پروژه شارژ و مدیریت ساختمان تحت وب

[



تقدیم به:

تشكر و قدردانی:

با احترام و سپاس فراوان،

چکیده

هدف این پروژه طراحی و پیاده‌سازی یک سیستم مدیریت ساختمان است که بتواند به صورت موثر و کارآمد تمامی امور مربوط به مدیریت ساختمان را ثبت و کنترل نماید. این سیستم به‌منظور ساده‌سازی فرآیندهای مدیریتی و افزایش بهره‌وری در استفاده از منابع و امکانات ساختمان طراحی شده است.

سیستم مدیریت و کاربری ساختمان شامل موارد زیر می باشد:

ویژه مدیریت :

* اضافه کردن آپارتمان
* مشاهده آپارتمان
* محاسبه جریمه دیرکرد
* اضافه کردن جریمه دیرکرد
* اضافه کردن ساکن جدید
* مشاهده شارژ های دریافتی
* مشاهده ساکنین
* پنل اختصاصی مدیریت
* حذف ساکنین
* ورود اختصاصی

ویژه ساکنین :

* ثبت نام و ورود ساکنین
* پرداخت شارژ آسانسور
* پرداخت شارژ پارکینگ
* پرداخت شارژ ماهانه
* دریافت فاکتور

این پروژه به زبان PHP سمت وب نوشته شده است بنابر این می شود در هاست لینوکس پیاده سازی کرده و مدیر ساختمان و ساکنین در هر لحظه و در هر مکان از آن استفاده کند.

کلیدواژه‌ها: مدیریت ساختمان، نرم‌افزار تحت وب، ثبت و رزرو امکانات، مدیریت پرداخت‌ها، تابلو اعلانات، درخواست‌های ساکنین، مدیریت قرار ملاقات‌ها.

فهرست نوشتار

[فصل 1: مقدمه **Error! Bookmark not defined.**](#_Toc57377650)

[1-1- انگیزه 9](#_Toc57377651)

[1-2- مروری بر پیشینه و کارهای مشابه 10](#_Toc57377652)

[1-3- هدف 12](#_Toc57377653)

[1-4- رئوس مطالب سایر فصل‌ها **Error! Bookmark not defined.**](#_Toc57377654)

[فصل 2: تجزیه و تحلیل نیازمندی‌ها 14](#_Toc57377655)

[2-1- مقدمه 15](#_Toc57377656)

[2-2- نتیجه‌گیری 16](#_Toc57377657)

[فصل 3: ساختار داده‌ها و بانک اطلاعات 16](#_Toc57377658)

[3-1- مقدمه 18](#_Toc57377659)

[3-2- طراحی 20](#_Toc57377660)

[3-3- نرم‌افزار مدیریت پایگاه داده‌ها 20](#_Toc57377661)

[3-4- نتیجه‌گیری 27](#_Toc57377662)

[فصل 4: پیاده‌سازی 29](#_Toc57377663)

[4-1- مقدمه 29](#_Toc57377664)

[4-2- کدنویسی 33](#_Toc57377665)

[4-3- اجرای نرم‌افزار 33](#_Toc57377666)

[4-4- نتیجه‌گیری 99](#_Toc57377667)

[فصل 5: جمع‌بندی و پیشنهادها 101](#_Toc57377668)

[5-1- نتیجه‌گیری **Error! Bookmark not defined.**](#_Toc57377669)

[5-2- پیشنهادهایی برای کارهای آتی 102](#_Toc57377670)

[منابع **Error! Bookmark not defined.**](#_Toc57377671)

[پیوست‌ها 104](#_Toc57377672)

فصل اول :

مقدمه

انگیزه

انگیزه‌ی انجام پروژه

ایده‌ی انجام این پروژه از نیاز مبرم به بهبود و کارآمدسازی فرآیندهای مدیریتی ساختمان‌ها نشأت گرفته است. با توجه به پیچیدگی‌های روزافزون در مدیریت ساختمان‌های مسکونی و تجاری، نیاز به یک سیستم جامع و کارآمد برای مدیریت امور مربوط به ساختمان به شدت احساس می‌شود. در بسیاری از ساختمان‌ها، مدیریت امور مربوط به شارژ، رزرو امکانات، پرداخت‌ها، اعلانات و درخواست‌های ساکنین به صورت سنتی و دستی انجام می‌شود که این امر نه تنها وقت‌گیر و هزینه‌بر است، بلکه احتمال بروز خطاهای انسانی و ناهماهنگی‌ها را نیز افزایش می‌دهد.

