导出时间: 2025-10-18 15:35:17

**文件: frontend/src/components/UserList.vue**

<template>  
 <div class="user-list">  
 <h2>用户列表</h2>  
 <el-table :data="users" style="width: 100%" v-loading="loading">  
 <el-table-column prop="id" label="ID" width="80"></el-table-column>  
 <el-table-column prop="name" label="姓名"></el-table-column>  
 <el-table-column prop="email" label="邮箱"></el-table-column>  
 <el-table-column prop="createdAt" label="创建时间"></el-table-column>  
 <el-table-column label="操作" width="150">  
 <template slot-scope="scope">  
 <el-button size="mini" @click="editUser(scope.row)">编辑</el-button>  
 <el-button size="mini" type="danger" @click="deleteUser(scope.row)">删除</el-button>  
 </template>  
 </el-table-column>  
 </el-table>  
  
 <el-dialog title="编辑用户" :visible.sync="dialogVisible">  
 <el-form :model="currentUser">  
 <el-form-item label="姓名">  
 <el-input v-model="currentUser.name"></el-input>  
 </el-form-item>  
 <el-form-item label="邮箱">  
 <el-input v-model="currentUser.email"></el-input>  
 </el-form-item>  
 </el-form>  
 <div slot="footer" class="dialog-footer">  
 <el-button @click="dialogVisible = false">取消</el-button>  
 <el-button type="primary" @click="saveUser">保存</el-button>  
 </div>  
 </el-dialog>  
 </div>  
</template>  
  
<script>  
import { getUsers, updateUser, deleteUser } from '@/api/user'  
  
export default {  
 name: 'UserList',  
 data() {  
 return {  
 users: [],  
 loading: false,  
 dialogVisible: false,  
 currentUser: {  
 id: null,  
 name: '',  
 email: ''  
 }  
 }  
 },  
 created() {  
 this.fetchUsers()  
 },  
 methods: {  
 async fetchUsers() {  
 this.loading = true  
 try {  
 const response = await getUsers()  
 this.users = response.data  
 } catch (error) {  
 this.$message.error('获取用户列表失败')  
 console.error(error)  
 } finally {  
 this.loading = false  
 }  
 },  
 editUser(user) {  
 this.currentUser = { ...user }  
 this.dialogVisible = true  
 },  
 async saveUser() {  
 try {  
 await updateUser(this.currentUser.id, this.currentUser)  
 this.$message.success('用户更新成功')  
 this.dialogVisible = false  
 this.fetchUsers()  
 } catch (error) {  
 this.$message.error('用户更新失败')  
 console.error(error)  
 }  
 },  
 async deleteUser(user) {  
 try {  
 await this.$confirm('确定要删除该用户吗？', '提示', {  
 confirmButtonText: '确定',  
 cancelButtonText: '取消',  
 type: 'warning'  
 })  
 await deleteUser(user.id)  
 this.$message.success('用户删除成功')  
 this.fetchUsers()  
 } catch (error) {  
 if (error !== 'cancel') {  
 this.$message.error('用户删除失败')  
 console.error(error)  
 }  
 }  
 }  
 }  
}  
</script>  
  
<style scoped>  
.user-list {  
 padding: 20px;  
}  
</style>

--------------------------------------------------------------------------------

**文件: frontend/src/views/Home.vue**

<template>  
 <div class="home">  
 <nav class="navbar">  
 <div class="nav-brand">  
 <h1>用户管理系统</h1>  
 </div>  
 <div class="nav-menu">  
 <router-link to="/" class="nav-link">首页</router-link>  
 <router-link to="/users" class="nav-link">用户管理</router-link>  
 <router-link to="/about" class="nav-link">关于</router-link>  
 </div>  
 </nav>  
  
 <main class="main-content">  
 <router-view></router-view>  
 </main>  
  
 <footer class="footer">  
 <p>&copy; 2024 用户管理系统. All rights reserved.</p>  
 </footer>  
 </div>  
</template>  
  
