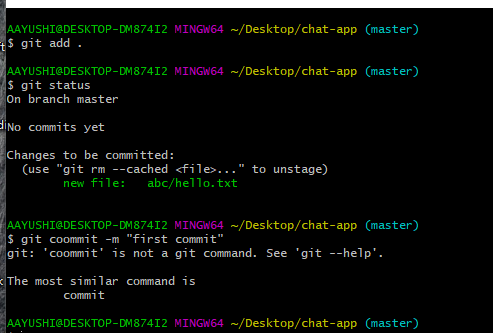
**Chat-App**

**Steps:-**

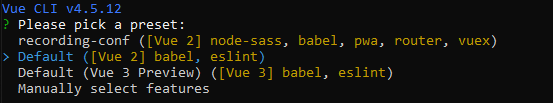
1) Created Repo



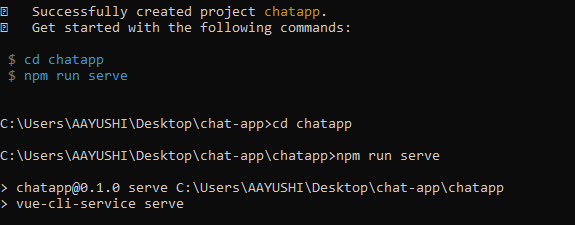
2) Created Vue App:

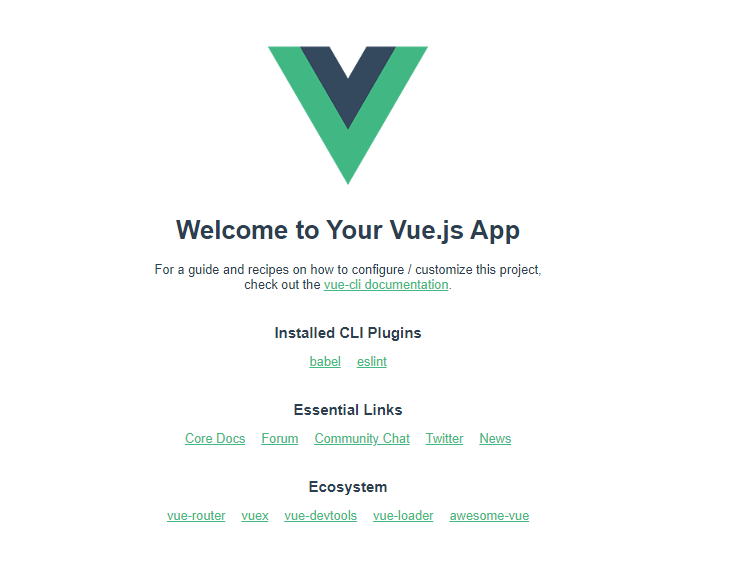
* Here we used Vue-2 to create our app.





* Navigate to our app and run it.





3) Installed dependencies



4) Created UI

* **Main.js:-** Here we import bootstrap.css and bootstrap-vue.css in-order to use functionality of bootstrap-vue.