هدف از انجام این پروژه، ایجاد یک راهکار نوین و دیجیتال برای مدیریت یکپارچه تمامی امور مربوط به ساختمان است. با طراحی و پیاده‌سازی یک سیستم مدیریت ساختمان تحت وب، می‌توانیم فرآیندهای پیچیده را ساده‌تر کنیم، از بروز اشتباهات جلوگیری کرده و تجربه‌ای بهتر و کارآمدتر برای مدیران و ساکنین ساختمان‌ها فراهم آوریم.

علاوه بر این، این پروژه فرصتی منحصر به‌فرد برای به‌کارگیری دانش و مهارت‌های برنامه‌نویسی و توسعه نرم‌افزار در یک زمینه عملی و کاربردی را فراهم می‌آورد. انگیزه اصلی ما از انجام این پروژه، ارائه یک ابزار مدیریتی هوشمند است که بتواند به بهبود کیفیت زندگی ساکنین و کاهش بار کاری مدیران ساختمان کمک کند. این سیستم با هدف افزایش بهره‌وری، کاهش هزینه‌ها و افزایش رضایت کاربران نهایی طراحی شده است. امیدواریم که با اجرای این پروژه، گامی موثر در راستای دیجیتال‌سازی و هوشمندسازی مدیریت ساختمان‌ها برداریم و بتوانیم نقش مثبتی در بهبود عملکرد و رضایت عمومی ایفا کنیم.

مروری بر پیشینه و کارهای مشابه

با توجه به اهمیت مدیریت بهینه ساختمان‌ها و نیاز روزافزون به سیستم‌های مدیریتی پیشرفته، تحقیقات و پروژه‌های متعددی در این زمینه انجام شده است. این بخش به مرور پیشینه و کارهای مشابه در حوزه سیستم‌های مدیریت ساختمان می‌پردازد.

1. سیستم‌های مدیریت ساختمان سنتی:

در گذشته، مدیریت امور مربوط به ساختمان‌ها عمدتاً به‌صورت سنتی و دستی انجام می‌شد. این روش‌ها شامل ثبت دستی پرداخت‌ها، اطلاع‌رسانی از طریق تابلوهای اعلانات فیزیکی و پیگیری درخواست‌های ساکنین به‌صورت حضوری یا تلفنی بودند. این روش‌ها، اگرچه ساده و بدون نیاز به تکنولوژی پیچیده بودند، اما مشکلاتی نظیر بروز خطاهای انسانی، ناهماهنگی در مدیریت اطلاعات و افزایش زمان و هزینه‌های مدیریتی را به همراه داشتند.

2. سیستم‌های کامپیوتری اولیه:

با پیشرفت تکنولوژی و گسترش استفاده از کامپیوترها، نرم‌افزارهای اولیه مدیریت ساختمان به بازار عرضه شدند. این نرم‌افزارها بیشتر به‌صورت آفلاین و بر روی یک کامپیوتر شخصی اجرا می‌شدند. از جمله نمونه‌های اولیه این سیستم‌ها می‌توان به نرم‌افزارهای حسابداری ساده و برنامه‌های مدیریت امور ساختمان اشاره کرد. این نرم‌افزارها کمک شایانی به کاهش بار کاری مدیران ساختمان کردند، اما همچنان محدودیت‌هایی نظیر عدم دسترسی آنلاین و نیاز به نصب و نگهداری محلی داشتند.

3. سیستم‌های مدیریت ساختمان تحت وب و موبایل:

در سال‌های اخیر، با پیشرفت فناوری‌های وب و موبایل، سیستم‌های مدیریت ساختمان به سمت ارائه خدمات آنلاین و ابری حرکت کردند. این سیستم‌ها امکان دسترسی از هر مکان و در هر زمان را برای مدیران و ساکنین فراهم کردند. نمونه‌هایی از این سیستم‌ها شامل نرم‌افزارهای مدیریت ساختمان تحت وب مانند BuildingLink و Condo Control Central هستند. این سیستم‌ها امکاناتی نظیر مدیریت پرداخت‌ها، رزرو امکانات، اطلاع‌رسانی از طریق ایمیل و پیامک، و پیگیری درخواست‌های ساکنین را به‌صورت یکپارچه ارائه می‌دهند.

