Formik + Zod + Material UI Contact Form with Debug Panel

Let's make a TypeScript project and add:

- **Type-safe validation** with Zod
- **Real-time debugging** with a dev-only panel

Required Packages

Install the necessary libraries:

```
npm install formik zod @mui/material @emotion/react
@emotion/styled
```

- `formik`: for managing form state, validation, submission, and user interactions.
- `zod`: for schema-based validation with TypeScript type inference.
- `@mui/material` and `@emotion/*`: for UI components and styling using Material UI.

`validationSchema.ts`

- **Zod schema** helps define all rules declaratively in one place.
- **TypeScript inference** makes form data strongly typed.

```
//Defines the validation rules using Zod and provides
TypeScript types.

import { z } from 'zod';

export const contactSchema = z.object({
  name: z.string().min(2, 'Name must be at least 2
characters').max(50),
  email: z.string().email('Invalid email format'),
  message: z.string().min(10, 'Message must be at least 10
characters').max(500),
});
```

```
export type ContactFormValues = z.infer<typeof
contactSchema>;
```

App.tsx

```
//Wraps the contact form in a responsive container.

import React from 'react';
import Container from '@mui/material/Container';
import ContactForm from './components/ContactForm';

const App = () => (
  <Container maxWidth="sm">
    <h1>Contact Us</h1>
    <ContactForm />
  </Container>
);

export default App;
```

components/TextInput.tsx

```
//reusable field that works with Formik
import React from 'react';
import TextField from '@mui/material/TextField';
import { FieldProps } from 'formik';

const TextInput: React.FC<FieldProps & { label: string;
multiline?: boolean; rows?: number }> = ({
  field,
  form,
  label,
  multiline,
  rows,
  ...props
}) => {
```

```
  const error = form.touched[field.name] &&
form.errors[field.name];

  return (
    <TextField
      {...field}
      {...props}
      fullWidth
      variant="outlined"
      label={label}
      multiline={multiline}
      rows={rows}
      error={!!error}
      helperText={error as string}
    />
  );
};

export default TextInput;
```

components/SubmitButton.tsx

```
//reusable submit button

import React from 'react';
import Button from '@mui/material/Button';

type Props = {
  children: React.ReactNode;
};

const SubmitButton: React.FC<Props> = ({ children }) => (
  <Button type="submit" variant="contained" color="primary"
fullWidth>
    {children}
  </Button>
);

export default SubmitButton;
```

components/FormikDebugPanel.tsx

- Helps developers debug `values`, `errors`, and `touched` fields during development.
- Does not render in production, keeping user experience clean.

```
//Panel to visualize live Formik state.

import React from 'react';
import { useFormikContext } from 'formik';
import { Box } from '@mui/material';

const FormikDebugPanel: React.FC = () => {
  const formik = useFormikContext<any>();

  if (process.env.NODE_ENV !== 'development') return null;

  return (
    <Box
      mt={4}
      p={2}
      sx={{
        backgroundColor: '#f5f5f5',
        fontFamily: 'monospace',
        fontSize: 12,
        whiteSpace: 'pre-wrap',
        overflowX: 'auto',
        border: '1px solid #ccc',
        borderRadius: '4px',
      }}
    >
      <strong>Formik Debug Panel</strong>
      <pre>{JSON.stringify(formik, null, 2)}</pre>
    </Box>
  );
};

export default FormikDebugPanel;
```

components/ContactForm.tsx

- Central form logic: defines initial values, validation, and submission behavior.
- Uses `safeParse()` to validate against Zod schema.
- Manually maps Zod validation errors to Formik's expected format.
- Debug panel gives live feedback for developers.

```
//Main form logic using Formik + Zod + UI components
together. Implements FormikDevPanel

import React from 'react';
import { Formik, Form, Field } from 'formik';
import { contactSchema, ContactFormValues } from '../
validationSchema';
import TextInput from './TextInput';
import SubmitButton from './SubmitButton';
import FormikDebugPanel from './FormikDebugPanel';
import { Box } from '@mui/material';

const ContactForm: React.FC = () => {
  const initialValues: ContactFormValues = {
    name: '',
    email: '',
    message: '',
  };

  const validate = (values: ContactFormValues) => {
    const result = contactSchema.safeParse(values);
    if (result.success) return {};
    const formErrors: Record<string, string> = {};
    result.error.errors.forEach((err) => {
      if (err.path[0]) {
        formErrors[err.path[0]] = err.message;
      }
    });
    return formErrors;
  };
```

```
  const handleSubmit = (values: ContactFormValues, actions:
any) => {
    alert(JSON.stringify(values, null, 2));
    actions.setSubmitting(false);
    actions.resetForm();
  };

  return (
    <Formik initialValues={initialValues}
validate={validate} onSubmit={handleSubmit}>
      {() => (
        <>
          <Form>
            <Box mb={2}>
              <Field name="name" label="Name"
component={TextInput} />
            </Box>
            <Box mb={2}>
              <Field name="email" label="Email"
component={TextInput} />
            </Box>
            <Box mb={2}>
              <Field name="message" label="Message"
component={TextInput} multiline rows={4} />
            </Box>
            <SubmitButton>Submit</SubmitButton>
          </Form>
          {process.env.NODE_ENV === 'development' &&
<FormikDebugPanel />}
        </>
      )}
    </Formik>
  );
};

export default ContactForm;
```