**src\main\java\edu\ynu\arduino\ArduinoApplication.java**

package edu.ynu.arduino;  
  
import lombok.extern.slf4j.Slf4j;  
import org.springframework.boot.SpringApplication;  
import org.springframework.boot.autoconfigure.SpringBootApplication;  
  
@Slf4j  
@SpringBootApplication  
public class ArduinoApplication {  
  
 public static void main(String[] args) {  
 SpringApplication.run(ArduinoApplication.class, args);  
 log.info("logger class: {}", log.getClass());  
 }  
  
  
// @GetMapping("/hello")  
// public String hello(@RequestParam(value = "name", defaultValue = "world") String name) {  
// return String.format("hello %s", name);  
// }  
  
}

**src\main\java\edu\ynu\arduino\configuration\DruidConfig.java**

package edu.ynu.arduino.configuration;  
  
import com.alibaba.druid.pool.DruidDataSource;  
import com.alibaba.druid.support.http.StatViewServlet;  
import com.alibaba.druid.support.http.WebStatFilter;  
import org.springframework.boot.context.properties.ConfigurationProperties;  
import org.springframework.boot.web.servlet.FilterRegistrationBean;  
import org.springframework.boot.web.servlet.ServletRegistrationBean;  
import org.springframework.context.annotation.Bean;  
import org.springframework.context.annotation.Configuration;  
  
import javax.sql.DataSource;  
import java.util.HashMap;  
import java.util.Map;  
  
@Configuration  
public class DruidConfig {  
  
 /\*\*  
 将自定义的 Druid数据源添加到容器中，不再让 Spring Boot 自动创建  
 绑定全局配置文件中的 druid 数据源属性到 com.alibaba.druid.pool.DruidDataSource从而让它们生效  
 @ConfigurationProperties(prefix = "spring.datasource")：作用就是将 全局配置文件中  
 前缀为 spring.datasource的属性值注入到 com.alibaba.druid.pool.DruidDataSource 的同名参数中  
 \*/  
 @ConfigurationProperties(prefix = "spring.datasource")  
 @Bean  
 public DataSource druidDataSource() {  
 return new DruidDataSource();  
 }  
  
  
 /\*\*  
 \* 配置 Druid 监控管理后台的Servlet；  
 \* 内置 Servlet 容器时没有web.xml文件，所以使用 Spring Boot 的注册 Servlet 方式  
 \* @return  
 \*/  
 @Bean  
 public ServletRegistrationBean statViewServlet() {  
 ServletRegistrationBean bean = new ServletRegistrationBean(new StatViewServlet(), "/druid/\*");  
  
 // 这些参数可以在 com.alibaba.druid.support.http.StatViewServlet  
 // 的父类 com.alibaba.druid.support.http.ResourceServlet 中找到  
 Map<String, String> initParams = new HashMap<>();  
 //后台管理界面的登录账号  
 initParams.put("loginUsername", "admin");  
 //后台管理界面的登录账号  
 initParams.put("loginPassword", "123");  
  
 //后台允许谁可以访问  
 //initParams.put("allow", "localhost")：表示只有本机可以访问  
 //initParams.put("allow", "")：为空或者为null时，表示允许所有访问  
 initParams.put("allow", "");  
 //deny：Druid 后台拒绝谁访问  
 //initParams.put("zh", "ip地址");表示禁止此ip访问  
  
 //设置初始化参数  
 bean.setInitParameters(initParams);  
 return bean;  
 }  
  
 @Bean  
 public FilterRegistrationBean webStatFilter() {  
 FilterRegistrationBean bean = new FilterRegistrationBean();  
 bean.setFilter(new WebStatFilter());  
 Map<String, String> initParameters = new HashMap<>();  
 //这些东西不进行统计  
 initParameters.put("exclusions", "\*.css,\*.js,/druid/\*");  
 bean.setInitParameters(initParameters);  
 return bean;  
 }  
  
  
}

**src\main\java\edu\ynu\arduino\configuration\SecurityConfig.java**

//package edu.ynu.arduino.configuration;  
//  
//import org.springframework.context.annotation.Bean;  
//import org.springframework.context.annotation.Configuration;  
//import org.springframework.security.config.Customizer;  
//import org.springframework.security.config.annotation.web.builders.HttpSecurity;  
//import org.springframework.security.crypto.password.NoOpPasswordEncoder;  
//import org.springframework.security.crypto.password.PasswordEncoder;  
//import org.springframework.security.web.SecurityFilterChain;  
//  
//@Configuration  
//public class SecurityConfig {  
//// @Bean  
//// UserDetailsService userDetailsService() {  
//// InMemoryUserDetailsManager users = new InMemoryUserDetailsManager();  
//// users.createUser(User.withUsername("swb").password("{noop}123").roles("USER").build());  
//// users.createUser(User.withUsername("admin").password("{noop}123").roles("ADMIN").build());  
//// return users;  
//// }  
//  
//// @Resource  
//// UserService userService;  
////  
//// @Bean  
//// UserDetailsService userDetailsService() {  
//// return userService;  
//// }  
//  
// @Bean  
// SecurityFilterChain securityFilterChain(HttpSecurity http) throws Exception {  
//// 忽略认证的请求，也可以写到application.properties中  
// var getIgnoreToken = new String[] {"/doc.html", "/swagger-ui/\*\*", "/webjars/\*\*", "/v3/\*\*"};  
// http.authorizeRequests()  
// .antMatchers(getIgnoreToken).permitAll()  
// .anyRequest()  
// .authenticated()  
// .and()  
// .formLogin(Customizer.withDefaults())  
//// .loginPage("/login")自定义登录页,需要去除Customizer.withDefaults(),表示不使用默认的登录页  
// .anonymous()  
// .and().csrf().disable();  
//// 如果不将csrf禁用,则会报错(上传文件时)  
// return http.build();  
// }  
// @Bean  
// PasswordEncoder passwordEncoder() {  
// return NoOpPasswordEncoder.getInstance();  
// }  
//}

