ConexusHub

Sara Martínez, Rubén Lora, Eric Gómez,
Julie Villegas

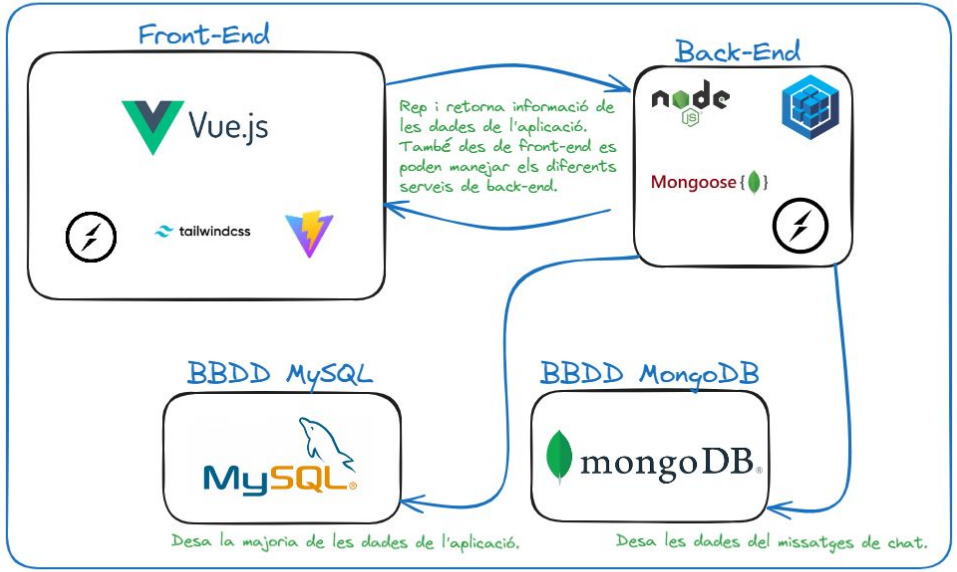
conexushub.cat

01

ESTRUCTURA DEL PROJECTE

Docker

Per executar i treballar en el projecte s'ha fet un Docker on es crea les imatges de cada part i es connecta tot.



SPRINTS I FUNCIONALITATS

- **MVP:**
  - Acabar les funcionalitats i corregir errors.
  - Implementar les noves funcionalitats que volíem.
- **Sprint 1:**
  - Replantejament del projecte.
  - Refer el Back implementar Sequelize.
- **Sprint 2:**
  - Volem que hi hagi totes les funcions de tots els diferents tipus d'usuaris.
  - I que tot el relacionat amb producció funcioni correctament amb Docker i GitHub Actions.
- **Sprint 3:**
  - Millorar el disseny de totes les pàgines.
  - I que els Sockets amb cantina i professors funcionin correctament.

PROBLEMES I **SOLUCIONS**

Alguns dels problemes que hem tingut després de l'MVP:

- Refer tota la màquina de producció.
- Canviar tot el back-end de Node per implementar Sequelize.
- Redissenyar tota la interfície amb els diferents tipus de perfils.
- Refer tot el sistema de xat.

Requisits Servidor

Parts més **destacables**

Això seria el xat entre professor i cantina.

```
const retryMessage = async (message, index) => {
  if (message.failed && chatId.value) {
    try {
      messages.value[index].failed = false;
      messages.value[index].sending = true;

      if (!message.hasLinks) {
        const links = detectLinks(message.message);
        message.hasLinks = links.length > 0;
        message.links = links;
        message.linkPreviews = [];
      }

      const result = await chatManager.sendMessage(
        chatId.value,
        message.userId,
        message.message
      );

      if (result.interaction && result.interaction.length > 0) {
        const newMsg = result.interaction[result.interaction.length - 1];
        messages.value[index] = {
          ...messages.value[index],
          id: newMsg._id,
          hasLinks: newMsg.hasLinks || message.hasLinks,
          links: newMsg.links || message.links,
          linkPreviews: newMsg.linkPreviews || [],
          sending: false,
          failed: false,
          local: true,
        };
      } else {
        messages.value[index].sending = false;
        messages.value[index].failed = false;
      }
    } catch (error) {
      console.error("Error al reenviar missatge:", error);
      messages.value[index].failed = true;
      messages.value[index].sending = false;
    }
  }
};
```

Parts més **destacables**

I això serien les funcions del xat entre professors.

```
const setupSocketEventHandlers = () => {
  socket.value.on("new_message", (data) => {
    let messageData = null;

    if (
      data.interaction &&
      Array.isArray(data.interaction) &&
      data.interaction.length > 0
    ) {
      const lastInteraction = data.interaction[data.interaction.length - 1];
      messageData = {
        id: lastInteraction._id,
        userId: lastInteraction.teacherId,
        message: lastInteraction.message,
        hasLinks: lastInteraction.hasLinks || false,
        links: lastInteraction.links || [],
        linkPreviews: lastInteraction.linkPreviews || [],
        timestamp: new Date(lastInteraction.date),
      };
    } else if (data.message && data.userId) {
      messageData = {
        id: data.id || Date.now().toString(),
        userId: data.userId,
        message: data.message,
        hasLinks: data.hasLinks || false,
        links: data.links || [],
        linkPreviews: data.linkPreviews || [],
        timestamp: data.timestamp || new Date(),
      };
    } else if (
      data.chatId &&
      data.interaction &&
      !Array.isArray(data.interaction)
    ) {

      const msg = data.interaction;
      messageData = {
        id: msg._id || Date.now().toString(),
        userId: msg.teacherId,
        message: msg.message,
        hasLinks: msg.hasLinks || false,
        links: msg.links || [],
        linkPreviews: msg.linkPreviews || [],
        timestamp: new Date(msg.date) || new Date(),
      };
    }
```