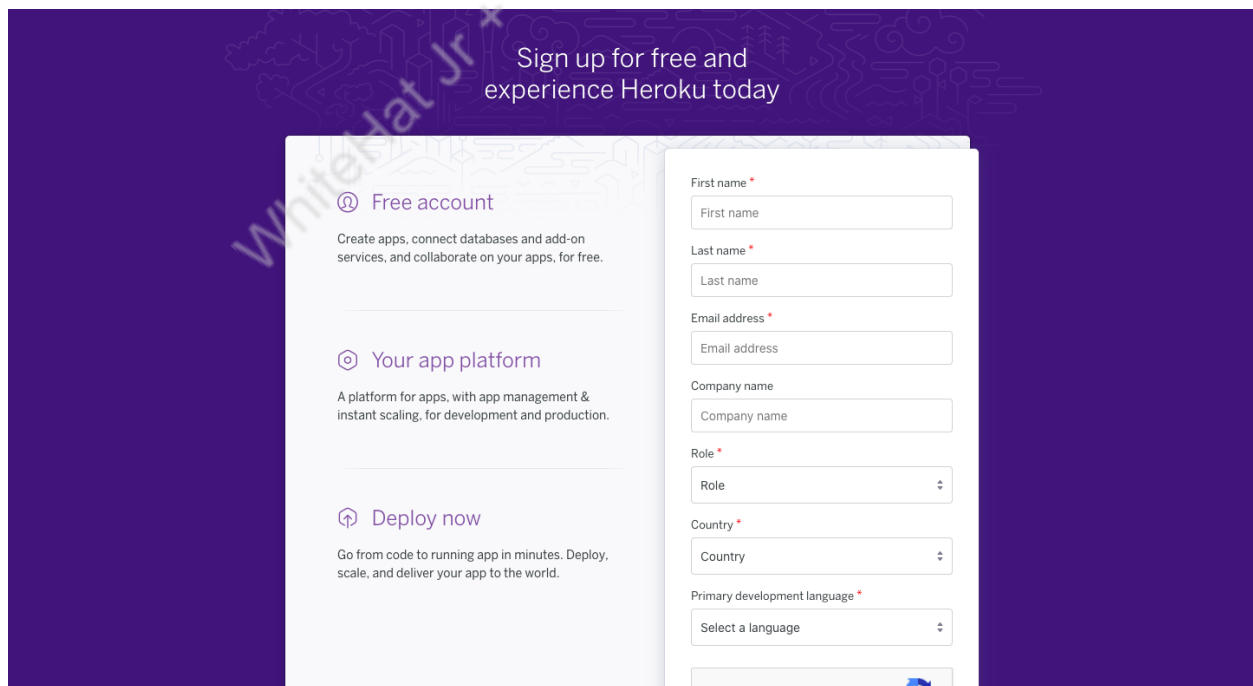
git config --global user.password "Personal Access Token"

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

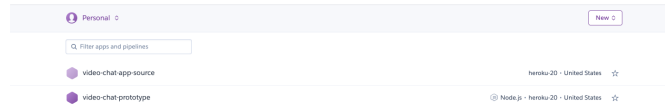


We can now use it to deploy our code into our remote device (or our remote server!)

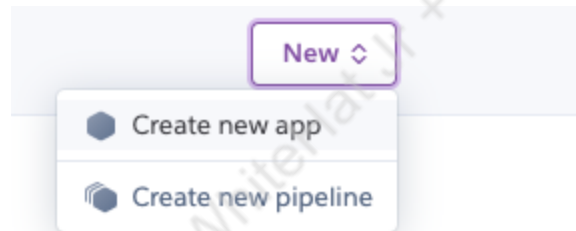
- There is a platform known as **Heroku**. It is a platform that offers remote devices to host/deploy applications, and these remote devices are known as **dyno**. Signin to Heroku.