**src\main\java\edu\ynu\arduino\configuration\SysSpringdocConfig.java**

package edu.ynu.arduino.configuration;  
  
import io.swagger.v3.oas.models.OpenAPI;  
import io.swagger.v3.oas.models.info.Info;  
import io.swagger.v3.oas.models.info.License;  
import org.springdoc.core.GroupedOpenApi;  
import org.springframework.context.annotation.Bean;  
import org.springframework.context.annotation.Configuration;  
//knife4j使用openapi3风格  
@Configuration  
public class SysSpringdocConfig {  
  
 @Bean  
 public GroupedOpenApi sysPlatformApiDoc() {  
 return GroupedOpenApi.builder()  
 .group("sysPlatform-controller")  
 .packagesToScan("edu.ynu.arduino.controller")  
 .build();  
 }  
  
 @Bean  
 public OpenAPI customOpenAPI() {  
 return new OpenAPI()  
 .info(new Info()  
 .title("arduino数据监控管理")  
 .version("1.0.0")  
 .description("数据管理平台")  
 .termsOfService("http://doc.xiaominfo.com")  
 .license(new License().name("Apache 2.0")  
 .url("http://doc.xiaominfo.com"))  
 );  
 }  
  
  
}

**src\main\java\edu\ynu\arduino\controller\AbstractTypedController.java**

package edu.ynu.arduino.controller;  
  
import edu.ynu.arduino.dao.specification.AbstractQueryCondition;  
import edu.ynu.arduino.entity.AbstractDomainEntity;  
import edu.ynu.arduino.service.AbstractTypedService;  
import io.swagger.v3.oas.annotations.Operation;  
import lombok.extern.slf4j.Slf4j;  
import org.springframework.data.domain.Page;  
import org.springframework.data.domain.Pageable;  
import org.springframework.data.domain.Sort;  
import org.springframework.http.HttpStatus;  
import org.springframework.http.ResponseEntity;  
import org.springframework.lang.Nullable;  
import org.springframework.web.bind.annotation.\*;  
  
import java.util.List;  
  
@Slf4j  
public class AbstractTypedController<T extends AbstractDomainEntity, IdType> {  
  
 /\*\*  
 \* 学生服务  
 \*/  
// @Resource  
 protected AbstractTypedService<T, IdType> svcContext;  
  
 //#region 关联的数据实体  
  
 @Operation(summary = "查询 单个实体")  
 @GetMapping("/{id}")  
 public T queryById(@PathVariable IdType id) {  
 return svcContext.getByIdNotNull(id);  
 }  
  
 @GetMapping("/all")  
 @Operation(summary = "查询 全部实体")  
 public List<T> findAll() {  
 return svcContext.findAll();  
 }  
  
 @GetMapping("/by")  
 @Operation(summary = "查询 符合条件的所有实体")  
 public List<T> findBy(@ModelAttribute AbstractQueryCondition condition, @Nullable Sort sort) {  
 return svcContext.findBy(condition, sort);  
 }  
  
 /\*\*  
 \* @param pageable 分页  
 \* @return  
 \*/  
 @Operation(summary = "查询 分页")  
 @RequestMapping(value = "/page", method = RequestMethod.GET)  
 public Page<T> pageQuery(Pageable pageable) {  
 return svcContext.pageQuery(pageable);  
 }  
  
 /\*\*  
 \* 根据条件分页查询对象  
 \*  
 \* @param condition 查询条件  
 \* @param pageable 分页  
 \* @return  
 \*/  
 public Page<T> pageQueryBy(@ModelAttribute AbstractQueryCondition condition, Pageable pageable) {  
 return svcContext.pageQuery(pageable, condition);  
 }  
  
 //#endregion  
  
 //#region 关联的数据实体基本操作  
  
 @PostMapping  
 @Operation(summary = "创建 数据实体")  
 public T create(@RequestBody T item) {  
 return svcContext.create(item);  
 }  
  
 @PutMapping  
 @Operation(summary = "修改 数据实体")  
 public T update(@RequestBody T item) {  
 return svcContext.update(item);  
 }  
  
 @DeleteMapping("/{id}")  
 @Operation(summary = "删除 数据实体")  
 public ResponseEntity<Void> delete(@PathVariable IdType id) {  
 svcContext.delete(id);  
 return new ResponseEntity<>(HttpStatus.OK);  
 }  
  
 @PostMapping("/batch")  
 @Operation(summary = "批量新建")  
 public ResponseEntity<Void> batchCreate(@RequestBody List<T> items) {  
 var res = svcContext.batchCreate(items);  
 return new ResponseEntity<>(HttpStatus.OK);  
 }  
  
 @PutMapping("/batch")  
 @Operation(summary = "批量更新")  
 public ResponseEntity<Void> batchUpdate(@RequestBody List<T> items) {  
 var res = svcContext.batchUpdate(items);  
 var ids = res.stream().map(r -> r.getEntityId()).toArray();  
  
 log.debug("items: {}", ids.length);  
  
 return new ResponseEntity<>(HttpStatus.OK);  
 }  
  
 @DeleteMapping("/batch")  
 @Operation(summary = "批量删除")  
 public ResponseEntity<Integer> batchDelete(@RequestParam List<IdType> ids) {  
 var res = svcContext.deleteAll(ids);  
 return new ResponseEntity<Integer>(res, HttpStatus.OK);  
 }  
  
