PROGRAM:

FRONTEND:

APP.JS:

```
import{RouterProvider,createBrowserRouter}from"react-router-dom";
import './App.css';
import TaskCreate from "./TaskCreate";
import User from"./User";
import Edit from "./Edit";
function App() {
const route = createBrowserRouter([
{ path:"/Assign",
element:<TaskCreate/>
},
{
path:"/Task",
element:<User/>
},
{
path:"/Edit/:id",
element:<Edit/>
}
])
return (
<div className="App">
<RouterProvider router={route}></RouterProvider>
</div>
);
}
export default App;
```

EDIT JS:

```
import React, { useState } from 'react';
import { Container, Grid, TextField, Button, CardHeader, CardContent, Card } from
"@mui/material";
import axios from 'axios';
import { useNavigate ,useParams } from 'react-router-dom';
export default function Edit() {
const [progress, setProgress] = useState("");
const navigate = useNavigate();
const{id} = useParams();
const handleSend = async (e) => {
e.preventDefault();
try {
const output = { progress };
const response = await fetch(`http://localhost:8080/update/${id}`, {
```

```
method: 'PUT',
headers: {
'Content-Type': 'application/json',
},
body: JSON.stringify(output),
});
if (response.ok) {
navigate("/Task");
} else {
const errorData = await response.json();
console.error("Error:", errorData);
}
} catch (error) {
console.error("Error:", error);
}
};
return (
<Container>
<Grid container spacing={4} justifyContent="center" marginTop="20px">
<Grid item xs={12} sm={6} md={9}>
<Card>
<CardHeader title="Task Assignment" />
<CardContent>
<TextField
variant='outlined'
label="Progression"
value={progress}
onChange={(e) => setProgress(e.target.value)} // Update the progress state
/><br></br>
<br></br>
<Button
variant="contained"
onClick={handleSend} // Call handleSend directly on button click
>
Send
</Button>
</CardContent>
</Card>
</Grid>
</Grid>
</Container>
)
}
```

USERJS:

```
import React, { useEffect, useState } from 'react';
import { Container, Grid, Card, CardHeader, Typography, IconButton, TextField,CardContent }
from "@mui/material";
import EditIcon from '@mui/icons-material/Edit';
import { useNavigate } from 'react-router-dom';
export default function Admin() {
const [show, setshow] = useState([]);
useEffect(() => {
fetch("http://localhost:8080/get")
.then(res => res.json())
.then(res => setshow(res));
}, []);
const navigate = useNavigate();
return (
show.map((task) => (
<Container key={task.id}>
<Grid container spacing={4} justifyContent="center" marginTop="20px">
<Grid item xs={12} sm={6} md={9}>
<Card>
<CardHeader
title={task.task}
action={
<IconButton onClick={() => navigate(`/edit/${task.id}`)}>
<EditIcon />
</IconButton>
}
/>
<CardContent>
<Typography>
Task Description: {task.taskdescp}<br></br>
Team Name: {task.teamname}<br></br>
Progress: {task.progress}<br></br>
Start Date: {task.startdate}<br></br>
End Date: {task.enddate}<br></br>
</Typography>
</CardContent>
</Card>
</Grid>
</Grid>
</Container>
))
);
}
```

BACKEND:

CONTROLLER:

```
package com.example.Backend.Controller;
import com.example.Backend.Entity.Task;
import com.example.Backend.Repository.TaskRepository;
import com.example.Backend.Service.TaskService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.*;
import java.util.List;
@RestController
@CrossOrigin
public class TaskController {
@Autowired
public TaskService taskService;
@Autowired
public TaskRepository taskRepository;
@PostMapping("/add")
public String addTask(@RequestBody Task task){
taskService.addTask(task);
return "Success";
}
@GetMapping("/get")
public List<Task> getTask(Task task){
return taskService.getTask(task);
}
@PutMapping("/update/{id}")
public Task updateTask(@RequestBody Task newTask, @PathVariable Long id){
return taskRepository.findById(id).map(task-> {
task.setProgress(newTask.getProgress());
return taskRepository.save(task);
}).orElseThrow(()->new RuntimeException());
}
}
```