7. Now, after logging into Heroku, you would see-



8. Click on the **New** button on the top right, and click on **create a new app**.



9. Enter a name for your app and click on the **Create App** button!

Create New App

App name

video-chat-app-216



video-chat-app-216 is available

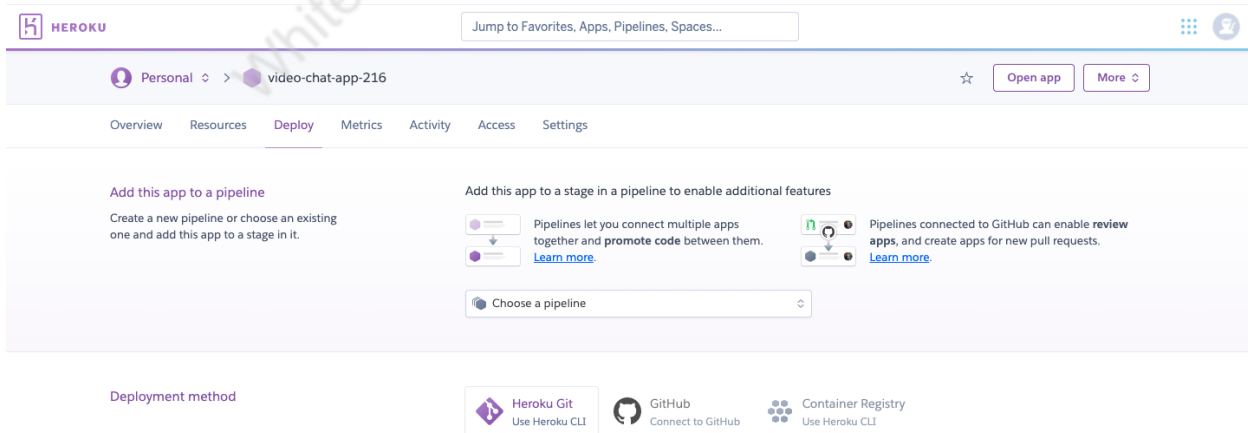
Choose a region

 United States

Add to pipeline...

Create app

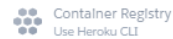
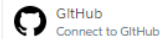
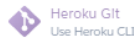
10. Once done, we will be prompted to the following screen -



The screenshot shows the Heroku dashboard for the app 'video-chat-app-216'. The top navigation bar includes the Heroku logo, a search bar, and a user profile icon. The main header shows 'Personal' and 'video-chat-app-216' with 'Open app' and 'More' buttons. The 'Deploy' tab is selected in the sidebar. The main content area has two sections: 'Add this app to a pipeline' and 'Add this app to a stage in a pipeline to enable additional features'. The first section explains how to create a new pipeline or add the app to an existing one. The second section explains how to connect the app to a pipeline to enable features like 'review apps'. Below these sections is a 'Choose a pipeline' dropdown menu. At the bottom, the 'Deployment method' section offers three options: 'Heroku Git' (Use Heroku CLI), 'GitHub' (Connect to GitHub), and 'Container Registry' (Use Heroku CLI).

11. Select Github from Deployment methods

Deployment method



Connect to GitHub

Connect this app to GitHub to enable code diffs and deploys.

View your code diffs on GitHub

Connect your app to a GitHub repository to see commit diffs in the activity log.

Deploy changes with GitHub

Connecting to a repository will allow you to deploy a branch to your app.