 //#endregion  
  
}

**src\main\java\edu\ynu\arduino\controller\EnvironmentDataController.java**

package edu.ynu.arduino.controller;  
  
  
import edu.ynu.arduino.entity.EnvironmentData;  
import edu.ynu.arduino.service.EnvironmentDataService;  
import io.swagger.v3.oas.annotations.Operation;  
import io.swagger.v3.oas.annotations.tags.Tag;  
import lombok.extern.slf4j.Slf4j;  
import org.springframework.data.domain.Page;  
import org.springframework.data.domain.Pageable;  
import org.springframework.web.bind.annotation.GetMapping;  
import org.springframework.web.bind.annotation.PostMapping;  
import org.springframework.web.bind.annotation.RequestMapping;  
import org.springframework.web.bind.annotation.RestController;  
  
import javax.annotation.Resource;  
import javax.websocket.server.PathParam;  
import java.math.BigDecimal;  
import java.text.SimpleDateFormat;  
import java.time.Instant;  
import java.util.Date;  
  
/\*\*  
 \* 控制层  
 \*  
 \* @author shiwenbo  
 \* @since 2022-11-29 11:55:23  
 \*/  
@Slf4j  
@RestController  
@Tag(name = "EnvironmentDataController")  
@RequestMapping("/environmentData")  
public class EnvironmentDataController extends AbstractTypedController<EnvironmentData, String> {  
  
 @Resource  
 private EnvironmentDataService environmentDataService;  
  
 EnvironmentDataController(EnvironmentDataService service) {  
 this.svcContext = service;  
 this.environmentDataService = service;  
 }  
  
 @Operation(summary = "新建数据记录")  
 @PostMapping("/create")  
 public EnvironmentData create(@PathParam("airHumidity") Double airHumidity,  
 @PathParam("light\_intensity") Integer lightIntensity,  
 @PathParam("soil\_moisture") Double soilMoisture,  
 @PathParam("temperature") Double temperature) {  
 EnvironmentData environmentData = new EnvironmentData();  
 environmentData.setAirHumidity(BigDecimal.valueOf(airHumidity));  
 environmentData.setLightIntensity(lightIntensity);  
 environmentData.setSoilMoisture(BigDecimal.valueOf(soilMoisture));  
 environmentData.setTemperature(BigDecimal.valueOf(temperature));  
 environmentData.setTime(Instant.now());  
 return environmentDataService.create(environmentData);  
 }  
  
 @Operation(summary = "数据记录分页查询")  
 @GetMapping("/pageQuery")  
 public Page<EnvironmentData> pageQuery(Pageable pageable) {  
 return environmentDataService.queryEnvironmentDataPage(pageable);  
 }  
  
 @Operation(summary = "数据记录分页查询 按时间排序")  
 @GetMapping("/pageQueryOrderByTime")  
 public Page<EnvironmentData> pageQueryOrderByTime(Pageable pageable) {  
 return environmentDataService.queryEnvironmentDataPageByTime(pageable);  
 }  
}

**src\main\java\edu\ynu\arduino\controller\RoleController.java**

package edu.ynu.arduino.controller;  
   
   
  
  
  
import edu.ynu.arduino.controller.AbstractTypedController;  
import java.lang.String;  
import edu.ynu.arduino.entity.Role;  
import edu.ynu.arduino.service.RoleService;  
import org.springframework.data.domain.Page;  
import org.springframework.data.domain.Pageable;  
import org.springframework.web.bind.annotation.GetMapping;  
import org.springframework.web.bind.annotation.PostMapping;  
import org.springframework.web.bind.annotation.RequestBody;  
import org.springframework.web.bind.annotation.RequestMapping;  
import org.springframework.web.bind.annotation.RestController;  
import io.swagger.v3.oas.annotations.Operation;  
import io.swagger.v3.oas.annotations.Parameter;  
import io.swagger.v3.oas.annotations.tags.Tag;  
import lombok.extern.slf4j.Slf4j;  
import javax.annotation.Resource;  
/\*\*  
 \* 控制层  
 \*  
 \* @author shiwenbo  
 \* @since 2022-11-29 11:55:24  
 \*/  
@Slf4j  
@RestController  
@Tag(name = "RoleController")  
@RequestMapping("/role")  
public class RoleController extends AbstractTypedController<Role, String>{  
   
 @Resource  
 private RoleService roleService;  
   
 RoleController(RoleService service) {  
 this.svcContext = service;  
 this.roleService = service;  
 }  
}

**src\main\java\edu\ynu\arduino\controller\UserController.java**

package edu.ynu.arduino.controller;  
  
  
import edu.ynu.arduino.entity.User;  
import edu.ynu.arduino.service.UserService;  
import io.swagger.v3.oas.annotations.Operation;  
import io.swagger.v3.oas.annotations.Parameter;  
import io.swagger.v3.oas.annotations.tags.Tag;  
import lombok.extern.slf4j.Slf4j;  
import org.springframework.web.bind.annotation.\*;  
  
import javax.annotation.Resource;  
  
/\*\*  
 \* 控制层  
 \*  
 \* @author shiwenbo  
 \* @since 2022-11-29 11:55:25  
 \*/  
@Slf4j  
@RestController  
@Tag(name = "UserController")  
@RequestMapping("/user")  
public class UserController extends AbstractTypedController<User, String> {  
  
 @Resource  
 private UserService userService;  
  
 UserController(UserService service) {  
 this.svcContext = service;  
 this.userService = service;  
 }  
  
