

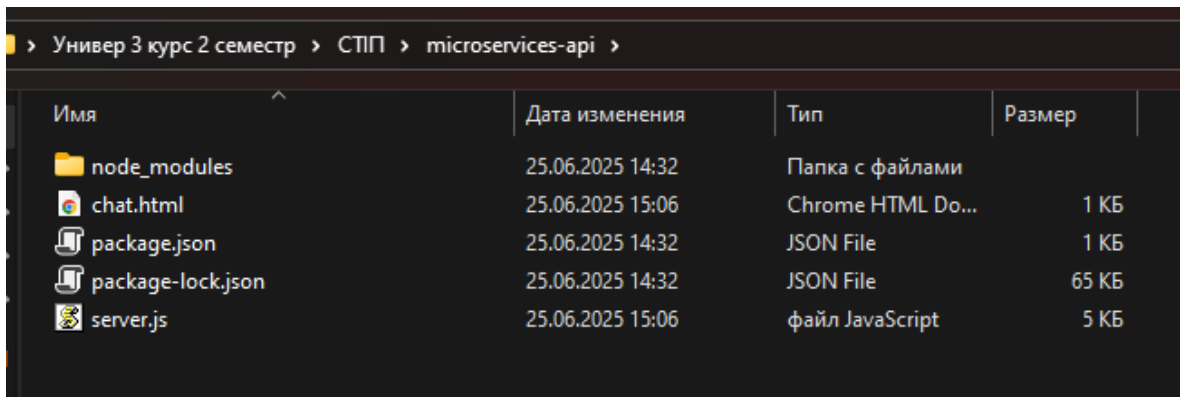
Лабораторна робота № 5

з теми: Мікросервіси та API-first підхід

Варіант 10

10. Створити API для системи підтримки: заявки користувачів, статуси, JWT-аутентифікація, WebSockets для онлайн-чату з підтримкою.

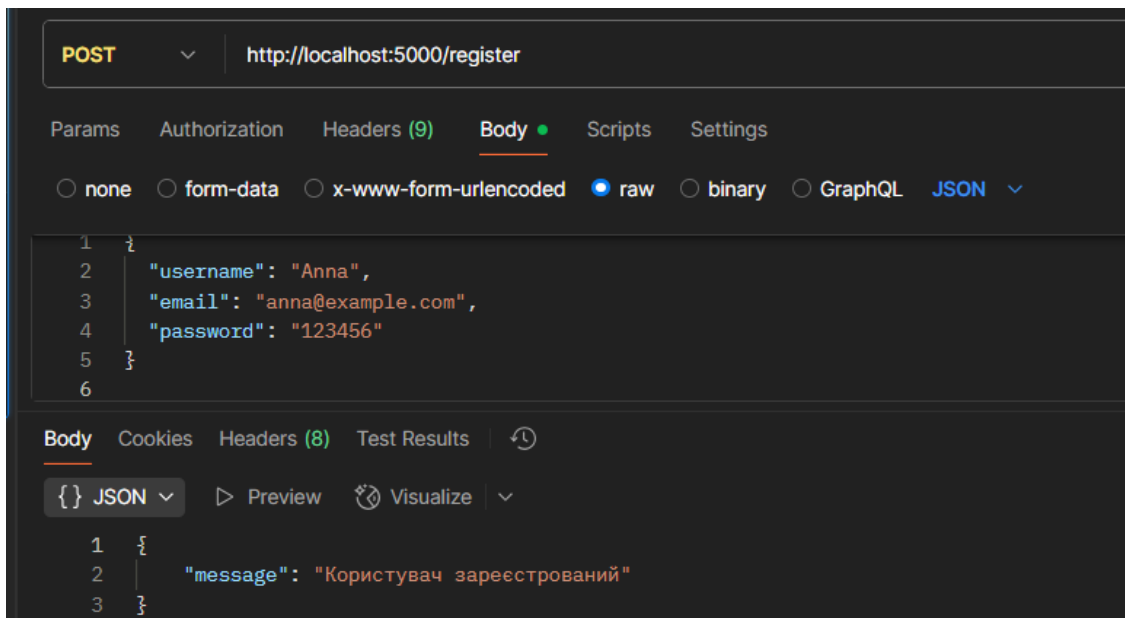
Структура:



Имя	Дата изменения	Тип	Размер
node_modules	25.06.2025 14:32	Папка с файлами	
chat.html	25.06.2025 15:06	Chrome HTML Do...	1 КБ
package.json	25.06.2025 14:32	JSON File	1 КБ
package-lock.json	25.06.2025 14:32	JSON File	65 КБ
server.js	25.06.2025 15:06	файл JavaScript	5 КБ