Automatic deploys from GitHub

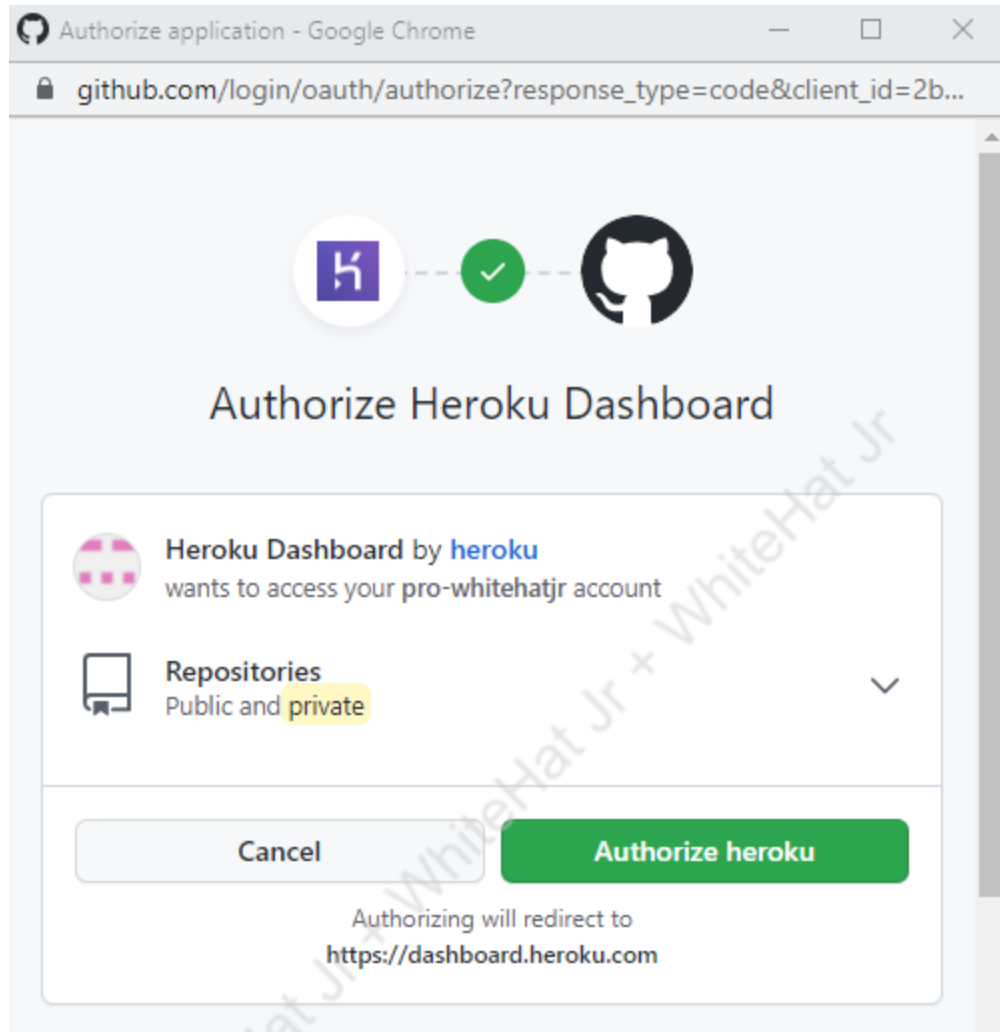
Select a branch to deploy automatically whenever it is pushed to.

Create review apps in pipelines

Pipelines connected to GitHub can enable **review apps**, and create apps for new pull requests. [Learn more](#).

Connect to GitHub

12. Connect with Github and Authorize it -



13. Search the repository name in which you pushed your code -

Connect to GitHub

Connect this app to GitHub to enable code diffs and deploys.

Search for a repository to connect to

 apoorvelous

Missing a GitHub organization? [Ensure Heroku Dashboard has team access.](#)

 apoorvelous/video-chat-app-v2

14. Connect the repository with Heroku

Automatic deploys

Enables a chosen branch to be automatically deployed to this app.



You can now change your main deploy branch from "master" to "main" for both manual and automatic deploys, please follow the instructions [here](#).

Enable automatic deploys from GitHub

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](#).

Choose a branch to deploy

main

☐ Wait for CI to pass before deploy

Only enable this option if you have a Continuous Integration service configured on your repo.

Enable Automatic Deploys

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

main

Deploy Branch

Awesome! Now, Heroku is linked to our Github Repository!