 @Operation(summary = "login")  
 @PostMapping("/login")  
 public User login(@RequestParam("username") String username,  
 @RequestParam("password") String password) {  
 return userService.login(username, password);  
 }  
  
 @Operation(summary = "register")  
 @PostMapping("/register")  
 public User register(@RequestParam("username") String username,  
 @RequestParam("password") String password) {  
 return userService.register(username, password);  
 }  
}

**src\main\java\edu\ynu\arduino\controller\User\_RoleController.java**

package edu.ynu.arduino.controller;  
   
   
  
  
  
import edu.ynu.arduino.controller.AbstractTypedController;  
import java.lang.String;  
import edu.ynu.arduino.entity.User\_Role;  
import edu.ynu.arduino.service.User\_RoleService;  
import org.springframework.data.domain.Page;  
import org.springframework.data.domain.Pageable;  
import org.springframework.web.bind.annotation.GetMapping;  
import org.springframework.web.bind.annotation.PostMapping;  
import org.springframework.web.bind.annotation.RequestBody;  
import org.springframework.web.bind.annotation.RequestMapping;  
import org.springframework.web.bind.annotation.RestController;  
import io.swagger.v3.oas.annotations.Operation;  
import io.swagger.v3.oas.annotations.Parameter;  
import io.swagger.v3.oas.annotations.tags.Tag;  
import lombok.extern.slf4j.Slf4j;  
import javax.annotation.Resource;  
/\*\*  
 \* 控制层  
 \*  
 \* @author shiwenbo  
 \* @since 2022-11-29 11:55:25  
 \*/  
@Slf4j  
@RestController  
@Tag(name = "User\_RoleController")  
@RequestMapping("/user\_Role")  
public class User\_RoleController extends AbstractTypedController<User\_Role, String>{  
   
 @Resource  
 private User\_RoleService user\_RoleService;  
   
 User\_RoleController(User\_RoleService service) {  
 this.svcContext = service;  
 this.user\_RoleService = service;  
 }  
}

**src\main\java\edu\ynu\arduino\dao\AbstractDao.java**

package edu.ynu.arduino.dao;  
  
import com.sun.istack.Nullable;  
import edu.ynu.arduino.dao.specification.SpecificationBuilder;  
import edu.ynu.arduino.utils.DaoUtils;  
import lombok.SneakyThrows;  
import org.springframework.data.domain.Page;  
import org.springframework.data.domain.Pageable;  
import org.springframework.data.jpa.domain.Specification;  
import org.springframework.data.jpa.repository.JpaRepository;  
import org.springframework.data.jpa.repository.JpaSpecificationExecutor;  
import org.springframework.data.repository.NoRepositoryBean;  
  
import java.util.Collection;  
import java.util.List;  
import java.util.Map;  
  
/\*\*  
 \* 抽象dao  
 \*  
 \* @param <T>  
 \*/  
@NoRepositoryBean  
public interface AbstractDao<T, ID>  
 extends JpaRepository<T, ID>  
 , JpaSpecificationExecutor<T> {  
  
 default List<T> findByPrimaryKeyIn(Collection<ID> ids) {  
 return DaoUtils.findByPrimaryKeyIn(this, ids);  
 }  
  
 default List<T> findByPrimaryKeyNotIn(Collection<ID> ids) {  
 return DaoUtils.findByPrimaryKeyNotIn(this, ids);  
 }  
  
 default <S extends T> List<S> safeSaveAll(Collection<S> entities) {  
 return DaoUtils.safeSaveAll(this, entities);  
 }  
  
 /\*\*  
 \* 根据id删除所有  
 \*  
 \* @param ids id集合  
 \* @return 删除的数量  
 \*/  
 int deleteByIdIn(Collection<ID> ids);  
  
 default void safeDeleteAll(Collection<T> entities) {  
 DaoUtils.safeDeleteAll(this, entities);  
 }  
  
 default void deleteByPrimaryKeyIn(Collection<ID> ids) {  
 DaoUtils.deleteByPrimaryKeyIn(this, ids);  
 }  
  
 @SneakyThrows  
 default Map<ID, T> findMapByPrimaryKeyIn(Collection<ID> ids) {  
 return DaoUtils.findMapByPrimaryKeyIn(this, ids);  
 }  
  
 /\*\*  
 \* @param pageable  
 \* @return  
 \*/  
 default Page<T> queryPage(Pageable pageable, @Nullable Specification<T> spec) {  
 if (spec == null) {  
 spec = (Specification<T>) SpecificationBuilder.builder().build();  
 }  
  
 return this.findAll(spec, pageable);  
 }  
}

**src\main\java\edu\ynu\arduino\dao\EnvironmentDataDao.java**

package edu.ynu.arduino.dao;  
   
   
  
  
import edu.ynu.arduino.dao.AbstractDao;  
import java.lang.String;  
import edu.ynu.arduino.entity.EnvironmentData;  
import org.springframework.stereotype.Repository;  
  
/\*\*  
 \* 持久层  
 \*  
 \* @author shiwenbo  
 \* @since 2022-11-29 11:55:23  
 \*/  
@Repository  
public interface EnvironmentDataDao extends AbstractDao<EnvironmentData, String> {  
  
}

**src\main\java\edu\ynu\arduino\dao\RoleDao.java**

package edu.ynu.arduino.dao;  
   
   
  
  
import edu.ynu.arduino.dao.AbstractDao;  
import java.lang.String;  
import edu.ynu.arduino.entity.Role;  
import org.springframework.stereotype.Repository;  
  
/\*\*  
 \* 持久层  
 \*  
 \* @author shiwenbo  
 \* @since 2022-11-29 11:55:23  
 \*/  
@Repository  
public interface RoleDao extends AbstractDao<Role, String> {  
  