ENTITY:

```
package com.example.Backend.Entity;
import jakarta.persistence.*;
@Entity
@Table(name = "taskdetails")
public class Task {
@Id
@GeneratedValue(strategy = GenerationType.AUTO)
private Long id;
private String task;
private String taskdescp;
```

```
 private String teamname;
 private String startdate;
 private String enddate;
 private String progress;
 public Task(Long id, String task, String taskdescp, String teamname, String startdate, String
 enddate, String progress) {
 this.id = id;
 this.task = task;
 this.taskdescp = taskdescp;
 this.teamname = teamname;
 this.startdate = startdate;
 this.enddate = enddate;
 this.progress = progress;
 }
 public Task() {
 }
 public Long getId() {
 return id;
 }
 public void setId(Long id) {
 this.id = id;
 }
 public String getTask() {
 return task;
 }
 public void setTask(String task) {
 this.task = task;
 }
 public String getTaskdescp() {
 return taskdescp;
 }
 public void setTaskdescp(String taskdescp) {
this.taskdescp = taskdescp;
 }
 public String getTeamname() {
 return teamname;
 }
 public void setTeamname(String teamname) {
 this.teamname = teamname;
 }
 public String getStartdate() {
 return startdate;
 }
 public void setStartdate(String startdate) {
```

```
this.startdate = startdate;
}
public String getEnddate() {
return enddate;
}
public void setEnddate(String enddate) {
this.enddate = enddate;
}
public String getProgress() {
return progress;
}
public void setProgress(String progress) {
this.progress = progress;
}
}
```

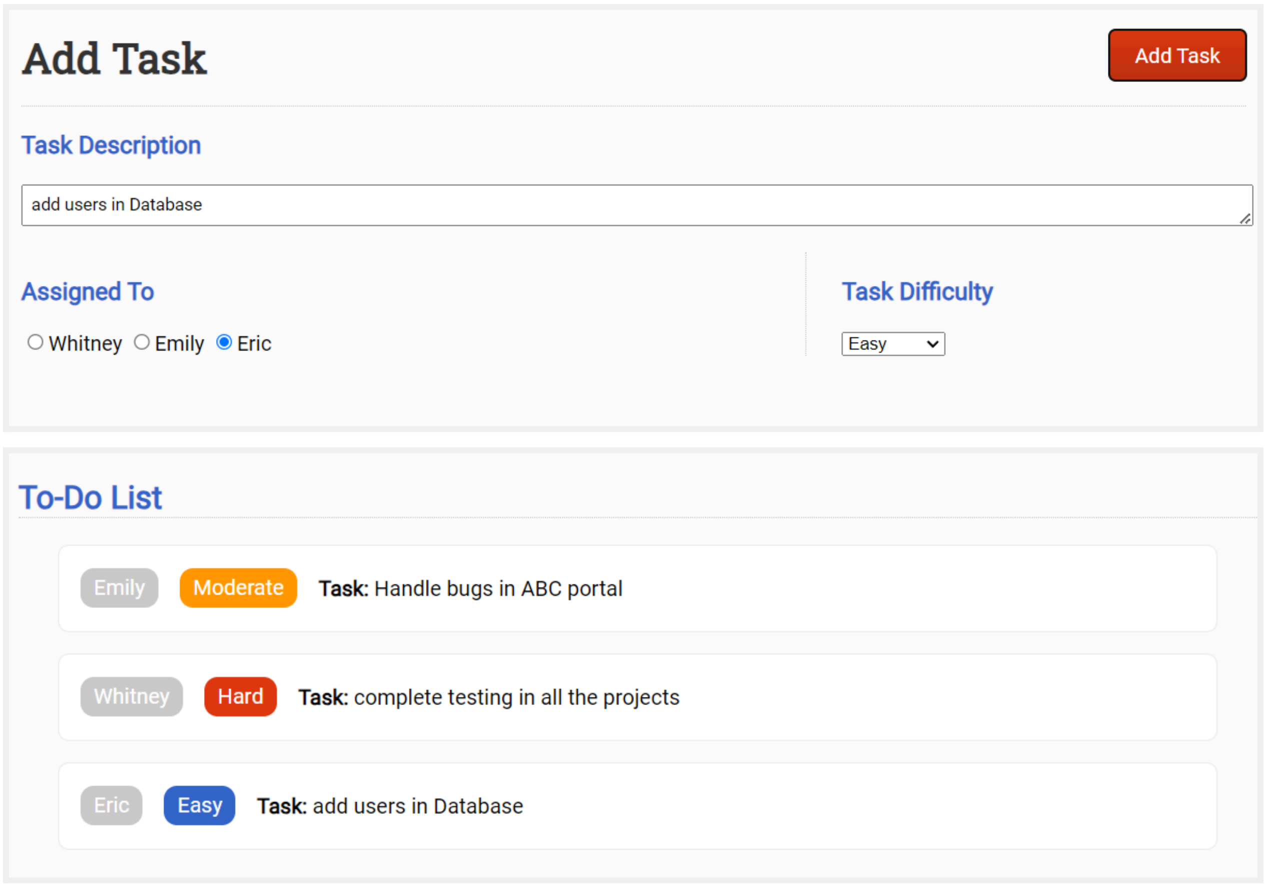
SERVICE:

```
package com.example.Backend.Service;
import com.example.Backend.Entity.Task;
import com.example.Backend.Repository.TaskRepository;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import java.util.List;
@Service
public class TaskServiceImpl implements TaskService{
@Autowired
public TaskRepository taskRepository;
@Override
public Task addTask(Task task) {
return taskRepository.save(task);
}
@Override
public List<Task> getTask(Task task) {
return taskRepository.findAll();
}
@Override
public Task updateTask(Task task) {
return taskRepository.save(task);
}
}
```