4. هوشمندسازی و اینترنت اشیا IoT):)

با ظهور تکنولوژی‌های هوشمند و اینترنت اشیا، سیستم‌های مدیریت ساختمان به مرحله جدیدی از تکامل رسیده‌اند. این سیستم‌ها با استفاده از سنسورها و دستگاه‌های هوشمند، امکان نظارت و مدیریت بلادرنگ بر امکانات و تجهیزات ساختمان را فراهم می‌کنند. به‌عنوان مثال، سیستم‌های مدیریت انرژی هوشمند می‌توانند با نظارت بر مصرف انرژی، هزینه‌ها را کاهش داده و کارایی انرژی را افزایش دهند.

کارهای مشابه:

- BuildingLink : یک سیستم مدیریت ساختمان تحت وب که امکان مدیریت همه‌جانبه امور ساختمان‌ها را فراهم می‌کند. امکانات این سیستم شامل مدیریت پرداخت‌ها، رزرو امکانات، اطلاع‌رسانی، پیگیری درخواست‌ها و گزارش‌گیری است.

- Condo Control Central: این نرم‌افزار نیز خدمات مشابهی ارائه می‌دهد و تمرکز آن بر ساده‌سازی فرآیندهای مدیریتی و افزایش ارتباطات بین مدیران و ساکنین است.

- Entrata: یک پلتفرم مدیریت املاک که امکانات گسترده‌ای برای مدیریت ساختمان‌ها و ارتباط با ساکنین ارائه می‌دهد، از جمله پرداخت آنلاین، رزرو امکانات، و مدیریت درخواست‌های تعمیرات.

این پیشینه و بررسی کارهای مشابه نشان می‌دهد که با وجود پیشرفت‌های قابل‌توجه در حوزه مدیریت ساختمان، همچنان نیاز به توسعه سیستم‌های جامع‌تر و کارآمدتر وجود دارد. پروژه‌ی ما با هدف بهبود و یکپارچه‌سازی این فرآیندها، می‌تواند گامی موثر در جهت رفع این نیازها باشد.

هدف

هدف

هدف اصلی از این پروژه طراحی و پیاده‌سازی یک سیستم مدیریت شارژ ساختمان جامع و کارآمد است که بتواند به‌طور موثری فرآیندهای مالی و پرداخت‌های مرتبط با شارژ ساختمان را بهینه‌سازی و ساده‌سازی کند. این سیستم به منظور افزایش شفافیت مالی، کاهش خطاهای انسانی و افزایش رضایت ساکنین طراحی شده است. اهداف مشخص این پروژه عبارتند از:

1. مدیریت دقیق و شفاف پرداخت‌ها

- توسعه یک سیستم یکپارچه برای ثبت و پیگیری تمامی پرداخت‌های شارژ ساختمان.

- فراهم کردن امکان مشاهده و مدیریت صورتحساب‌ها برای ساکنین به‌صورت آنلاین.

2. خودکارسازی فرآیندهای مالی

- ایجاد سازوکاری برای محاسبه خودکار شارژهای ماهانه، هزینه‌های مشترک و سایر مبالغ قابل پرداخت.

- ارسال اعلان‌های پرداخت و یادآوری‌های خودکار به ساکنین از طریق پیامک یا ایمیل.

3. افزایش شفافیت و دسترسی‌پذیری اطلاعات مالی

- ارائه گزارش‌های مالی دقیق و جامع برای مدیران ساختمان به‌منظور بررسی وضعیت پرداخت‌ها و بودجه‌بندی.

- امکان دسترسی ساکنین به گزارش‌های مالی و مشاهده وضعیت پرداخت‌های خود در هر زمان و از هر مکان.

4. بهبود روند جمع‌آوری و مدیریت پرداخت‌ها

- توسعه مکانیزم‌های پرداخت آنلاین از طریق درگاه‌های بانکی مختلف به منظور سهولت در انجام پرداخت‌ها.

- ثبت و پیگیری پرداخت‌های انجام شده و ارائه رسیدهای دیجیتال به ساکنین.

5. کاهش خطاها و بهبود دقت مالی

- کاهش خطاهای انسانی در ثبت و محاسبه پرداخت‌ها با استفاده از سیستم‌های خودکار و هوشمند.

- افزایش دقت و صحت اطلاعات مالی و کاهش اختلافات مالی بین ساکنین و مدیریت ساختمان.

6. تسهیل در پیگیری و مدیریت بدهی‌ها

- امکان پیگیری بدهی‌های معوق و ارسال یادآوری‌های منظم به ساکنین برای پرداخت معوقات.

- ارائه امکاناتی برای تنظیم برنامه‌های پرداخت اقساطی در صورت نیاز.