}

**src\main\java\edu\ynu\arduino\dao\UserDao.java**

package edu.ynu.arduino.dao;  
   
   
  
  
import edu.ynu.arduino.dao.AbstractDao;  
import java.lang.String;  
import edu.ynu.arduino.entity.User;  
import org.springframework.stereotype.Repository;  
  
/\*\*  
 \* 持久层  
 \*  
 \* @author shiwenbo  
 \* @since 2022-11-29 11:55:25  
 \*/  
@Repository  
public interface UserDao extends AbstractDao<User, String> {  
 User findByUsernameAndPassword(String username, String password);  
 User findByUsername(String username);  
}

**src\main\java\edu\ynu\arduino\dao\User\_RoleDao.java**

package edu.ynu.arduino.dao;  
   
   
  
  
import edu.ynu.arduino.dao.AbstractDao;  
import java.lang.String;  
import edu.ynu.arduino.entity.User\_Role;  
import org.springframework.stereotype.Repository;  
  
/\*\*  
 \* 持久层  
 \*  
 \* @author shiwenbo  
 \* @since 2022-11-29 11:55:25  
 \*/  
@Repository  
public interface User\_RoleDao extends AbstractDao<User\_Role, String> {  
  
}

**src\main\java\edu\ynu\arduino\entity\AbstractDomainEntity.java**

package edu.ynu.arduino.entity;  
  
import io.swagger.v3.oas.annotations.media.Schema;  
import lombok.Data;  
import org.hibernate.annotations.GenericGenerator;  
import org.springframework.data.annotation.CreatedDate;  
import org.springframework.data.annotation.LastModifiedDate;  
import org.springframework.data.jpa.domain.support.AuditingEntityListener;  
  
import javax.persistence.\*;  
import javax.validation.constraints.Size;  
import java.io.Serializable;  
import java.util.Date;  
  
@Data  
@MappedSuperclass  
@EntityListeners(AuditingEntityListener.class)  
public class AbstractDomainEntity  
 implements Cloneable, Serializable {  
 /\*\*  
 \* 统一物理主键 uuid 版本，需要提供给子类访问所以用protected  
 \*/  
 @Size(max = 32)  
 @Id  
 @GenericGenerator(name = "jpa-uuid", strategy = "uuid")  
 @GeneratedValue(generator = "jpa-uuid")  
 @Column(name = "id", nullable = false, length = 32)  
 @Schema(description = "id")  
 protected String id;  
 /\*\*  
 \* 创建日期  
 \*/  
 @Schema(description = "实体创建时间")  
 @CreatedDate  
 protected Date createdDate;  
  
 /\*\*  
 \* 最后更新日期 Timestamp  
 \*/  
 @Schema(description = "实体最后更新时间")  
 @LastModifiedDate  
 private Date lastModifiedDate;  
  
 /\*\*  
 \* 获取实体的id  
 \*  
 \* @return  
 \*/  
 public Object getEntityId() {  
 return id;  
 }  
  
 /\*\*  
 \* 克隆当前对象  
 \*  
 \* @return AbstractDomainEntity  
 \* @throws CloneNotSupportedException  
 \*/  
 @Override  
 public Object clone() throws CloneNotSupportedException {  
 return super.clone();  
 }  
  
}

**src\main\java\edu\ynu\arduino\entity\EnvironmentData.java**

package edu.ynu.arduino.entity;  
  
import lombok.\*;  
import org.hibernate.Hibernate;  
  
import javax.persistence.Column;  
import javax.persistence.Entity;  
import java.math.BigDecimal;  
import java.time.Instant;  
import java.util.Objects;  
  
@Getter  
@Setter  
@ToString  
@RequiredArgsConstructor  
@Entity  
@Builder  
@AllArgsConstructor  
public class EnvironmentData extends AbstractDomainEntity {  
  
 @Column(name = "light\_intensity")  
 private Integer lightIntensity;  
  
 @Column(name = "temperature", precision = 4, scale = 2)  
 private BigDecimal temperature;  
  
 @Column(name = "air\_humidity", precision = 5, scale = 2)  
 private BigDecimal airHumidity;  
  
 @Column(name = "soil\_moisture", precision = 5, scale = 2)  
 private BigDecimal soilMoisture;  
  
 @Column(name = "time")  
 private Instant time;  
  
 @Override  
 public boolean equals(Object o) {  
 if (this == o) return true;  
 if (o == null || Hibernate.getClass(this) != Hibernate.getClass(o)) return false;  
 EnvironmentData that = (EnvironmentData) o;  
 return id != null && Objects.equals(id, that.id);  
 }  
  
 @Override  
 public int hashCode() {  
 return getClass().hashCode();  
 }  
}

**src\main\java\edu\ynu\arduino\entity\Role.java**

package edu.ynu.arduino.entity;  
  
import lombok.\*;  
import org.hibernate.Hibernate;  
  
import javax.persistence.Entity;  
import java.util.Objects;  
  
@Getter  
@Setter  
@ToString  
@RequiredArgsConstructor  
@Entity  
@AllArgsConstructor  
public class Role extends AbstractDomainEntity{  
 private String name;  
  
 @Override  
 public boolean equals(Object o) {  
 if (this == o) return true;  
 if (o == null || Hibernate.getClass(this) != Hibernate.getClass(o)) return false;  
 Role role = (Role) o;  
 return id != null && Objects.equals(id, role.id);  
 }  
  
