CA390 甲鱼智养 SmartTurtle----甲鱼养殖管理系统

CA390 SmartTurtle - soft-shelled turtle breeding management system

项目编号：CA390

目录

甲鱼智养 SmartTurtle - 甲鱼养殖管理系统

一、项目背景

养甲鱼行业是水产养殖行业中一个重要的分支，甲鱼具有很高的经济价值，尤其是在食品和药品市场。随着科技的进步和市场需求的提升，传统的甲鱼养殖模式逐渐显现出许多管理和效率问题。为了解决这些问题，提高养殖效率和管理水平，开发一款智能化的养甲鱼管理系统具有重要意义。

本项目旨在开发一款基于 PHP7.1 和 MySQL 8 的养甲鱼管理系统，系统将涵盖甲鱼养殖、环境控制、喂养、库存管理、销售及数据统计等功能。通过该系统，用户可以实现高效、智能的养殖管理，提升养殖效益。

二、项目目标

1. 提高养殖效率：通过系统化的管理，减少人工操作，提高甲鱼养殖效率。
2. 精准的数据管理：记录甲鱼的各项数据，包括成长、健康状况、喂养情况等，提供全面的数据支持。
3. 智能化的环境控制：通过系统实时监控水质、温度、湿度等环境因素，确保甲鱼健康生长。
4. 库存与销售管理：精准管理饲料、药物等库存，支持甲鱼的销售管理。
5. 报表与数据分析：通过生成报表和数据分析，帮助管理者做出更有依据的决策。

三、需求分析

本系统的主要用户包括养殖场管理员、养殖人员、销售人员等。系统应提供以下主要功能：

1. 用户与权限管理：支持不同角色的用户注册、登录、权限分配等。
2. 甲鱼养殖管理：记录甲鱼信息、成长记录、健康管理、繁殖管理等。
3. 饲料与投喂管理：记录饲料信息和投喂计划，确保甲鱼的合理喂养。
4. 水质与环境管理：实时监控水质、温度、湿度等因素，确保环境适宜甲鱼生长。
5. 库存与销售管理：管理饲料、药品等库存，并记录甲鱼销售情况。
6. 数据统计与报表：生成各类数据报告，如成长报告、销售报告、健康报告等。
7. 系统设置与运维：提供系统配置、备份、恢复和日志管理功能。

四、技术可行性

1. 开发技术：本系统将使用 PHP7.1 和 MySQL 8 作为技术栈，PHP7.1 具有较强的兼容性和扩展性，能够支持本系统的功能需求；MySQL 8 是目前成熟且高效的关系型数据库系统，能够处理大规模的数据存储和查询需求。

2. MVC 架构：本系统将采用 MVC（Model-View-Controller）架构，确保系统的可维护性和扩展性。通过将数据处理（Model）、业务逻辑（Controller）和界面展示（View）分离，提高系统的可操作性和可测试性。

3. 前端技术：为了保证系统的易用性和响应性，前端将采用简洁的 HTML、CSS 和 PHP 渲染页面，尽量避免复杂的 JavaScript 代码，提高系统稳定性。

4. 安全性：系统将采用用户认证与权限控制机制，确保数据的安全性和用户隐私保护。同时，数据库操作将使用 PDO 防止 SQL 注入攻击，增强系统的安全

性。

五、经济可行性

1. 开发成本：开发该系统需要一定的技术人员（如开发人员、数据库管理员等），预计开发周期为 3-4 个月，成本主要集中在开发人员的薪酬和项目管理费用上。

2. 运营成本：系统的运营维护成本相对较低，主要包括服务器租赁、域名注册、备份存储和技术支持等费用。通过使用现有的技术栈，可以降低开发和运营成本。

3. 经济效益：

- 提高效率：通过系统化管理，可以减少人工操作，提升工作效率，节省管理成本。
- 增加销售：通过系统的库存和销售管理，能够更准确地进行产品销售和客户管理，提升甲鱼销售的利润。
- 数据分析：系统能够提供精准的报表与数据分析，帮助管理人员做出更加科学的经营决策，从而增加整体效益。

六、操作可行性

1. 系统操作简便：系统采用简单易用的界面，支持常见的操作流程如甲鱼信息录入、库存管理、销售记录等。即使没有专业的技术人员，也能够迅速上手。

2. 培训与支持：为保证系统的顺利运行，开发团队将提供操作手册和培训支持，帮助用户理解并熟练使用系统。同时，开发团队会提供一段时间的技术支持，确保系统在使用过程中顺利运行。