****

* **ChatMeassagebox.vue**

****

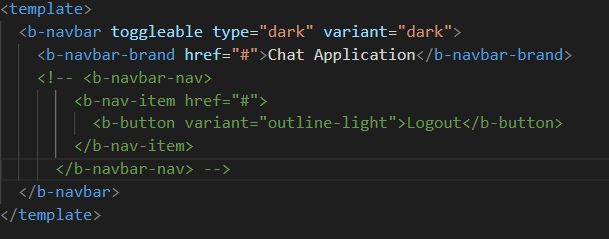
* **ChatChatbox.vue**

****

* **ChatHeader.vue**

****

* **ChatNavbar.vue**

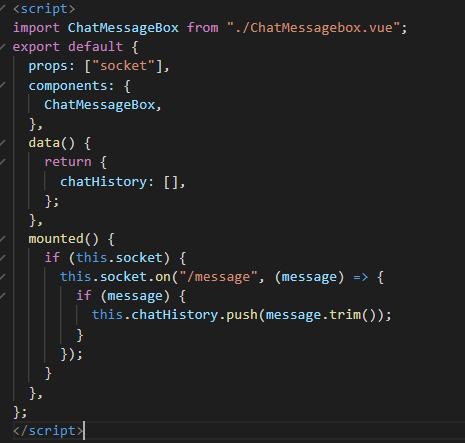
****

5) WebSocket Implementation

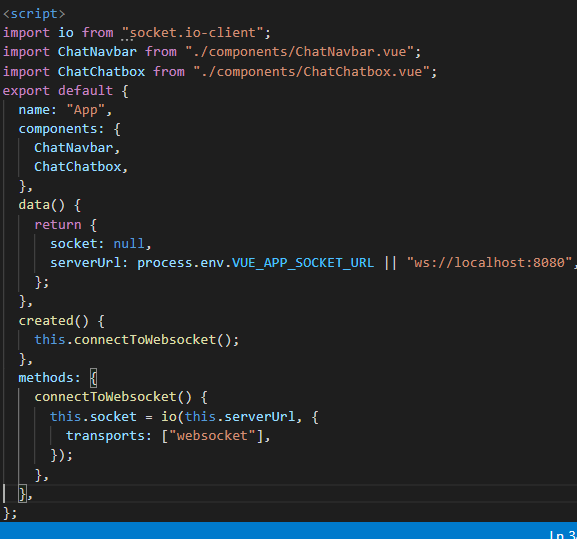
* **ChatMessagebox.vue**: Here when we click on the send button, the message user writes in the input-box , it goes through socket, we emit ‘/chat’ event which we have defined in this file.

|  |
| --- |
| <script> |
|  | export default { |
|  | props: ["socket"], |
|  | data() { |
|  | return { |
|  | newMsg: "", |
|  | }; |
|  | }, |
|  | methods: { |
|  | sendMessage() { |
|  | if (this.newMsg) { |
|  | this.socket.emit("/chat", { message: this.newMsg }); |
|  | this.newMsg = ""; |
|  | } |
|  | }, |
|  | }, |
|  | }; |
|  | </script> |

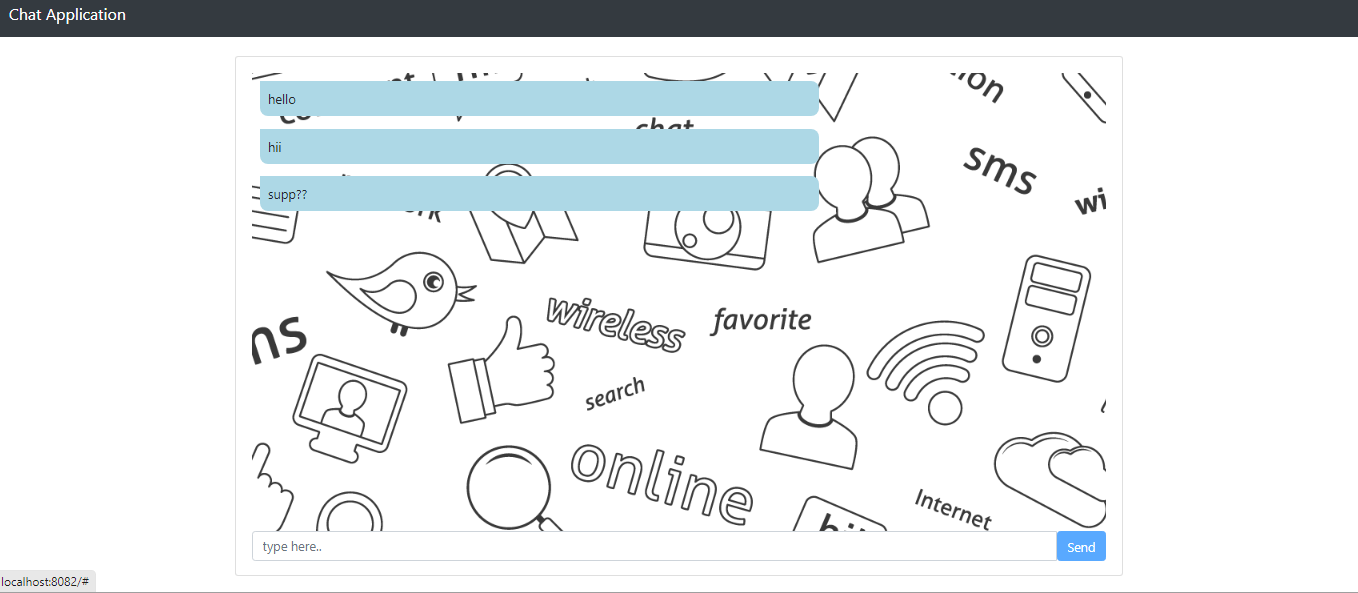
* **ChatChatBox.vue:** Here our app loads first in the browser, because we used mounted function. Here in ‘socket.on’ method all the messages are pushed inside the chatHistory array. Through socket we listen ‘/message’ event from the server. Here we used trim(), which removes white-spaces.

****

* **App.vue:** We create socket here. env file contains sensitive information about serverUrl. Here connectToWebSocket() is used to connect client to golang server.

****

**Output**

****

**References:**

<https://socket.io/docs/v3/client-api/index.html>