 @Override  
 public int hashCode() {  
 return getClass().hashCode();  
 }  
}

**src\main\java\edu\ynu\arduino\entity\User.java**

package edu.ynu.arduino.entity;  
  
import com.fasterxml.jackson.annotation.JsonIgnore;  
import io.swagger.v3.oas.annotations.media.Schema;  
import lombok.\*;  
import org.hibernate.Hibernate;  
import org.springframework.security.core.GrantedAuthority;  
import org.springframework.security.core.authority.SimpleGrantedAuthority;  
import org.springframework.security.core.userdetails.UserDetails;  
  
import javax.management.relation.Role;  
import javax.persistence.Column;  
import javax.persistence.Entity;  
import javax.persistence.Transient;  
import javax.validation.constraints.NotNull;  
import javax.validation.constraints.Size;  
import java.util.Collection;  
import java.util.List;  
import java.util.Objects;  
import java.util.stream.Collectors;  
  
@Getter  
@Setter  
@ToString  
@RequiredArgsConstructor  
@Builder  
@AllArgsConstructor  
@Entity  
public class User extends AbstractDomainEntity implements UserDetails {  
  
 @Size(max = 20)  
 @NotNull  
 @Column(name = "password", nullable = false, length = 20)  
 @Schema(description = "用户密码")  
 private String password;  
  
 @Size(max = 20)  
 @NotNull  
 @Column(name = "username", nullable = false, length = 20)  
 @Schema(description = "用户名")  
 private String username;  
  
 @NotNull  
 @Column(name = "sex", nullable = false)  
 @Schema(description = "用户性别")  
 private Integer sex;  
  
 @Schema(description = "用户是否没有过期")  
 private boolean accountNonExpired;  
  
 @Schema(description = "用户是否没有被锁定")  
 private boolean accountNonLocked;  
  
 @Schema(description = "用户凭证是否没有过期")  
 private boolean credentialsNonExpired;  
  
 @Schema(description = "用户是否可用")  
 private boolean enabled;  
  
 @Transient  
 @JsonIgnore  
 private List<Role> roles;  
  
 /\*\*  
 \* 获取用户权限  
 \* 使用stream流将roles中的role转换为SimpleGrantedAuthority  
 \* 使用改函数时需要主义，多表连接只在service中完成，当service中为本对象传输了roles，该函数才不会返回空值  
 \* @return  
 \*/  
 /\*\*  
 \* 当前属性不存入数据库  
 \*/  
 @Transient  
 @Override  
 /\*\*客户端调用的时候，这个会在json中忽略\*/  
 @JsonIgnore  
 public Collection<? extends GrantedAuthority> getAuthorities() {  
// TODO:完成查询  
// List<SimpleGrantedAuthority> authorities = new ArrayList<>();  
// for (Role role : getRoles()) {  
// authorities.add(new SimpleGrantedAuthority(role.getName()));  
// }  
// return authorities;  
 /\*\*用map函数完成List<roles>向List<SimpleCrantedAuthority>的转换\*/  
 return roles.stream().map(r -> new SimpleGrantedAuthority(r.getRoleName())).collect(Collectors.toList());  
 }  
  
 @Override  
 public String getPassword() {  
 return password;  
 }  
  
 @Override  
 public String getUsername() {  
 return username;  
 }  
  
 @Override  
 public boolean isAccountNonExpired() {  
 return accountNonExpired;  
 }  
  
 @Override  
 public boolean isAccountNonLocked() {  
 return accountNonLocked;  
 }  
  
 @Override  
 public boolean isCredentialsNonExpired() {  
 return credentialsNonExpired;  
 }  
  
 @Override  
 public boolean isEnabled() {  
 return enabled;  
 }  
  
 @Override  
 public boolean equals(Object o) {  
 if (this == o) return true;  
 if (o == null || Hibernate.getClass(this) != Hibernate.getClass(o)) return false;  
 User user = (User) o;  
 return id != null && Objects.equals(id, user.id);  
 }  
  
 @Override  
 public int hashCode() {  
 return getClass().hashCode();  
 }  
}

**src\main\java\edu\ynu\arduino\entity\User\_Role.java**

package edu.ynu.arduino.entity;  
  
import lombok.\*;  
import org.hibernate.Hibernate;  
  
import javax.persistence.Column;  
import javax.persistence.Entity;  
import javax.validation.constraints.NotNull;  
import javax.validation.constraints.Size;  
import java.util.Objects;  
  
@Getter  
@Setter  
@ToString  
@RequiredArgsConstructor  
@AllArgsConstructor  
@Entity  
public class User\_Role extends AbstractDomainEntity {  
  
 @NotNull  
 @Size(max = 32)  
 @Column(name = "user$id", nullable = false, length = 32)  
 private String user$id;  
  
 @NotNull  
 @Size(max = 32)  
 @Column(name = "role$id", nullable = false, length = 32)  
 private String role$id;  
  
 @Override  
 public boolean equals(Object o) {  
 if (this == o) return true;  
 if (o == null || Hibernate.getClass(this) != Hibernate.getClass(o)) return false;  
 User\_Role user\_role = (User\_Role) o;  
 return id != null && Objects.equals(id, user\_role.id);  
 }  
  
 @Override  
 public int hashCode() {  
 return getClass().hashCode();  
 }  
}

**src\main\java\edu\ynu\arduino\service\AbstractTypedService.java**

package edu.ynu.arduino.service;  
  
import edu.ynu.arduino.dao.AbstractDao;  
import edu.ynu.arduino.dao.specification.AbstractQueryCondition;  
import edu.ynu.arduino.dao.specification.SpecificationBuilder;  
import edu.ynu.arduino.entity.AbstractDomainEntity;  
import edu.ynu.arduino.entity.EnvironmentData;  
import io.swagger.v3.oas.annotations.Operation;  
import org.springframework.data.domain.Page;  
import org.springframework.data.domain.Pageable;  
import org.springframework.data.domain.Sort;  
import org.springframework.data.jpa.domain.Specification;  
import org.springframework.lang.Nullable;  
import org.springframework.util.CollectionUtils;  
  
import javax.validation.Valid;  
import java.util.Collection;  
import java.util.LinkedList;  
import java.util.List;  
import java.util.concurrent.locks.Condition;  
  