REPOSITORY:

```
package com.example.Backend.Repository;
import com.example.Backend.Entity.Task;
import org.springframework.data.jpa.repository.JpaRepository;
public interface TaskRepository extends JpaRepository<Task,Long> { }
```

OUTPUT :



Add Task

Add Task

Task Description

add users in Database

Assigned To

Whitney Emily [✓ Eric]

Task Difficulty

Easy

To-Do List

Emily Moderate **Task:** Handle bugs in ABC portal

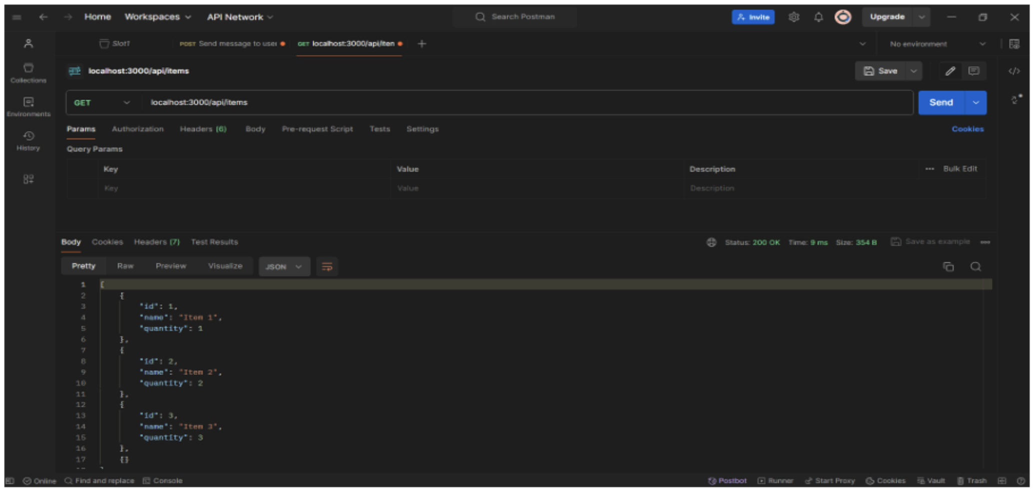
Whitney Hard **Task:** complete testing in all the projects

Eric Easy **Task:** add users in Database

PROGRAM:

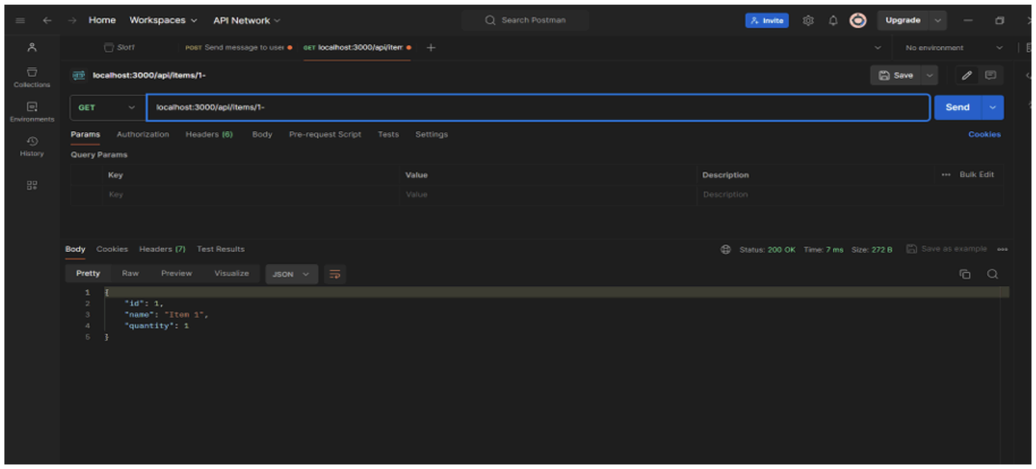
```
 const express = require('express');
 const bodyParser = require('body-parser');
 const app = express();
 const port = 3000;
 // Parse JSON bodies for POST and PUT requests
 app.use(bodyParser.json());
 let items = [
 { id: 1, name: 'Item 1',quantity:1 },
 { id: 2, name: 'Item 2' ,quantity:2},
 { id: 3, name: 'Item 3',quantity:3},
 ];
 // GET all items
 app.get('/api/items', (req, res) => {
 res.json(items);
 });
 // GET single item by ID
 app.get('/api/items/:id', (req, res) => {
 const id = parseInt(req.params.id);
 const item = items.find(item => item.id === id);
 if (!item) return res.status(404).send('Item not found');
 res.json(item);
 });
 // POST new item
 app.post('/api/items', (req, res) => {
 const newItem = req.body;
 items.push(newItem);
 res.status(201).json(newItem);
 });
 // PUT update item by ID
 app.put('/api/items/:id', (req, res) => {
 const id = parseInt(req.params.id);
 const updatedItem = req.body;
 const index = items.findIndex(item => item.id === id);
 if (index ===-1) return res.status(404).send('Item not found');
 items[index] = updatedItem;
 res.json(updatedItem);
 });
 // DELETE item by ID
 app.delete('/api/items/:id', (req, res) => {
 const id = parseInt(req.params.id);
 items = items.filter(item => item.id !== id);
res.sendStatus(204);
});
// PATCH update item partially by ID
app.patch('/api/items/:id', (req, res) => {
const id = parseInt(req.params.id);
const updatedFields = req.body;
const index = items.findIndex(item => item.id === id);
if (index ===-1) return res.status(404).send('Item not found');
items[index] = { ...items[index], ...updatedFields };
res.json(items[index]);
});
// Start the server
app.listen(port, () => {
console.log(`Server running at http://localhost:${port}`);
});
```

OUTPUT:


GET localhost:3000/api/items

Status: 200 OK Time: 9 ms Size: 354 B

```
[
    {
        "id": 1,
        "name": "Item 1",
        "quantity": 1
    },
    {
        "id": 2,
        "name": "Item 2",
        "quantity": 2
    },
    {
        "id": 3,
        "name": "Item 3",
        "quantity": 3
    },
    {}
```


GET localhost:3000/api/items/1-

Status: 200 OK Time: 7 ms Size: 272 B

```
{
    "id": 1,
    "name": "Item 1",
    "quantity": 1
}
```