3. 系统扩展性：系统采用模块化设计，可以根据需求进行功能扩展，如后期增

加新的水质监测功能、添加移动端应用等。

七、法律与伦理可行性

1. 数据隐私与安全：系统将遵守相关的数据隐私保护法律法规，确保用户数据的安全。系统将采用加密技术保护敏感数据，如用户账号信息、销售记录等。

2. 合法合规性：系统的使用和数据处理将严格遵守相关的水产养殖、食品安全和电子商务法规，确保运营合法合规。

八、开发设计

本项目的养甲鱼管理系统在技术、经济、操作和法律等方面均具备可行性。通过该系统的开发和实施，能够提高养殖管理效率、降低运营成本，并且为管理者提供科学决策支持。系统具备良好的扩展性，能够适应未来的需求变化，因此本项目值得实施。

建议通过详细的需求分析、系统设计和开发计划，逐步推进该项目的实施。

8.1. 用户与权限管理

用户注册与登录：支持管理员、养殖人员等不同角色的注册与登录。

权限控制：为不同角色分配不同权限，如查看、编辑、删除数据等。

用户资料管理：管理员可以查看和修改用户资料。

8.2. 甲鱼养殖管理

甲鱼信息管理：记录甲鱼的基本信息，如种类、数量、入池日期、健康状况等。

甲鱼成长记录：记录每只甲鱼的成长情况，包括体重、体长、鳞片等数据。

喂养记录：记录喂养情况，包括喂食时间、饲料种类、数量等。

健康管理：记录疾病预防、药物使用、治疗过程等信息。

繁殖管理：记录甲鱼的繁殖情况，包括配种、孵化、出壳等信息。

8.3. 饲料与投喂管理

饲料管理：管理不同种类的饲料信息，如饲料名称、成分、供应商、价格等。

投喂计划：根据甲鱼的生长阶段制定投喂计划，并记录实际投喂情况。

8.4. 水质与环境管理

水质监控：记录池塘水质的各项数据，如温度、PH 值、溶氧量等。

环境控制：记录池塘环境的变化，调整温度、湿度等因素，确保甲鱼的最佳生长环境。

8.5. 库存与销售管理

库存管理：记录甲鱼、饲料、药物等物资的库存情况。

销售管理：记录甲鱼的销售情况，包括销售日期、数量、价格、客户信息等。

8.6. 数据统计与报表

成长报告：生成甲鱼的成长数据报告，帮助养殖人员评估养殖情况。

健康报告：生成甲鱼健康情况报告，帮助及时发现健康问题。

销售报告：统计销售数据，包括收入、销量等。

8.7. 系统设置与运维

系统设置：包括管理者信息设置、通知设置、日志记录等。

备份与恢复：定期备份数据库，防止数据丢失。

日志管理：记录系统操作日志，以便追踪系统操作。

九、PHP 程序文件架构

为了使用 MVC（ModelViewController）架构开发养甲鱼系统，每个模块可以被拆解为模型、控制器和视图文件。以下是根据每个模块的功能，列出的应该有的 PHP 文件。