با تحقق این اهداف، سیستم مدیریت شارژ ساختمان می‌تواند به‌طور موثری به بهبود فرآیندهای مالی و افزایش رضایت ساکنین کمک کرده و نقش مهمی در افزایش شفافیت و کارایی مدیریت مالی ساختمان ایفا کند. این سیستم نه تنها بار کاری مدیران را کاهش می‌دهد، بلکه با ارائه خدمات سریع و دقیق به ساکنین، تجربه کاربری بهتری را برای آنان فراهم می‌کند.

فصل دوم  
تجزیه و تحلیل نیازمندی‌ها

مقدمه

مقدمه

در دنیای امروز، مدیریت مالی و پرداخت‌های مربوط به ساختمان‌ها یکی از چالش‌های اساسی برای مدیران و ساکنین است. با افزایش پیچیدگی‌ها و نیازهای ساختمان‌های مسکونی و تجاری، ضرورت بهره‌گیری از سیستم‌های مدیریت مالی دقیق و کارآمد بیش از پیش احساس می‌شود. پروژه حاضر با هدف طراحی و پیاده‌سازی یک سیستم جامع مدیریت ساختمان، به‌ویژه تمرکز بر مدیریت شارژ و پرداخت‌ها، سعی دارد تا با استفاده از تکنولوژی‌های مدرن و رویکردهای نوین، فرآیندهای مالی را بهینه‌سازی و ساده‌سازی کند.

خصوصیات دقیق سیستم منتخب جهت ساخت

این سیستم تحت وب با استفاده از زبان‌های برنامه‌نویسی HTML، CSS و PHP طراحی شده و از پایگاه‌داده MySQL برای ذخیره‌سازی و مدیریت داده‌ها بهره می‌برد. سیستم شامل ماژول‌های مختلفی برای مدیریت تمامی جنبه‌های ساختمان است، اما تمرکز اصلی آن بر روی ثبت و مدیریت پرداخت‌ها و شارژهای ساختمان می‌باشد. از جمله ویژگی‌های این سیستم می‌توان به ثبت و مدیریت اطلاعات ساختمان، ثبت و رزرو استفاده از امکانات ساختمان، ثبت و مدیریت پرداخت‌ها، ثبت و مدیریت شارژهای ساختمان، ثبت و مدیریت تابلو اعلانات، ثبت و مدیریت درخواست‌های ساکنین از مدیر ساختمان و ثبت و مدیریت قرار ملاقات‌ها اشاره کرد. سیستم به‌گونه‌ای طراحی شده که کاربران می‌توانند به‌صورت آنلاین به تمامی اطلاعات مالی خود دسترسی داشته و پرداخت‌های خود را به‌صورت الکترونیکی انجام دهند.

قطعات مورد استفاده و مبانی تئوری

برای پیاده‌سازی این سیستم از زبان‌های HTML و CSS برای طراحی رابط کاربری، PHP برای منطق سمت سرور و MySQL برای مدیریت پایگاه داده استفاده شده است. این تکنولوژی‌ها به دلیل انعطاف‌پذیری و سازگاری بالا انتخاب شده‌اند. مبانی تئوری به‌کارگرفته‌شده شامل الگوریتم‌های محاسبه خودکار شارژ، اصول طراحی سیستم‌های مدیریت مالی و تکنیک‌های بهینه‌سازی پایگاه داده است. PHP به عنوان یک زبان اسکریپتی سمت سرور، امکان ایجاد صفحات وب پویا را فراهم می‌کند و MySQL به عنوان یک سیستم مدیریت پایگاه‌داده رابطه‌ای، امکان ذخیره‌سازی و بازیابی کارآمد داده‌ها را فراهم می‌سازد.

تحلیل نیازمندی‌های سیستم تحت طراحی

تحلیل نیازمندی‌های سیستم شامل شناسایی سناریوها و نمودارهای تحلیلی مختلف است که به‌منظور ارائه یک سیستم کارآمد و کاربرپسند انجام شده است. سناریوهای مختلفی مانند ثبت پرداخت‌ها، ارسال یادآوری‌های پرداخت، مشاهده گزارش‌های مالی و پیگیری بدهی‌ها بررسی شده و برای هر سناریو، نمودارهای توالی و کلاس طراحی شده است. این تحلیل‌ها کمک کرده‌اند تا نیازهای مختلف کاربران سیستم شناسایی و برآورده شوند.