15. Click on **Deploy Branch** to deploy your code on Heroku.

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

main

Deploy Branch

Receive code from GitHub



Build main 21f5f000



Release phase



Deploy to Heroku

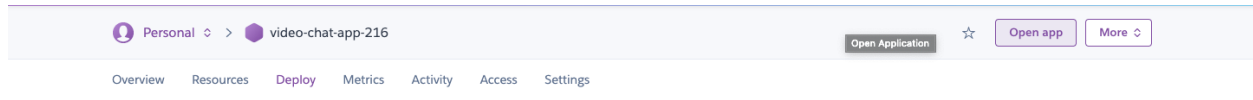


Your app was successfully deployed.

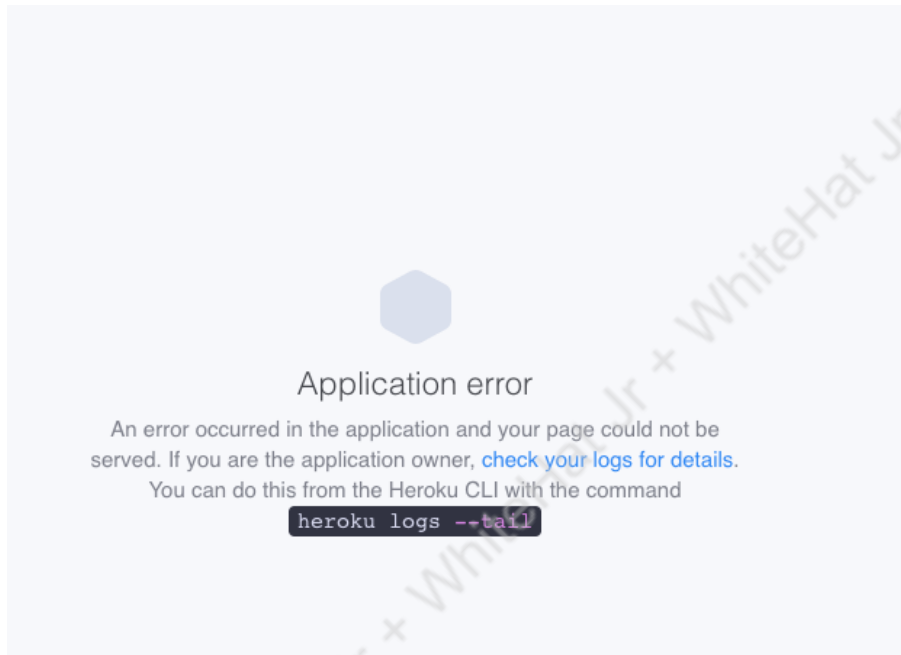
[View](#)

16. You'll notice it has already installed **NodeJS** and run **yarn install** on the remote device for you.

17. With this, our App is deployed. There is a button on the web page that says:



18. Let's click on **Open App** to see our App Deployed!



It shows an Application Error! It also gives us a command **heroku logs --tail** to investigate what the error was! Let's try and run that in **cmd/terminal** -

```

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

```

19. 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 -

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

20. Now, let's push this small change again with the following commands and deploy it on Heroku again-

```

git add -A
git commit -m "small change"
git push

```

And then -

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)