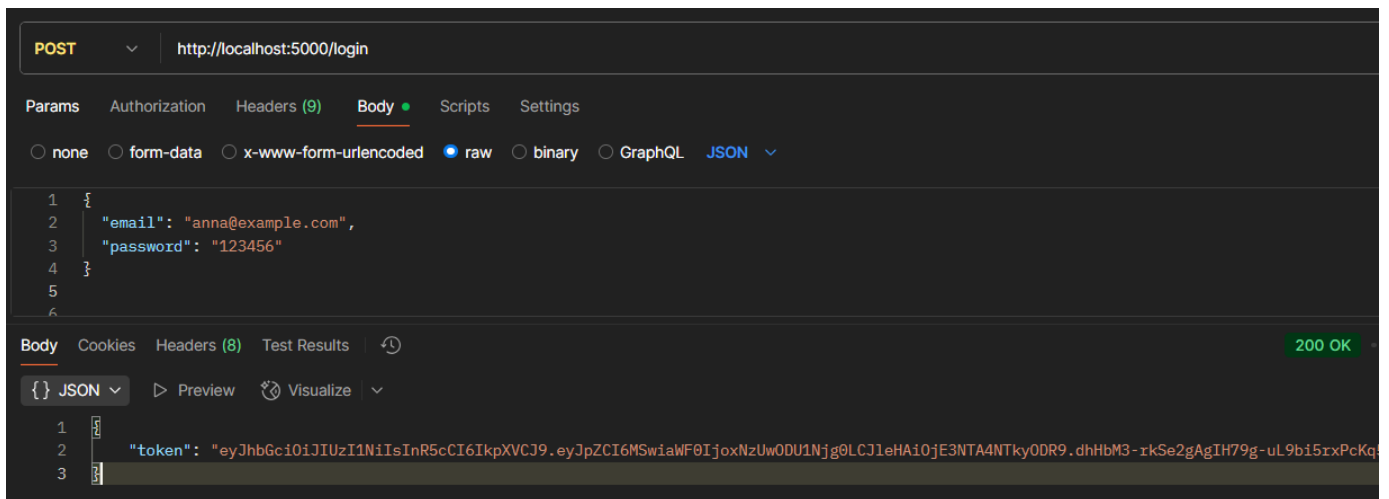
Реалізація реєстрації:

```
app.post("/register", async (req, res) => {  
  const { username, email, password } = req.body;  
  const hashed = await bcrypt.hash(password, 10);  
  const user = { id: currentUserId++, username, email, password: hashed };  
  users.push(user);  
  res.json({ message: "Користувач зареєстрований" });  
});
```



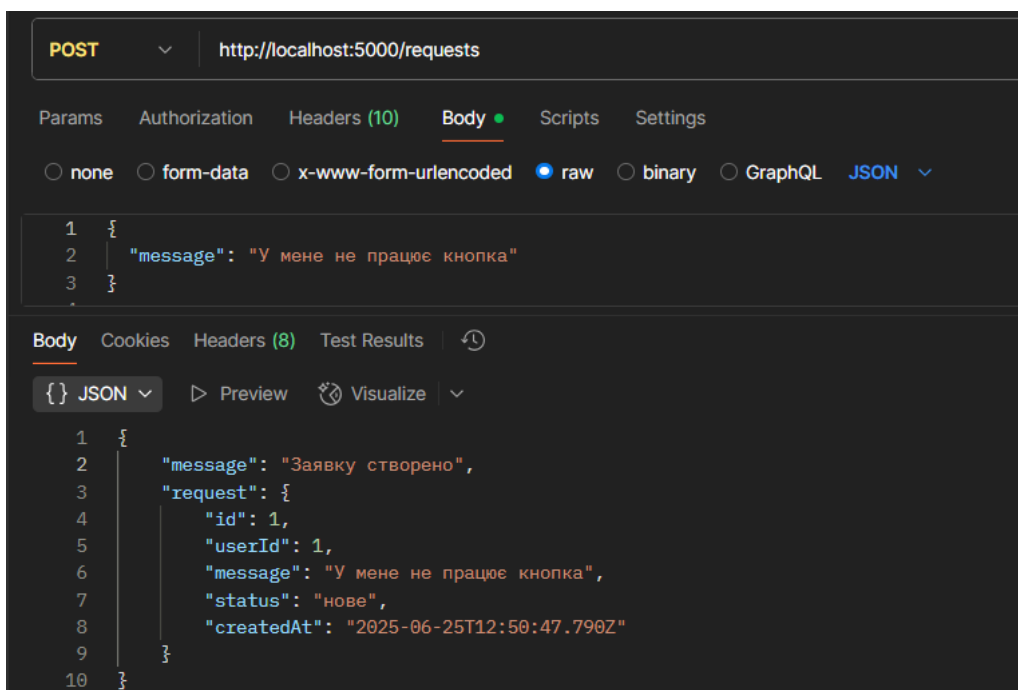
Реалізація авторизації:

```
app.post("/login", async (req, res) => {
  const { email, password } = req.body;
  const user = users.find(u => u.email === email);
  if (!user || !(await bcrypt.compare(password, user.password))) {
    return res.status(401).json({ message: "Невірний email або пароль" });
  }
  const token = jwt.sign({ id: user.id }, JWT_SECRET, { expiresIn: "1h" });
  res.json({ token });
});
```