کاربران و سطح دسترسی‌ها

این سیستم به‌گونه‌ای طراحی شده که سطوح دسترسی مختلفی برای کاربران تعریف شده است. کاربران سیستم شامل مدیران ساختمان و ساکنین هستند. مدیران ساختمان دارای دسترسی کامل به تمامی ماژول‌ها و امکانات سیستم هستند و می‌توانند گزارش‌های مالی را مشاهده، پرداخت‌ها را مدیریت و اعلان‌ها را ارسال کنند. ساکنین نیز می‌توانند به اطلاعات مالی خود دسترسی داشته، پرداخت‌های خود را انجام داده و گزارش‌های مالی خود را مشاهده کنند. سیستم به‌گونه‌ای طراحی شده که امنیت و حریم خصوصی کاربران به‌طور کامل حفظ شود.

تمرکز بر مدیریت شارژ و پرداخت‌ها

بخش اصلی این سیستم مربوط به مدیریت شارژ و پرداخت‌های ساختمان است. این سیستم با فراهم کردن امکان محاسبه خودکار شارژ ماهانه، هزینه‌های مشترک و ارسال اعلان‌های پرداخت به ساکنین، فرآیندهای مالی را ساده‌تر و شفاف‌تر می‌کند. امکان پرداخت آنلاین از طریق درگاه‌های بانکی مختلف، دسترسی به گزارش‌های مالی دقیق و پیگیری بدهی‌های معوق از دیگر ویژگی‌های این بخش است. با خودکارسازی این فرآیندها، احتمال خطاهای انسانی به حداقل رسیده و مدیران می‌توانند به‌صورت موثرتری امور مالی ساختمان را مدیریت کنند.

این پروژه با تحلیل دقیق نیازمندی‌ها و طراحی یک سیستم جامع، تلاش دارد تا تمامی مشکلات و چالش‌های مرتبط با مدیریت شارژ و پرداخت‌ها در ساختمان‌ها را برطرف کرده و تجربه‌ای بهتر و کارآمدتر برای کاربران فراهم آورد.

نتیجه‌گیری

در این قسمت اشاره ای به زبان برنامه نویسی – اهمیت پروژه ـآشنا شدیم و در ادامه ساختار داده‌ها و بانک اطلاعات می پردازیم

فصل سوم

ساختار داده‌ها و بانک اطلاعات

مقدمه

فصل سوم: ساختار داده‌ها و بانک اطلاعات

مقدمه

در پروژه‌های نرم‌افزاری، یکی از مهم‌ترین بخش‌ها طراحی و پیاده‌سازی ساختار داده‌ها و بانک اطلاعات است. این فصل به بررسی ساختار داده‌ها و نحوه مدیریت بانک اطلاعاتی در سیستم مدیریت ساختمان اختصاص دارد. هدف اصلی این فصل، تشریح چگونگی ذخیره‌سازی، سازماندهی و بازیابی داده‌ها به‌صورتی است که سیستم بتواند به‌طور موثری نیازهای کاربران را برآورده سازد.

ساختار داده‌ها و بانک اطلاعاتی

در این پروژه از پایگاه‌داده MySQL به‌عنوان سیستم مدیریت پایگاه‌داده استفاده شده است. MySQL به‌دلیل عملکرد بالا، قابلیت اطمینان و امکانات گسترده، انتخاب مناسبی برای مدیریت داده‌های مرتبط با ساختمان‌ها می‌باشد. در این سیستم، داده‌های مختلفی شامل اطلاعات ساختمان، واحدها، ساکنین، پرداخت‌ها، شارژها، درخواست‌ها، امکانات رزرو شده، اعلان‌ها و قرار ملاقات‌ها ذخیره می‌شود.

مدل‌سازی داده‌ها

برای طراحی بانک اطلاعاتی، ابتدا نیازمندی‌های سیستم تحلیل و شناسایی شده‌اند. سپس بر اساس این نیازمندی‌ها، جداول و روابط بین آن‌ها طراحی شده است. مدل‌سازی داده‌ها به‌صورت دقیق و صحیح، اساس یک سیستم پایگاه‌داده قوی و کارآمد را تشکیل می‌دهد. این مدل‌سازی شامل تعریف جداول، فیلدها، کلیدهای اصلی و خارجی و روابط بین جداول است.

معماری بانک اطلاعاتی

بانک اطلاعاتی این سیستم به‌صورت رابطه‌ای طراحی شده است. در این معماری، داده‌ها در جداول مختلف ذخیره می‌شوند و روابط بین این جداول از طریق کلیدهای اصلی و خارجی برقرار می‌شود. این رویکرد به سیستم اجازه می‌دهد تا داده‌ها را به‌صورت کارآمد و موثر مدیریت کند و دسترسی سریع و مطمئن به اطلاعات را فراهم آورد.