<script>  
export default {  
 name: 'Home'  
}  
</script>  
  
<style scoped>  
.home {  
 min-height: 100vh;  
 display: flex;  
 flex-direction: column;  
}  
  
.navbar {  
 background-color: #2c3e50;  
 color: white;  
 padding: 1rem 2rem;  
 display: flex;  
 justify-content: space-between;  
 align-items: center;  
}  
  
.nav-brand h1 {  
 margin: 0;  
 font-size: 1.5rem;  
}  
  
.nav-menu {  
 display: flex;  
 gap: 2rem;  
}  
  
.nav-link {  
 color: white;  
 text-decoration: none;  
 padding: 0.5rem 1rem;  
 border-radius: 4px;  
 transition: background-color 0.3s;  
}  
  
.nav-link:hover {  
 background-color: #34495e;  
}  
  
.nav-link.router-link-active {  
 background-color: #3498db;  
}  
  
.main-content {  
 flex: 1;  
 padding: 2rem;  
}  
  
.footer {  
 background-color: #34495e;  
 color: white;  
 text-align: center;  
 padding: 1rem;  
}  
</style>

--------------------------------------------------------------------------------

**文件: src/main/java/com/example/UserController.java**

package com.example;  
  
import org.springframework.web.bind.annotation.\*;  
import org.springframework.beans.factory.annotation.Autowired;  
import java.util.List;  
import java.util.Optional;  
  
@RestController  
@RequestMapping("/api/users")  
public class UserController {  
  
 @Autowired  
 private UserService userService;  
  
 @GetMapping  
 public List<User> getAllUsers() {  
 return userService.findAll();  
 }  
  
 @GetMapping("/{id}")  
 public ResponseEntity<User> getUserById(@PathVariable Long id) {  
 Optional<User> user = userService.findById(id);  
 return user.map(ResponseEntity::ok)  
 .orElse(ResponseEntity.notFound().build());  
 }  
  
 @PostMapping  
 public User createUser(@RequestBody User user) {  
 return userService.save(user);  
 }  
  
 @PutMapping("/{id}")  
 public ResponseEntity<User> updateUser(@PathVariable Long id, @RequestBody User userDetails) {  
 Optional<User> user = userService.findById(id);  
 if (user.isPresent()) {  
 User updatedUser = user.get();  
 updatedUser.setName(userDetails.getName());  
 updatedUser.setEmail(userDetails.getEmail());  
 return ResponseEntity.ok(userService.save(updatedUser));  
 }  
 return ResponseEntity.notFound().build();  
 }  
  
 @DeleteMapping("/{id}")  
 public ResponseEntity<Void> deleteUser(@PathVariable Long id) {  
 userService.deleteById(id);  
 return ResponseEntity.noContent().build();  
 }  
}

--------------------------------------------------------------------------------

**文件: src/main/resources/application.yml**

server:  
 port: 8080  
 servlet:  
 context-path: /api  
  
spring:  
 datasource:  
 url: jdbc:mysql://localhost:3306/user\_db?useSSL=false&serverTimezone=UTC  
 username: root  
 password: password  
 driver-class-name: com.mysql.cj.jdbc.Driver  
  
 jpa:  
 hibernate:  
 ddl-auto: update  
 show-sql: true  
 properties:  
 hibernate:  
 dialect: org.hibernate.dialect.MySQL8Dialect  
 format\_sql: true  
  
 security:  
 user:  
 name: admin  
 password: admin123  
 roles: ADMIN  
  
logging:  
 level:  
 com.example: DEBUG  
 org.springframework.security: DEBUG  
 pattern:  
 console: "%d{yyyy-MM-dd HH:mm:ss} - %msg%n"  
 file: "%d{yyyy-MM-dd HH:mm:ss} [%thread] %-5level %logger{36} - %msg%n"  
 file:  
 name: logs/application.log

--------------------------------------------------------------------------------