9.1. 用户与权限管理

Model（模型）：

`User.php`：处理用户数据的增删改查操作，包含用户验证、角色权限等。

Controller（控制器）：

`UserController.php`：处理用户注册、登录、编辑、删除、权限分配等操作。

View（视图）：

`views/user/login.php`：用户登录界面。

`views/user/register.php`：用户注册界面。

`views/user/profile.php`：用户个人资料页面。

`views/user/manage.php`：管理员管理用户页面。

9.2. 甲鱼养殖管理

Model（模型）：

`Turtle.php`：处理甲鱼信息的增删改查操作，包括种类、数量、入池日期等。

`GrowthRecord.php`：处理甲鱼成长记录的数据存储和操作。

Controller（控制器）：

`TurtleController.php`：管理甲鱼的增、删、改操作。

`GrowthRecordController.php`：管理甲鱼成长记录的操作。

View（视图）：

`views/turtle/index.php`：显示所有甲鱼的列表。

`views/turtle/create.php`：甲鱼信息录入界面。

`views/turtle/edit.php`：甲鱼信息编辑界面。

`views/growth_record/index.php`：展示甲鱼成长记录。

9.3. 饲料与投喂管理

Model（模型）：

`Feed.php`：处理饲料信息的增删改查操作。

`FeedingRecord.php`：处理投喂记录的存储和操作。

Controller（控制器）：

`FeedController.php`：管理饲料信息的增、删、改操作。

`FeedingRecordController.php`：管理投喂记录的增、删、改操作。

View（视图）：

`views/feed/index.php`：显示所有饲料信息。

`views/feed/create.php`：饲料信息录入页面。

`views/feeding_record/index.php`：展示投喂记录。

9.4. 水质与环境管理

Model（模型）：

`WaterQuality.php`：处理水质记录的增删改查操作。

`Environment.php`：处理环境控制记录的存储和操作。

Controller（控制器）：

`WaterQualityController.php`：管理水质记录的增、删、改操作。

`EnvironmentController.php`：管理环境控制记录的操作。

View（视图）：

`views/water_quality/index.php`：显示水质监控记录。

`views/environment/index.php`：显示环境控制记录。

9.5. 库存与销售管理

Model（模型）：

`Inventory.php`：处理库存信息的增删改查操作，包括饲料、药品、甲鱼等。

`Sale.php`：处理销售订单数据的增删改查。

Controller（控制器）：

`InventoryController.php`：管理库存记录的增、删、改操作。

`SaleController.php`：管理销售订单的增、删、改操作。

View（视图）：

`views/inventory/index.php`：显示库存信息。

`views/inventory/create.php`：库存信息录入页面。

`views/sale/index.php`：展示销售记录。

`views/sale/create.php`：录入销售订单。

9.6. 数据统计与报表

Model（模型）：

`Report.php`：处理数据报表生成的逻辑，如甲鱼成长报表、健康报表等。

Controller（控制器）：

`ReportController.php`：生成各种报表，如成长报告、健康报告、销售报告等。

View（视图）：

`views/report/growth.php`：甲鱼成长报告页面。

`views/report/health.php`：健康报告页面。

`views/report/sales.php`：销售报表页面。

9.7. 系统设置与运维

Model（模型）：

`Setting.php`：处理系统设置的增删改查操作。

`Backup.php`：处理系统备份与恢复。

`Log.php`：处理系统日志记录。

Controller（控制器）：

`SettingController.php`：管理系统设置的增、删、改操作。

`BackupController.php`：管理备份与恢复操作。

`LogController.php`：显示系统操作日志。

View（视图）：

`views/setting/index.php`：显示系统设置页面。

`views/backup/index.php`：显示备份管理页面。

`views/log/index.php`：显示操作日志页面。

提示：

每个模块下的 PHP 文件基本上分为三个部分：模型（Model）用于处理数据库

操作，控制器（Controller）用于协调业务逻辑和视图（View）展示数据。通过这样的结构，系统的扩展性、可维护性将大大增强。

十、MYSQL8 数据库设计

根据上述养甲鱼管理系统的功能需求，我们可以设计相应的 MySQL 数据库结构。以下是基于系统各模块的数据库表设计，涵盖用户管理、甲鱼养殖管理、饲料管理、环境管理、库存管理、销售管理等模块。

10.1. 用户与权限管理

用户表（`users`）

```sql
CREATE TABLE `users` (
    `id` INT AUTO_INCREMENT PRIMARY KEY,
    `username` VARCHAR(100) NOT NULL UNIQUE,
    `password` VARCHAR(255) NOT NULL,
    `role` ENUM('admin', 'staff', 'manager') DEFAULT 'staff',
    `email` VARCHAR(100) NULL,
    `phone` VARCHAR(20) NULL,
    `created_at` TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    `updated_at` TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP
);
```

权限表（`permissions`）

```sql
CREATE TABLE `permissions` (
    `id` INT AUTO_INCREMENT PRIMARY KEY,
    `user_id` INT NOT NULL,
    `module` VARCHAR(100) NOT NULL,
    `can_view` TINYINT(1) DEFAULT 0,
    `can_edit` TINYINT(1) DEFAULT 0,
    `can_delete` TINYINT(1) DEFAULT 0,
    FOREIGN KEY (`user_id`) REFERENCES `users`(`id`)
);
```

10.2. 甲鱼养殖管理

甲鱼表 (`turtles`)

```sql
CREATE TABLE `turtles` (
    `id` INT AUTO_INCREMENT PRIMARY KEY,
    `species` VARCHAR(100) NOT NULL,
    `quantity` INT NOT NULL,
    `entry_date` DATE NOT NULL,
    `health_status` ENUM('healthy', 'sick', 'recovered') DEFAULT 'healthy',
    `notes` TEXT NULL,
    `created_at` TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    `updated_at` TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP
);
```

成长记录表（`growth_records`）

```sql
CREATE TABLE `growth_records` (
    `id` INT AUTO_INCREMENT PRIMARY KEY,
    `turtle_id` INT NOT NULL,
    `weight` DECIMAL(5, 2) NOT NULL,
    `length` DECIMAL(5, 2) NOT NULL,
    `recorded_at` TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (`turtle_id`) REFERENCES `turtles`(`id`)
);
```