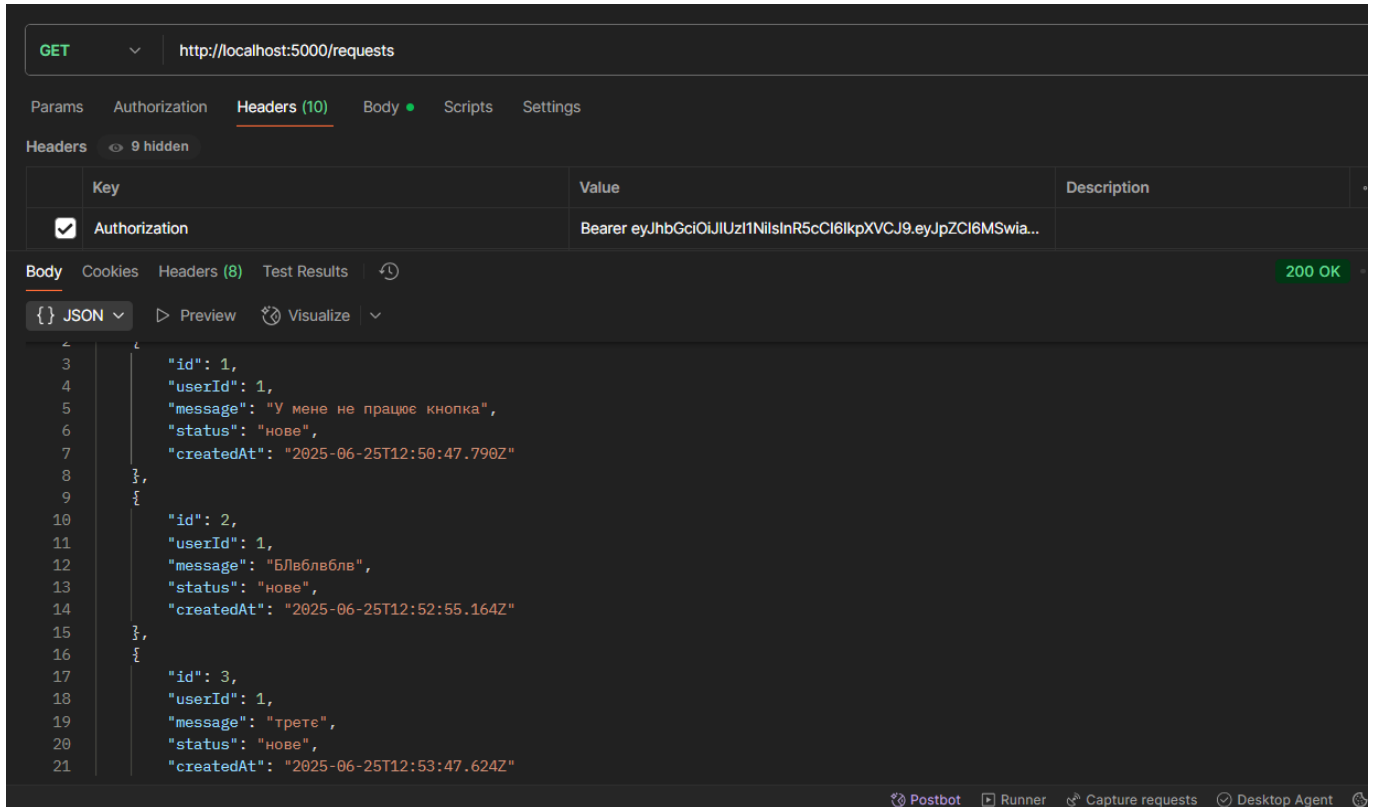
Реалізація створення заявки:

```
app.post("/requests", authMiddleware, (req, res) => {
  const request = {
    id: currentRequestId++,
    userId: req.userId,
    message: req.body.message,
    status: "нове",
    createdAt: new Date().toISOString(),
  };
  requests.push(request);
  res.json({ message: "Заявку створено", request });
});
```



