ДОДАТОК А

Код програми

```
//App.tsx
import { Refine, AuthProvider } from '@pankod/refine-core'
import {
  notificationProvider,
  RefineSnackbarProvider,
  CssBaseline,
  GlobalStyles,
  ReadyPage,
  ErrorComponent,
} from '@pankod/refine-mui'
import routerProvider from '@pankod/refine-react-router-v6'
import dataProvider from '@pankod/refine-simple-rest'
import axios, { AxiosRequestConfig } from 'axios'
import { Title, Sider, Layout, Header } from './components/layout'
import { ColorModeContextProvider } from './contexts'
import { CredentialResponse } from './interfaces/google'
import { CreateLeanProjectForm } from './components/lean/LeanProjectForm'
import { MyLeanProjects } from "./components/lean/MyLeanProjects";
import LeanProjectDetails from "./components/lean/LeanProjectDetails";
import {
  Login,
  Home,
  Agents,
  MyProfile,
  AgentProfile,
} from './pages'
import { parseJwt } from './utils/parse-jwt'
import React from 'react'
```

```
import {
  AccountCircleOutlined,
  ChatBubbleOutline,
  PeopleAltOutlined,
  StarOutlineRounded,
  VillaOutlined,
} from '@mui/icons-material'
const axiosInstance = axios.create()
axiosInstance.interceptors.request.use((request: AxiosRequestConfig) => {
  const token = localStorage.getItem('token')
  if (request.headers) {
    request.headers['Authorization'] = `Bearer ${token}`
  } else {
    request.headers = {
      Authorization: `Bearer ${token}`,
    }
  }
  return request
})
function App() {
  const authProvider: AuthProvider = {
    login: async ({ credential }: CredentialResponse) => {
      const profileObj = credential ? parseJwt(credential) : null
      if (profileObj) {
const response = await fetch(`${process.env.REACT_APP_BACKEND_URL}/users`,
{
          method: 'POST',
          headers: { 'Content-Type': 'application/json' },
          body: JSON.stringify({
            name: profileObj.name,
```

```
        email: profileObj.email,
        avatar: profileObj.picture,
      }),
    })
    const data = await response.json()
    if (response.status === 200) {
      localStorage.setItem(
        'user',
        JSON.stringify({
          ...profileObj,
          avatar: profileObj.picture,
          userid: data._id,
        }),
      )
    } else {
      return Promise.reject
    }
  }
  localStorage.setItem('token', `${credential}`)
  return Promise.resolve()
},
logout: () => {
  const token = localStorage.getItem('token')
  if (token && typeof window !== 'undefined') {
    localStorage.removeItem('token')
    localStorage.removeItem('user')
    axios.defaults.headers.common = {}
    window.google?.accounts.id.revoke(token, () => {
      return Promise.resolve()
    })
```

```
      }
      return Promise.resolve()
    },
    checkError: () => Promise.resolve(),
    checkAuth: async () => {
      const token = localStorage.getItem('token')
      if (token) {
        return Promise.resolve()
      }
      return Promise.reject()
    },
    getPermissions: () => Promise.resolve(),
    getUserIdentity: async () => {
      const user = localStorage.getItem('user')
      if (user) {
        return Promise.resolve(JSON.parse(user))
      }
    },
  }
  return (
    <ColorModeContextProvider>
      <CssBaseline />
      <GlobalStyles styles={{ html: { WebkitFontSmoothing: 'auto' } }} />

      <RefineSnackbarProvider>
        <Refine
          dataProvider={dataProvider(`${process.env.REACT_APP_BACKEND_URL}` ||
"")}
          notificationProvider={notificationProvider}
          ReadyPage={ReadyPage}
```

```
        catchAll={<ErrorComponent />}
        resources={[
          { name: "lean-projects",
            list: MyLeanProjects,
            create: CreateLeanProjectForm,
            show: LeanProjectDetails,
          }, {  name: 'agents',
            list: Agents,
            show: AgentProfile,
            icon: <PeopleAltOutlined />,
          },  {  name: 'my-profile', options: { label: 'My Profile' },
            list: MyProfile, icon: <AccountCircleOutlined />,  },   ]}
        Title={Title}
        Sider={Sider}
        Layout={Layout}
        Header={Header}
        routerProvider={routerProvider}
        authProvider={authProvider}
        LoginPage={Login}
        DashboardPage={Home}
      />
    </RefineSnackbarProvider>
  </ColorModeContextProvider>
 )
}
export default App
```