مهم‌ترین جداول و ساختار آن‌ها

در این سیستم، جداول اصلی شامل جداول ساختمان‌ها، واحدها، ساکنین، پرداخت‌ها، شارژها، درخواست‌ها، امکانات، اعلان‌ها و قرار ملاقات‌ها می‌باشد. هر جدول شامل فیلدهایی است که اطلاعات مرتبط با آن جدول را ذخیره می‌کنند. به‌عنوان مثال، جدول پرداخت‌ها شامل فیلدهایی نظیر شناسه پرداخت، شناسه ساکن، مبلغ پرداخت، تاریخ پرداخت و وضعیت پرداخت است.

مدیریت و بهینه‌سازی پایگاه‌داده

برای بهبود عملکرد و کارایی سیستم، از تکنیک‌های مختلفی نظیر ایندکس‌گذاری، نرمال‌سازی داده‌ها و استفاده از کوئری‌های بهینه استفاده شده است. ایندکس‌گذاری بر روی فیلدهای جستجو شونده، سرعت بازیابی داده‌ها را افزایش می‌دهد و نرمال‌سازی داده‌ها کمک می‌کند تا افزونگی داده‌ها کاهش یابد و انسجام داده‌ها حفظ شود.

امنیت و پشتیبان‌گیری

یکی از جنبه‌های حیاتی مدیریت پایگاه‌داده، حفظ امنیت داده‌ها و اجرای پشتیبان‌گیری منظم است. در این سیستم، مکانیزم‌های امنیتی مناسبی برای حفاظت از داده‌ها و دسترسی‌های کاربران پیاده‌سازی شده است. همچنین، سیاست‌های پشتیبان‌گیری منظم برای تضمین بازیابی داده‌ها در صورت بروز مشکلات یا حوادث پیش‌بینی شده است.

در این فصل، جزئیات بیشتری درباره طراحی و پیاده‌سازی ساختار داده‌ها و بانک اطلاعاتی سیستم مدیریت ساختمان ارائه خواهد شد. هدف این است که با ایجاد یک بانک اطلاعاتی قوی و کارآمد، تمامی نیازهای مدیریتی و مالی ساختمان‌ها به‌طور کامل پوشش داده شود و کاربران بتوانند به‌صورت سریع و مطمئن به اطلاعات مورد نیاز خود دسترسی پیدا کنند.

طراحی

در این بخش به توضیح جزئیات و روش طراحی جداول پایگاه داده، نرمال‌سازی، کلیدهای اصلی و خارجی، ساختار فایل‌ها و نمایه‌ها (ایندکس‌ها) می‌پردازیم.

طراحی جدول‌های پایگاه داده

فایل شامل ساختار و داده‌های پایگاه داده مربوط به سیستم مدیریت آپارتمان است. در ادامه، جداول موجود در پایگاه داده را توضیح داده و انواع ارتباطات میان آنها را مشخص می‌کنیم.

جداول و ساختار آنها:

1. admins (مدیران):

- id: شناسه منحصر به فرد مدیر (نوع: int)

- username: نام کاربری مدیر (نوع: varchar)

- password: رمز عبور مدیر (نوع: varchar)

2. apartments (آپارتمان‌ها):

- id: شناسه منحصر به فرد آپارتمان (نوع: int)

- apartment\_number: شماره آپارتمان (نوع: varchar)

- floor: طبقه آپارتمان (نوع: int)

- number\_of\_rooms: تعداد اتاق‌های آپارتمان (نوع: int)

- owner\_name: نام مالک آپارتمان (نوع: varchar)

- owner\_contact: شماره تماس مالک آپارتمان (نوع: varchar)

3. charges (هزینه‌ها):

- id: شناسه منحصر به فرد هزینه (نوع: int)

- charge\_type: نوع هزینه (enum: 'monthly', 'elevator', 'meter', 'parking', 'family')

- amount: مبلغ هزینه (نوع: decimal)

- date\_added: تاریخ اضافه شدن هزینه (نوع: date)

- resident\_id: شناسه ساکن مربوط به هزینه (نوع: int)

4. residents (ساکنان):

- id: شناسه منحصر به فرد ساکن (نوع: int)

- name: نام ساکن (نوع: varchar)

- apartment\_id: شناسه آپارتمان مربوط به ساکن (نوع: int)

- contact: شماره تماس ساکن (نوع: varchar)

ارتباطات بین جداول:

- رابطه بین جدول `apartments` و `residents`:

- هر آپارتمان می‌تواند چندین ساکن داشته باشد، اما هر ساکن متعلق به یک آپارتمان است. این رابطه به صورت یک به چند (One to Many) است.