/\*\*  
 \*  
 \* @param <T>  
 \* @param <IdType>  
 \*/  
public class AbstractTypedService<T extends AbstractDomainEntity, IdType>  
{  
 /\*\*  
 \* 配套的类型化dao  
 \*/  
 protected AbstractDao<T, IdType> dataContext;  
  
  
 //#region 实体类型化访问  
  
  
 @Operation(summary = "根据id获取数据对象")  
 public T getById(IdType id) {  
 return dataContext.getOne(id);  
 }  
  
  
 @Operation(summary = "根据id获取数据对象, 如果没有找到则引发一个异常")  
 public T getByIdNotNull(IdType id) {  
 return dataContext.findById(id).orElseThrow(() -> new IllegalArgumentException("无法找到请求的数据"));  
 }  
  
  
 @Operation(summary = "删除数据对象")  
 public List<T> findByIds(Collection<IdType> ids) {  
 if (CollectionUtils.isEmpty(ids)) {  
 return new LinkedList<>();  
 }  
 return dataContext.findByPrimaryKeyIn(ids);  
 }  
  
 /\*\*  
 \* 查询全部  
 \*  
 \* @return  
 \*/  
  
 @Operation(summary = "创建数据对象")  
 public List<T> findAll() {  
 return dataContext.findAll();  
 }  
  
 /\*\*  
 \* 查询全部  
 \*  
 \* @return  
 \*/  
 @Operation(summary = "根据条件查询数据对象")  
 public List<T> findBy(AbstractQueryCondition condition, @Nullable Sort sort) {  
 var sb = condition.builderCondition();  
  
 if (sort == null) {  
 sort = Sort.by(Sort.Direction.DESC, "createDate");  
 }  
  
 return dataContext.findAll(sb, sort);  
 }  
  
 /\*\*  
 \* 分页查询数据对象  
 \*  
 \* @param pageable 分页符  
 \* @return  
 \*/  
 @Operation(summary = "分页查询数据对象")  
 public Page<T> pageQuery(Pageable pageable) {  
 return this.pageQuery(pageable, null);  
 }  
  
 /\*\*  
 \* 分页查询数据对象  
 \*  
 \* @param pageable 分页符  
 \* @param condition 查询条件  
 \* @return  
 \*/  
  
 @Operation(summary = "分页查询数据对象")  
 public Page<T> pageQuery(Pageable pageable, @Valid AbstractQueryCondition condition) {  
  
 // TODO  
 var sb = condition.builderCondition();  
  
 // builderCondition(condition);  
  
 return dataContext.queryPage(pageable, sb);  
 }  
  
  
 @Operation(summary = "创建数据对象")  
 public T create(T item) {  
 return dataContext.save(item);  
 }  
  
  
 @Operation(summary = "批量创建数据对象")  
 public List<T> batchCreate(List<T> items) {  
  
 return dataContext.saveAll(items);  
 }  
  
  
 @Operation(summary = "更新数据对象")  
 public T update(T item) {  
 return dataContext.save(item);  
 }  
  
  
 @Operation(summary = "更新数据对象")  
 public List<T> batchUpdate(List<T> items) {  
  
 return dataContext.saveAll(items);  
 }  
  
  
 @Operation(summary = "删除数据对象")  
 public void delete(IdType id) {  
 T item = getByIdNotNull(id);  
 dataContext.delete(item);  
 }  
  
  
 @Operation(summary = "删除数据对象")  
 public int deleteAll(Collection<IdType> ids) {  
 if (CollectionUtils.isEmpty(ids)) {  
 return 0;  
 }  
 return dataContext.deleteByIdIn(ids);  
 }  
  
 // #endregion  
  
}

**src\main\java\edu\ynu\arduino\service\EnvironmentDataService.java**

package edu.ynu.arduino.service;  
  
  
import edu.ynu.arduino.dao.EnvironmentDataDao;  
import edu.ynu.arduino.entity.EnvironmentData;  
import io.swagger.v3.oas.annotations.Operation;  
import lombok.extern.slf4j.Slf4j;  
import org.springframework.data.domain.Page;  
import org.springframework.data.domain.PageRequest;  
import org.springframework.data.domain.Pageable;  
import org.springframework.data.jpa.domain.Specification;  
import org.springframework.stereotype.Service;  
  
import javax.annotation.Resource;  
  
/\*\*  
 \* 业务层  
 \*  
 \* @author shiwenbo  
 \* @since 2022-11-29 11:55:23  
 \*/  
@Slf4j  
@Service  
public class EnvironmentDataService extends AbstractTypedService<EnvironmentData, String> {  
  
 @Resource  
 private EnvironmentDataDao environmentDataDao;  
  
 public EnvironmentDataService(EnvironmentDataDao dao) {  
 this.dataContext = dao;  
 this.environmentDataDao = dao;  
 }  
  
 public Page<EnvironmentData> queryEnvironmentDataPage(Pageable page) {  
 Specification<EnvironmentData> spec = (root, query, criteriaBuilder) ->  
 query.orderBy(  
 criteriaBuilder.desc(root.get("id"))  
 ).getRestriction();  
 return environmentDataDao.queryPage(page, spec);  
 }  
  
