Copilot

Sure, let's complete the `ServiceRequest` component using route parameters to pass multiple values. Here's the continuation and completion of the previous example:

ServiceRequest Component (Continued)

```
import React, { useState, useEffect } from 'react';
import { useParams, useNavigate } from 'react-router-dom';
import api from "../services/api";
import { Formik } from 'formik';
import * as Yup from 'yup';
import { Form, Button, Container, Row, Col, Card, Alert } from 'react-bootstrap';

const ServiceRequest = () => {
  const { categoryId, param1, param2 } = useParams();
  const [category, setCategory] = useState(null);
  const [error, setError] = useState('');
  const navigate = useNavigate();

  useEffect(() => {
    const fetchCategory = async () => {
      try {
        const response = await api.get(`http://localhost:8989/api/services/${categoryId}`);
        setCategory(response.data);
      } catch (error) {
        setError('Error while fetching the category data');
      }
    };

    fetchCategory();
  }, [categoryId]);

  const initialValues = {
    serviceDescription: '',
    price: '',
    userId: category ? category.user.id : '',
    param1: param1 || '',
    param2: param2 || '',
  };

  const validationSchema = Yup.object({
    serviceDescription: Yup.string().required('Service description is required'),
    price: Yup.number().required('Price is required').positive('Price must be positive'),
  });

  const handleSubmit = async (values, { setSubmitting }) => {
    try {
      await api.post('http://localhost:8989/api/service-requests', values);
      navigate('/'); // Redirect to home or appropriate page after submission
    } catch (error) {
      setError('Error while submitting the service request');
    } finally {
      setSubmitting(false);
    }
  };

  return (
    <Container>
      <Row className="justify-content-md-center">
        <Col md={6}>
          <Card className="service-request-card">
            <Card.Body>
              <h2 className="service-request-header text-center">Request Service</h2>
              {error && <Alert variant="danger">{error}</Alert>}
              {category && (
                <Formik
                  initialValues={initialValues}
                  validationSchema={validationSchema}
```

```
      onSubmit={handleSubmit}
    >
      {({
        values,
        errors,
        touched,
        handleChange,
        handleBlur,
        handleSubmit,
        isSubmitting,
      }) => (
        <Form onSubmit={handleSubmit}>
          <Form.Group controlId="formServiceDescription">
            <Form.Label className="text-success">Service Description</Form.Label>
            <Form.Control
              type="text"
              name="serviceDescription"
              placeholder="Enter service description"
              value={values.serviceDescription}
              onChange={handleChange}
              onBlur={handleBlur}
              isInvalid={touched.serviceDescription && !!errors.serviceDescription}
            />
            <Form.Control.Feedback type="invalid">
              {errors.serviceDescription}
            </Form.Control.Feedback>
          </Form.Group>

          <Form.Group controlId="formPrice">
            <Form.Label className="text-success">Price</Form.Label>
            <Form.Control
              type="number"
              name="price"
              placeholder="Enter price"
              value={values.price}
              onChange={handleChange}
              onBlur={handleBlur}
              isInvalid={touched.price && !!errors.price}
            />
            <Form.Control.Feedback type="invalid">
              {errors.price}
            </Form.Control.Feedback>
          </Form.Group>

          <Form.Group controlId="formUserId">
            <Form.Label className="text-success">User ID</Form.Label>
            <Form.Control
              type="text"
              name="userId"
              value={values.userId}
              readOnly
            />
          </Form.Group>

          <Form.Group controlId="formParam1">
            <Form.Label className="text-success">Param 1</Form.Label>
            <Form.Control
              type="text"
              name="param1"
              value={values.param1}
              readOnly
            />
          </Form.Group>

          <Form.Group controlId="formParam2">
            <Form.Label className="text-success">Param 2</Form.Label>
            <Form.Control
              type="text"
              name="param2"
              value={values.param2}
              readOnly
            />
          </Form.Group>

          <Button variant="success" type="submit" disabled={isSubmitting}>
```

```
                    Submit Request
                  </Button>
                </Form>
              )}
            </Formik>
          )}
        </Card.Body>
      </Card>
    </Col>
  </Row>
</Container>
);
};

export default ServiceRequest;
```

Update Routes

Make sure to update your router configuration to include the new route for the `ServiceRequest` component. Here's an example using React Router:

```
import { BrowserRouter as Router, Route, Routes } from 'react-router-dom';
import CustomNavbar from './components/CustomNavbar';
import ViewCategories from './components/ViewCategories';
import ServiceRequest from './components/ServiceRequest';
import Login from './components/Login';
// Import other components as needed

function App() {
  const [userRole, setUserRole] = useState('');

  return (
    <Router>
      <CustomNavbar userRole={userRole} setUserRole={setUserRole} />
      <Routes>
        <Route path="/" element={<ViewCategories />} />
        <Route path="/login" element={<Login setUserRole={setUserRole} />} />
        <Route path="/request-service/:categoryId/:param1/:param2" element={<ServiceRequest />} />
        {/* Add other routes as needed */}
      </Routes>
    </Router>
  );
}

export default App;
```

Modify the handleRequestService Function

Ensure the `handleRequestService` function in your `ViewCategories` component navigates to the `ServiceRequest` component with the `categoryId` and additional parameters:

```
const handleRequestService = (categoryId, additionalParam1, additionalParam2) => {
  navigate(`/request-service/${categoryId}/${additionalParam1}/${additionalParam2}`);
};
```

Example Usage in ViewCategories Component

Here's how you might call `handleRequestService` with multiple parameters:

```
<Button variant="success" onClick={() => handleRequestService(category.id, 'paramValue1', 'paramValue2')}>
  Request Service
</Button>
```

This setup will ensure that when a user clicks the "Request Service" button, they are navigated to the `ServiceRequest` component with the `categoryId` and additional parameters passed as route parameters. The form will be pre-populated with the `userId` from the selected category and the additional parameters.

Feel free to ask if you have any questions or need further assistance! 😊