- در جدول `residents` ستون `apartment\_id` به شناسه آپارتمان در جدول `apartments` اشاره دارد.

- رابطه بین جدول `residents` و `charges`:

- هر ساکن می‌تواند چندین هزینه داشته باشد، اما هر هزینه مربوط به یک ساکن است. این رابطه نیز به صورت یک به چند (One to Many) است.

- در جدول `charges` ستون `resident\_id` به شناسه ساکن در جدول `residents` اشاره دارد.

خلاصه:

- admins: اطلاعات مدیران.

- apartments: اطلاعات آپارتمان‌ها.

- residents: اطلاعات ساکنان، با اشاره به آپارتمان مربوطه.

- charges: اطلاعات هزینه‌ها، با اشاره به ساکن مربوطه.

ارتباطات اصلی بین جداول:

- apartments به residents: یک به چند.

- residents به charges: یک به چند.

کلید های خارجی به کار رفته :

برای مشخص کردن کلیدهای خارجی و توضیحات مربوط به آنها، باید به دقت ساختار جداول را بررسی کنیم و ببینیم کدام ستون‌ها به عنوان کلید خارجی تعریف شده‌اند و به کدام جدول‌ها اشاره می‌کنند.

1. جدول `residents`:

- apartment\_id:

- توضیحات: این ستون به شناسه آپارتمان (`id` در جدول `apartments`) اشاره دارد.

- کلید خارجی: این ستون یک کلید خارجی است که ارتباط بین ساکنان و آپارتمان‌ها را برقرار می‌کند. هر ساکن به یک آپارتمان تعلق دارد.

- تعریف:

sql

FOREIGN KEY (`apartment\_id`) REFERENCES `apartments`(`id`)

2. جدول `charges`:

- resident\_id:

- توضیحات: این ستون به شناسه ساکن (`id` در جدول `residents`) اشاره دارد.

- کلید خارجی: این ستون یک کلید خارجی است که ارتباط بین هزینه‌ها و ساکنان را برقرار می‌کند. هر هزینه به یک ساکن خاص مربوط می‌شود.

- تعریف:

sql

FOREIGN KEY (`resident\_id`) REFERENCES `residents`(`id`)

توضیحات کامل‌تر:

1. کلید خارجی در جدول `residents`:

- apartment\_id:

- نقش: ارتباط بین ساکنان و آپارتمان‌ها را برقرار می‌کند.

- هدف: مشخص می‌کند که هر ساکن در کدام آپارتمان ساکن است.

- جدول مرجع: `apartments`

- ستون مرجع: `id`

این ارتباط به پایگاه داده امکان می‌دهد تا از یکپارچگی داده‌ها اطمینان حاصل کند. به این صورت که نمی‌توان ساکنی را به آپارتمانی نسبت داد که در جدول `apartments` وجود ندارد.

2. کلید خارجی در جدول `charges`:

- resident\_id:

- نقش: ارتباط بین هزینه‌ها و ساکنان را برقرار می‌کند.

- هدف: مشخص می‌کند که هر هزینه مربوط به کدام ساکن است.

- جدول مرجع: `residents`

- ستون مرجع: `id`

این کلید خارجی نیز به پایگاه داده کمک می‌کند تا اطمینان حاصل کند که هر هزینه تنها به ساکنی نسبت داده می‌شود که در جدول `residents` موجود است.

جمع‌بندی:

- در جدول `residents`، ستون `apartment\_id` به عنوان کلید خارجی به جدول `apartments` اشاره می‌کند.

- در جدول `charges`, ستون `resident\_id` به عنوان کلید خارجی به جدول `residents` اشاره می‌کند.

این کلیدهای خارجی به پایگاه داده کمک می‌کنند تا روابط منطقی بین داده‌ها را حفظ کند و از ورود داده‌های ناهماهنگ جلوگیری شود.

نرم­افزار مدیریت پایگاه­ داده­ها

3-3- نرم‌افزار مدیریت پایگاه داده‌ها

نرم‌افزار مدیریت پایگاه داده‌ها (DBMS) ابزاری است که امکان ایجاد، مدیریت، و دسترسی به پایگاه‌های داده را فراهم می‌کند. در این پروژه برای مدیریت پایگاه داده‌های سیستم مدیریت شارژ ساختمان، از MySQL استفاده شده است. MySQL یک سیستم مدیریت پایگاه داده رابطه‌ای (RDBMS) است که به دلیل کارایی بالا، مقیاس‌پذیری و سهولت استفاده، یکی از محبوب‌ترین سیستم‌های مدیریت پایگاه داده در جهان به شمار می‌رود.