10.3. 饲料与投喂管理

饲料表（`feeds`）

```sql
CREATE TABLE `feeds` (
    `id` INT AUTO_INCREMENT PRIMARY KEY,
    `name` VARCHAR(100) NOT NULL,
    `supplier` VARCHAR(100) NULL,
    `price` DECIMAL(10, 2) NOT NULL,
    `stock` INT DEFAULT 0,
    `created_at` TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    `updated_at` TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP
);
```

投喂记录表（`feeding_records`）

```sql
CREATE TABLE `feeding_records` (
    `id` INT AUTO_INCREMENT PRIMARY KEY,
    `feed_id` INT NOT NULL,
    `turtle_id` INT NOT NULL,
    `quantity` DECIMAL(5, 2) NOT NULL,
    `feeding_date` DATE NOT NULL,
    FOREIGN KEY (`feed_id`) REFERENCES `feeds`(`id`),
    FOREIGN KEY (`turtle_id`) REFERENCES `turtles`(`id`)
);
```

10.4. 水质与环境管理

水质记录表（`water_quality`）

```sql
CREATE TABLE `water_quality` (
    `id` INT AUTO_INCREMENT PRIMARY KEY,
    `temperature` DECIMAL(5, 2) NOT NULL,
    `ph` DECIMAL(3, 2) NOT NULL,
    `oxygen_level` DECIMAL(5, 2) NOT NULL,
    `recorded_at` TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

环境控制表（`environment_controls`）

```sql
CREATE TABLE `environment_controls` (
    `id` INT AUTO_INCREMENT PRIMARY KEY,
```

```
    `action` VARCHAR(100) NOT NULL,
    `description` TEXT NULL,
    `performed_at` TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

10.5. 库存与销售管理

库存表 (`inventory`)

```sql
CREATE TABLE `inventory` (
    `id` INT AUTO_INCREMENT PRIMARY KEY,
    `product_name` VARCHAR(100) NOT NULL,
    `quantity` INT NOT NULL,
    `price` DECIMAL(10, 2) NOT NULL,
    `stock_type` ENUM('feed', 'medicine', 'equipment') NOT NULL,
    `created_at` TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    `updated_at` TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP
);
```

销售表 (`sales`)

```sql
CREATE TABLE `sales` (
    `id` INT AUTO_INCREMENT PRIMARY KEY,
    `turtle_id` INT NOT NULL,
    `quantity` INT NOT NULL,
    `price` DECIMAL(10, 2) NOT NULL,
```

```
    `customer_name` VARCHAR(100) NOT NULL,
    `sale_date` DATE NOT NULL,
    FOREIGN KEY (`turtle_id`) REFERENCES `turtles`(`id`)
);
```

10.6. 数据统计与报表

为了生成报表，通常不需要单独的数据库表。报表可以通过在应用层进行数据查询、统计与分析来动态生成，具体的 SQL 查询将根据需要生成不同的报表，如甲鱼成长报表、销售报表等。

10.7. 系统设置与运维

系统设置表（`settings`）

```sql
CREATE TABLE `settings` (
    `id` INT AUTO_INCREMENT PRIMARY KEY,
    `setting_name` VARCHAR(100) NOT NULL,
    `setting_value` VARCHAR(255) NOT NULL,
    `created_at` TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    `updated_at` TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP
);
```

备份记录表（`backups`）

```sql
```

```
CREATE TABLE `backups` (
    `id` INT AUTO_INCREMENT PRIMARY KEY,
    `backup_date` TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    `backup_file` VARCHAR(255) NOT NULL
);
```

操作日志表 (`logs`)

```sql
CREATE TABLE `logs` (
    `id` INT AUTO_INCREMENT PRIMARY KEY,
    `user_id` INT NOT NULL,
    `action` VARCHAR(255) NOT NULL,
    `action_date` TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (`user_id`) REFERENCES `users`(`id`)
);
```

总结

上述数据库设计涵盖了养甲鱼系统中的主要功能模块。每个模块的核心数据都已考虑，并与其他模块进行了合理的关联。此数据库设计能够支撑系统的各项功能，并具备良好的扩展性和数据安全性。

在实际开发过程中，可以根据实际需求进一步优化表结构或添加更多的索引、约束等，以提升系统性能和可维护性。

叶梓阳 总经理

WWW.CAFFZ.COM

广东知周数字科技有限公司