Реалізація отримання заявок користувачів:

```
app.get("/requests", authMiddleware, (req, res) => {  
  const userRequests = requests.filter(r => r.userId === req.userId);  
  res.json(userRequests);  
});
```



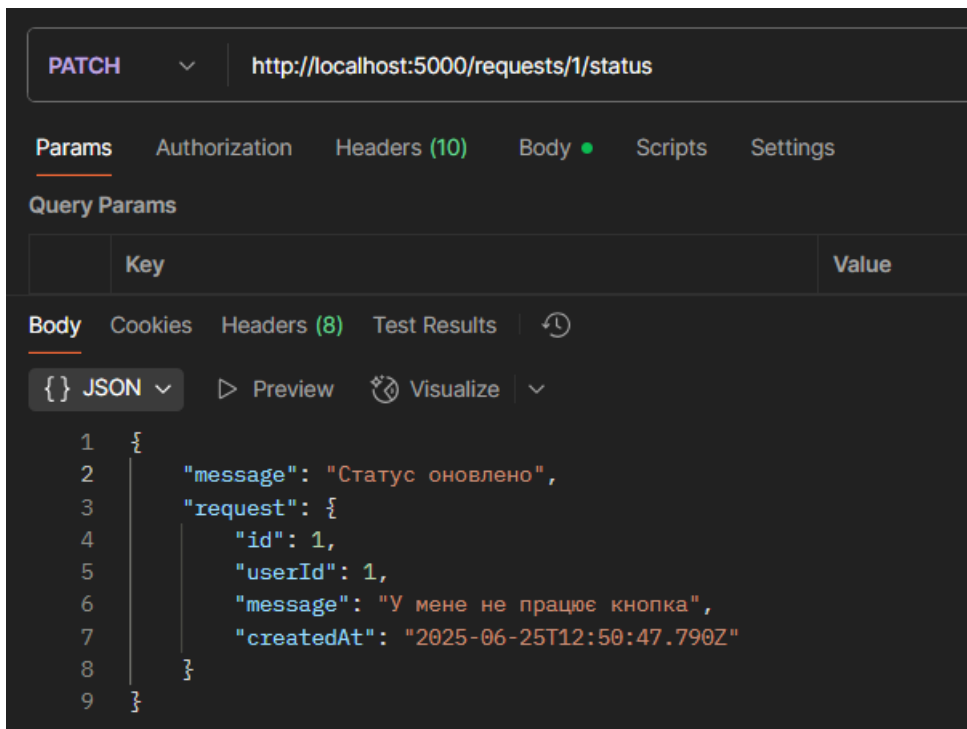
The screenshot shows a REST client interface with the following details:

- Method:** GET
- URL:** http://localhost:5000/requests
- Headers (10):** Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6MSwia...
- Body:** JSON
- Status:** 200 OK
- Response Body:**

```
[  
  {  
    "id": 1,  
    "userId": 1,  
    "message": "У мене не працює кнопка",  
    "status": "нове",  
    "createdAt": "2025-06-25T12:50:47.790Z"  
  },  
  {  
    "id": 2,  
    "userId": 1,  
    "message": "Блвблвблв",  
    "status": "нове",  
    "createdAt": "2025-06-25T12:52:55.164Z"  
  },  
  {  
    "id": 3,  
    "userId": 1,  
    "message": "трете",  
    "status": "нове",  
    "createdAt": "2025-06-25T12:53:47.624Z"  
  }  
]
```

Реалізація оновлення заявки (тільки для адміна):

```
app.patch("/requests/:id/status", (req, res) => {  
  const request = requests.find(r => r.id == req.params.id);  
  if (!request) return res.status(404).json({ message: "Заявка не знайдена" });  
  request.status = req.body.status;  
  res.json({ message: "Статус оновлено", request });  
});
```



WebSocket чат:

```
<!DOCTYPE html>
<html lang="uk">
<head>
  <meta charset="UTF-8" />
  <title>WebSocket Чат</title>
</head>
<body>
  <h2>WebSocket Чат</h2>
  <input id="msg" placeholder="Введи повідомлення..." />
  <button onclick="send()">Надіслати</button>
  <pre id="log"></pre>

  <script>
    const socket = new WebSocket("ws://localhost:5051");

    socket.onopen = () => log("З'єднання встановлено");
    socket.onmessage = (e) => log("Отримано: " + e.data);
    socket.onerror = (e) => log("Помилка: " + (e.message || 'невідома помилка'));

    function send() {
      const text = document.getElementById("msg").value;
      const data = { from: "Anna", text };
      socket.send(JSON.stringify(data));
    }

    function log(msg) {
      document.getElementById("log").textContent += msg + "\n";
    }
  </script>
</body>
</html>
```

WebSocket Чат

✖ Помилка: невідома помилка