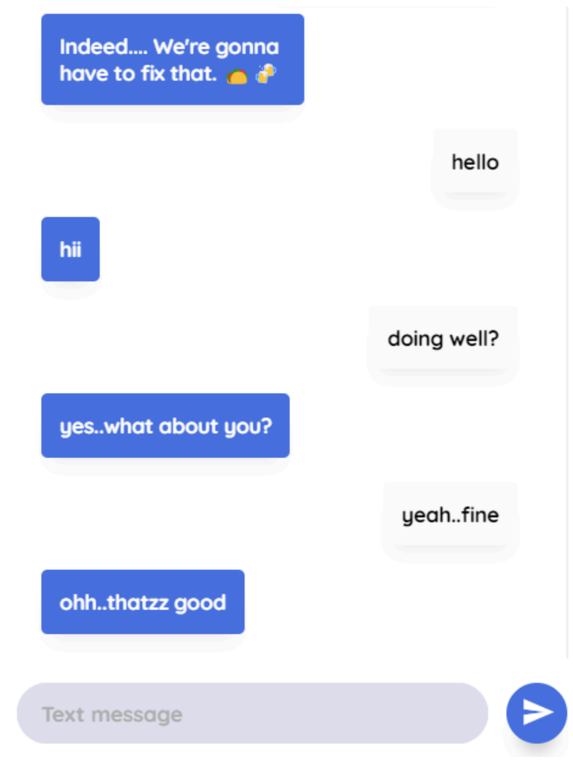
PROGRAM:
```
 <!DOCTYPE html>
 <html lang="en">
 <head>
 <meta charset="UTF-8">
 <meta name="viewport" content="width=device-width, initial-scale=1.0">
 <title>Chat Between User 1 and User 2</title>
 <link rel="stylesheet" href="styles.css">
 <style>
 body {
 font-family: Arial, sans-serif;
 }
 .chat-container {
 display: flex;
 justify-content: space-between;
 margin: 20px auto;
 max-width: 800px;
 }
 .user {
 flex: 1;
 border: 10px solid black;
 border-radius: 10px;
 padding: 10px;
 margin-right: 20px; /* Add margin-right for space between user divs */
 }
 .chat-box {
 height: 200px;
 overflow-y: auto;
 margin-bottom: 10px;
 padding: 10px;
 border: 1px solid black; /* Add border for the message box */
 border-radius: 5px; /* Add border radius for the message box */
 }
 .message-input {
 width: calc(100%- 70px);
 padding: 5px;
 margin-right: 5px;
 border: 1px solid #ccc;
 border-radius: 5px;
 outline: none;
 }
.send-btn {
 width: 60px;
```

```
padding: 5px;
cursor: pointer;
background-color: #4CAF50;
color: white;
border: none;
border-radius: 5px;
text-align: center;
text-decoration: none;
display: inline-block;
font-size: 16px;
}
.message.sent {
text-align: left;
}
.message.received {
text-align: right;
}
.message.sent::before {
content: "User 1: ";
}
.message.received::before {
content: "User 2: ";
}
</style>
</head>
<body>
<div class="chat-container">
<div class="user" id="user1">
<h2>User 1</h2>
<div class="chat-box" id="user1-chat-box"></div>
<input type="text" class="message-input" id="user1-message-input" placeholder="Type
a message..."><br/><br/>
<button class="send-btn" id="user1-send-btn">Send</button>
</div>
</div>
<div class="user" id="user2">
<h2>User 2</h2>
<div class="chat-box" id="user2-chat-box"></div>
<input type="text" class="message-input" id="user2-message-input" placeholder="Type a
message..."><br/><br/>
<button class="send-btn" id="user2-send-btn">Send</button>
</div>
<script src="script.js">
document.addEventListener("DOMContentLoaded", function() {
```

```
const user1ChatBox = document.getElementById("user1-chat-box");
const user1MessageInput = document.getElementById("user1-message-input");
const user1SendButton = document.getElementById("user1-send-btn");
const user2ChatBox = document.getElementById("user2-chat-box");
const user2MessageInput = document.getElementById("user2-message-input");
const user2SendButton = document.getElementById("user2-send-btn");
user1SendButton.addEventListener("click", function() {
sendMessage(user1MessageInput.value, user1ChatBox, user2ChatBox);
user1MessageInput.value = "";
});
user2SendButton.addEventListener("click", function() {
sendMessage(user2MessageInput.value, user2ChatBox, user1ChatBox);
user2MessageInput.value = "";
});
function sendMessage(message, senderChatBox, receiverChatBox) {
if (message.trim() !== "") {
displayMessage(message, senderChatBox, "sent");
// Display the message in the receiver's chat box
displayMessage(message, receiverChatBox, "received");
}
}
function displayMessage(message, chatBox, type) {
const messageElement = document.createElement("div");
messageElement.textContent = message;
messageElement.classList.add("message", type);
chatBox.appendChild(messageElement);
chatBox.scrollTop = chatBox.scrollHeight;
}
});
</script>
</body>
</html>
```

OUTPUT :

user1:



Indeed.... We're gonna have to fix that. 🌮 🍻

hello

hii

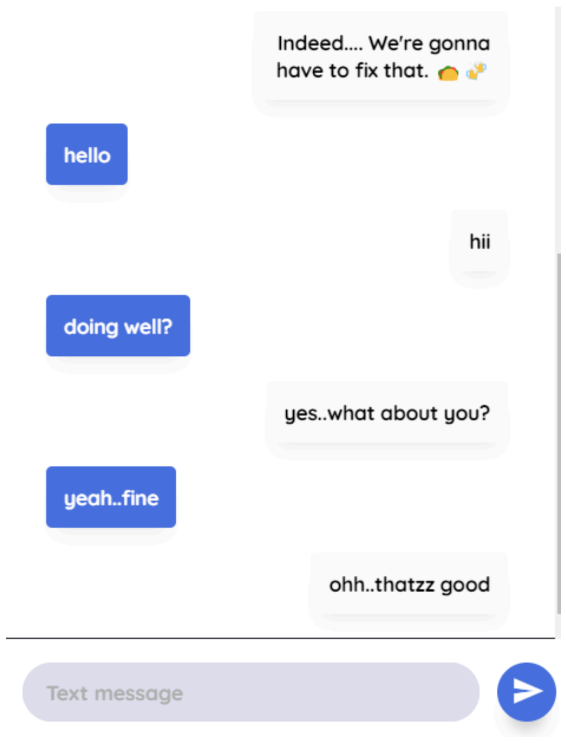
doing well?

yes..what about you?

yeah..fine

ohh..thatzz good

User2:



Indeed.... We're gonna have to fix that. 🌮 🍻

hello

hii

doing well?

yes..what about you?

yeah..fine

ohh..thatzz good