 @Operation(summary = "按照时间排序分页查询")  
 public Page<EnvironmentData> queryEnvironmentDataPageByTime(Pageable page) {  
// 前端不从1开始  
 Pageable page1 = PageRequest.of(page.getPageNumber() - 1, page.getPageSize());  
 Specification<EnvironmentData> spec = (root, query, criteriaBuilder) ->  
 query.orderBy(  
 criteriaBuilder.desc(root.get("time"))  
 ).getRestriction();  
 return environmentDataDao.queryPage(page1, spec);  
 }  
}

**src\main\java\edu\ynu\arduino\service\RoleService.java**

package edu.ynu.arduino.service;  
   
   
  
  
import edu.ynu.arduino.service.AbstractTypedService;  
import java.lang.String;  
import edu.ynu.arduino.entity.Role;  
import edu.ynu.arduino.service.RoleService;  
import edu.ynu.arduino.dao.RoleDao;  
import org.springframework.stereotype.Service;  
import javax.annotation.Resource;  
import org.springframework.data.domain.Page;  
import org.springframework.data.domain.Pageable;  
import java.util.Collection;  
import java.util.List;  
import java.util.stream.Collectors;  
import java.util.stream.StreamSupport;  
import javax.annotation.Resource;  
import lombok.extern.slf4j.Slf4j;  
/\*\*  
 \* 业务层  
 \*  
 \* @author shiwenbo  
 \* @since 2022-11-29 11:55:24  
 \*/  
@Slf4j  
@Service  
public class RoleService extends AbstractTypedService<Role, String> {  
  
 @Resource  
 private RoleDao roleDao;  
  
 public RoleService(RoleDao dao) {  
 this.dataContext = dao;  
 this.roleDao = dao;  
 }  
  
 public Page<Role> queryRoleByPage(Pageable page) {  
 return roleDao.queryPage(page, null);  
 }  
  
}

**src\main\java\edu\ynu\arduino\service\UserService.java**

package edu.ynu.arduino.service;  
   
   
  
  
import edu.ynu.arduino.service.AbstractTypedService;  
import java.lang.String;  
import edu.ynu.arduino.entity.User;  
import edu.ynu.arduino.service.UserService;  
import edu.ynu.arduino.dao.UserDao;  
import org.springframework.stereotype.Service;  
import javax.annotation.Resource;  
import org.springframework.data.domain.Page;  
import org.springframework.data.domain.Pageable;  
import java.util.Collection;  
import java.util.List;  
import java.util.stream.Collectors;  
import java.util.stream.StreamSupport;  
import javax.annotation.Resource;  
import lombok.extern.slf4j.Slf4j;  
  
import static com.alibaba.druid.util.Utils.md5;  
  
/\*\*  
 \* 业务层  
 \*  
 \* @author shiwenbo  
 \* @since 2022-11-29 11:55:25  
 \*/  
@Slf4j  
@Service  
public class UserService extends AbstractTypedService<User, String> {  
  
 @Resource  
 private UserDao userDao;  
  
 public UserService(UserDao dao) {  
 this.dataContext = dao;  
 this.userDao = dao;  
 }  
  
 public Page<User> queryUserByPage(Pageable page) {  
 return userDao.queryPage(page, null);  
 }  
  
 public User login(String username, String password) {  
 User user = userDao.findByUsernameAndPassword(username, password);  
 if (user == null) {  
 log.info("密码错误");  
 return null;  
 }  
  
 return user;  
 }  
 public User register(String username, String password) {  
 User user = userDao.findByUsername(username);  
 if (user != null) {  
 log.info("用户已存在");  
 return null;  
 }  
 User newUser = new User();  
 newUser.setUsername(username);  
// 用md5加密  
// log.info("加密后的密码为：{}", password);  
 newUser.setPassword(password);  
 newUser.setSex(1);  
 newUser.setEnabled(true);  
 newUser.setAccountNonExpired(true);  
 newUser.setAccountNonLocked(true);  
 newUser.setCredentialsNonExpired(true);  
 return userDao.save(newUser);  
 }  
  
}

**src\main\java\edu\ynu\arduino\service\User\_RoleService.java**

package edu.ynu.arduino.service;
  
  
  
  
  
import edu.ynu.arduino.service.AbstractTypedService;
  
import java.lang.String;
  
import edu.ynu.arduino.entity.User\_Role;
  
import edu.ynu.arduino.service.User\_RoleService;
  
import edu.ynu.arduino.dao.User\_RoleDao;
  
import org.springframework.stereotype.Service;
  
import javax.annotation.Resource;
  
import org.springframework.data.domain.Page;
  
import org.springframework.data.domain.Pageable;
  
import java.util.Collection;
  
import java.util.List;
  
import java.util.stream.Collectors;
  
import java.util.stream.StreamSupport;
  
import javax.annotation.Resource;
  
import lombok.extern.slf4j.Slf4j;
  
/\*\*
  
 \* 业务层
  
 \*
  
 \* @author shiwenbo
  
 \* @since 2022-11-29 11:55:25
  
 \*/
  
@Slf4j
  
@Service
  
public class User\_RoleService extends AbstractTypedService<User\_Role, String> {
  
  
 @Resource
  
 private User\_RoleDao user\_RoleDao;
  
  
 public User\_RoleService(User\_RoleDao dao) {
  
 this.dataContext = dao;
  
 this.user\_RoleDao = dao;
  
 }
  
  
 public Page<User\_Role> queryUser\_RoleByPage(Pageable page) {
  
 return user\_RoleDao.queryPage(page, null);
  
 }
  
  
}