Код файлу login.tsx

```
import { useLogin } from '@pankod/refine-core'
```

```
import { Container, Box } from '@pankod/refine-mui'
import { useEffect, useRef } from 'react'
import yariga from '../assets/yariga.svg'
import { CredentialResponse } from '../interfaces/google'
export const Login: React.FC = () => {
  const { mutate: login } = useLogin<CredentialResponse>()
  const GoogleButton = (): JSX.Element => {
    const divRef = useRef<HTMLDivElement>(null)
    useEffect(() => {
      if (typeof window === 'undefined' || !window.google || !divRef.current) {
        return
      }
      try {   window.google.accounts.id.initialize({
          ux_mode: 'popup',
          client_id: process.env.REACT_APP_GOOGLE_CLIENT_ID,
          callback: async (res: CredentialResponse) => {
            if (res.credential) { login(res)  }  },  })
        window.google.accounts.id.renderButton(divRef.current, {
          theme: 'filled_blue',
          size: 'medium',
          type: 'standard',
        })
      } catch (error) {
        console.log(error)
      }
    }, []) // you can also add your client id as dependency here

    return <div ref={divRef} />
  }
return (
```

```
<Box
  component="div"
  sx={{
    backgroundColor: '#FCFCFC',
    backgroundSize: 'cover',
  }}  >
  <Container
    component="main"
    maxWidth="xs"
    sx={{
      display: 'flex',
      flexDirection: 'column',
      justifyContent: 'center',
      height: '100vh',
    }}
  >
    <Box
      sx={{
        display: 'flex',
        justifyContent: 'center',
        flexDirection: 'column',
        alignItems: 'center',
      }}
    >
      <div>
        <img src={yariga} alt="Yariga Logo" />
      </div>
      <Box mt={4}>
        <GoogleButton />
      </Box>
```

```
      </Box>
    </Container>
  </Box>
 )
}
```

Код файлу FiveS.tsx

```
import React from "react";
import PropTypes from "prop-types";
import { Box, Typography } from "@mui/material";
import { FiveSTable } from "./FiveSTable";
export const FiveS = ({ fiveSData, onChange }) => {
 const updateData = (section, newData) => {
  onChange(section, newData);
 };
 return (
  <Box>
   <Typography variant="h5" gutterBottom>Методологія 5S (таблична форма)</Typography>
   <FiveSTable
    title="1. Sort (Сортування)"
    data={fiveSData?.sort || []}
    onChange={(newData) => updateData("sort", newData)}
    columns={["Об'єкт", "Статус", "Причина"]}
    fields={["object", "status", "reason"]}  />

   <FiveSTable
    title="2. Set in Order (Систематизація)"
    data={fiveSData?.setInOrder || []}
```

```
      onChange={(newData) => updateData("setInOrder", newData)}
      columns={["Об'єкт", "Місце зберігання", "Маркування"]}
      fields={["object", "location", "labeling"]}  />
    <FiveSTable  title="3. Shine (Прибирання)"
      data={fiveSData?.shine || []}
      onChange={(newData) => updateData("shine", newData)}
      columns={["Зона", "Відповідальний", "Частота"]}
      fields={["zone", "responsible", "frequency"]}  />
    <FiveSTable  title="4. Standardize (Стандартизація)"
      data={fiveSData?.standardize || []}
      onChange={(newData) => updateData("standardize", newData)}
      columns={["Процедура", "Інструкція", "Візуальний контроль"]}
      fields={["procedure", "instruction", "visual"]} />
    <FiveSTable  title="5. Sustain (Дотримання)"
      data={fiveSData?.sustain || []}
      onChange={(newData) => updateData("sustain", newData)}
      columns={["Дата перевірки", "Відповідальний", "Коментар"]}
      fields={["date", "responsible", "comment"]}  /> </Box> );};
FiveS.propTypes = {
  fiveSData: PropTypes.object.isRequired,
  onChange: PropTypes.func.isRequired,
};
```

Код файлу index.js

```
import express from 'express'
import * as dotenv from 'dotenv'
import cors from 'cors'
import connectDB from './mongodb/connect.js'
```

```
import userRouter from './routes/user.routes.js'
import propertyRouter from './routes/property.routes.js'
import bodyParser from 'body-parser'
import leanRouter from './routes/lean.routes.js'
import notificationRouter from './routes/notification.routes.js'
import kaizenRouter from './routes/kaizen.routes.js'
import aiRouter from './routes/ai.routes.js'
dotenv.config()
const app = express()
app.use(cors())
app.use(bodyParser.urlencoded({ extended: true }))
app.use(bodyParser.json({ limit: '10mb' }))
app.get('/', (req, res) => { res.send({ message: 'Hello World! ' })})
app.use('/api/kaizen', kaizenRouter)
app.use('/api/v1/notifications', notificationRouter)
app.use('/api/v1/ai', aiRouter)
app.use('/api/v1/lean', leanRouter)
app.use('/api/v1/users', userRouter)
app.use('/api/v1/properties', propertyRouter)
const startServer = async () => {
  try { await connectDB(process.env.MONGODB_URL)
    const PORT = process.env.PORT || 9000
    app.listen(PORT, () => {console.log(`Server started on port
http://localhost:${PORT}`)
    })
  } catch (error) { console.error('Server failed:', error) }} startServer()
```

ДОДАТОК Б

Демонстраційний матеріал

Формат	Позиція	Найменування	Додаткові відомості
		Текстові документи	
А4	ГЮІК. 442351.009 ПЗ	Пояснювальна записка	63 с.
		Додаткові матеріали	
		Код програми	10 с.
		Демонстраційний матеріал	10 с.

Змін.	Арк.	Номер докум.	Підпис	Дата	ГЮІК.442351.009 ВД			
Розроб.		Зеленько В. І.			Розроблення консалтингової системи для виробництва типу Lean production	Літера	Аркуш	Аркушів
Перевір.		Сезонова І. К.				Н	1	1
Н. контр.		Демська Н. П.				Кафедра КІТАР ХНУРЕ		
Затв.		Невлюдов І.Ш.						