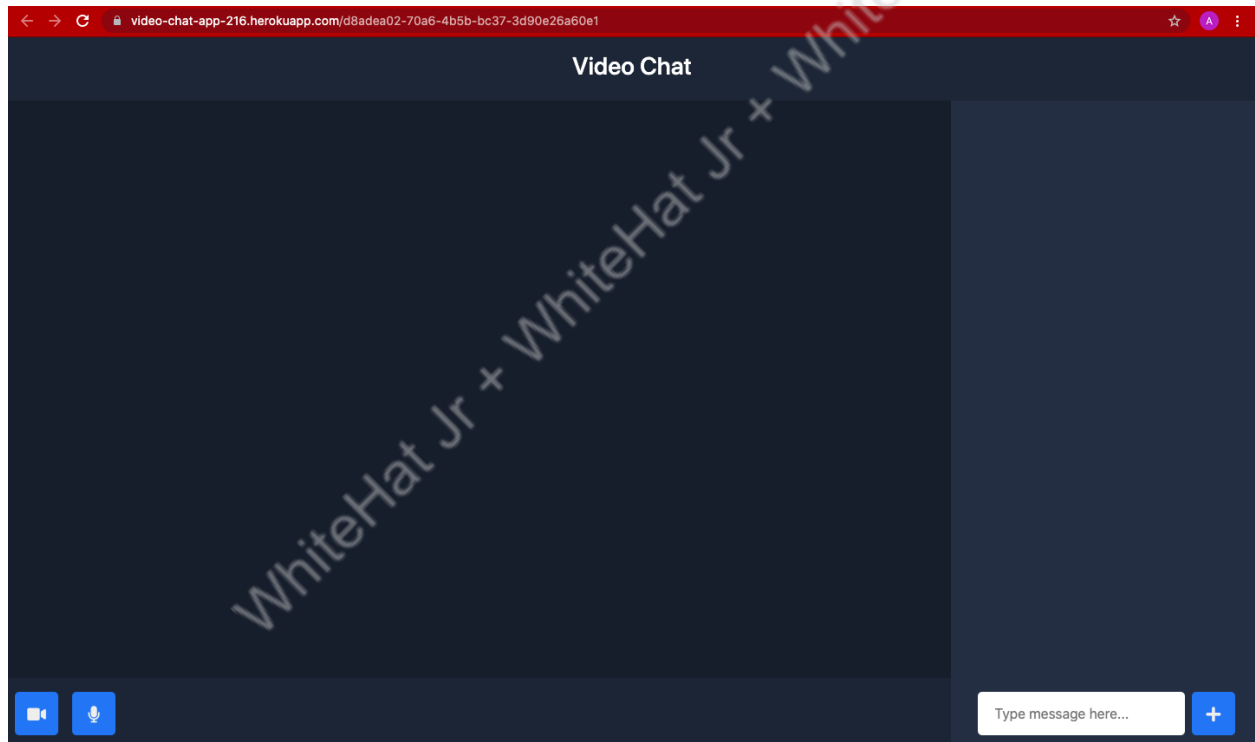
Release phase



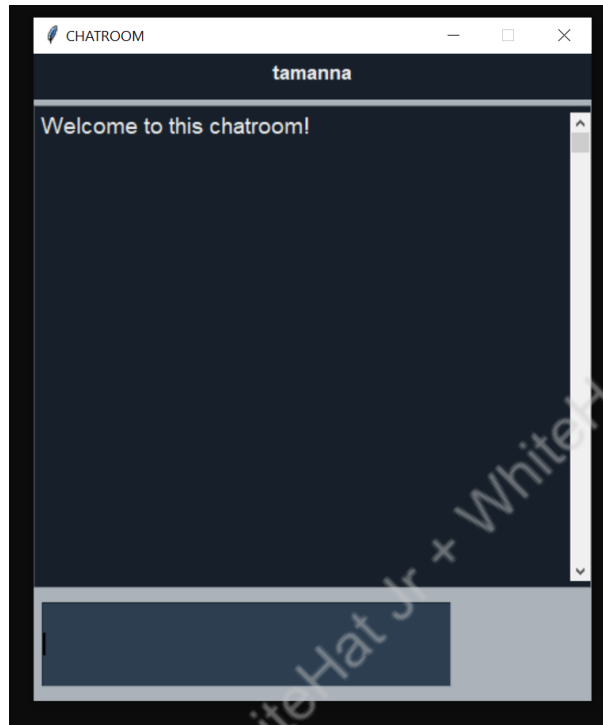
Deploy to Heroku



Your app was successfully deployed.

[View](#)

21. The chat window opens up and looks like -



What's NEXT?

In the next class, we will complete the chat interface of this app and have a fully functional GUI based chat applications

Expand Your Knowledge:

Explore more about DevOps library [here](#)