ویژگی‌های MySQL

1. کارایی بالا: MySQL بهینه‌سازی شده است تا برای بارهای کاری سنگین و پرتراکنش عملکرد مناسبی داشته باشد.

2. مقیاس‌پذیری: امکان گسترش پایگاه داده‌ها در MySQL به سادگی وجود دارد و می‌توان آن را برای حجم‌های مختلف داده‌ها استفاده کرد.

3. قابلیت اطمینان و امنیت: MySQL امکانات پیشرفته‌ای برای کنترل دسترسی، رمزگذاری داده‌ها و پشتیبان‌گیری ارائه می‌دهد.

4. سازگاری با استانداردها: MySQL از استانداردهای SQL پیروی می‌کند و با بسیاری از زبان‌های برنامه‌نویسی و سیستم‌عامل‌ها سازگار است.

5. جامعه کاربری گسترده: MySQL دارای یک جامعه کاربری بزرگ و فعال است که منابع زیادی برای آموزش و رفع مشکلات در اختیار کاربران قرار می‌دهد.

اجزاء و ابزارهای MySQL

1. MySQL Server: هسته اصلی MySQL که عملیات ذخیره و بازیابی داده‌ها را مدیریت می‌کند.

2. MySQL Workbench: یک ابزار گرافیکی که برای طراحی، توسعه و مدیریت پایگاه داده‌ها استفاده می‌شود.

3. MySQL Command Line Client: یک رابط خط فرمان که امکان اجرای دستورات SQL را فراهم می‌کند.

4. phpMyAdmin: یک ابزار مدیریتی تحت وب که برای مدیریت پایگاه داده‌های MySQL استفاده می‌شود. این ابزار امکان مدیریت جداول، اجرا و تست کوئری‌ها و پشتیبان‌گیری را به سادگی فراهم می‌کند.

مدیریت پایگاه داده در پروژه

در این پروژه، از MySQL برای مدیریت داده‌های مربوط به ساختمان، خانواده‌ها، امکانات ساختمان، رزروها، پرداخت‌ها و درخواست‌های تعمیرات استفاده شده است. با استفاده از ابزارهای مدیریتی MySQL مانند phpMyAdmin، کاربر قادر به انجام عملیات زیر خواهد بود:

- ایجاد و مدیریت جداول: امکان ایجاد، تغییر و حذف جداول پایگاه داده و تنظیمات مربوط به کلیدهای اصلی و خارجی.

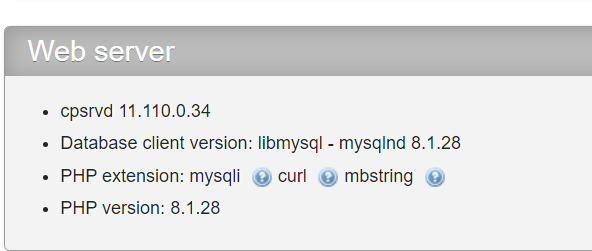
- پشتیبان‌گیری و بازیابی: ایجاد نسخه‌های پشتیبان از داده‌ها و بازیابی آن‌ها در صورت لزوم.

- کنترل دسترسی: تعریف کاربران و سطوح دسترسی مختلف برای مدیریت امنیت داده‌ها.

- اجرای کوئری‌ها: اجرای دستورات SQL برای مدیریت و جستجوی داده‌ها.

نتیجه‌گیری

انتخاب MySQL به عنوان سیستم مدیریت پایگاه داده برای این پروژه، به دلیل مزایای متعدد آن از جمله کارایی، امنیت و سهولت استفاده، یک انتخاب مناسب است. با استفاده از ابزارهای مدیریتی مرتبط، مدیریت و نگهداری پایگاه داده‌ها به صورت موثر و کارآمد امکان‌پذیر خواهد بود.



نتیجه‌گیری

در این فصل، به بررسی ساختار داده‌ها و بانک اطلاعاتی پروژه مدیریت شارژ ساختمان پرداختیم. ابتدا مقدمه‌ای بر مفاهیم اولیه و اهمیت استفاده از پایگاه داده در سیستم‌های مدیریت پرداخت ارائه شد. سپس جداول اصلی پایگاه داده را معرفی کردیم و ساختار آن‌ها را با جزئیات شرح دادیم. در بخش طراحی، نحوه طراحی جداول، نرمال‌سازی، تعریف کلیدهای اصلی و خارجی، و ایجاد نمایه‌ها مورد بحث قرار گرفت. همچنین، به بررسی کلیدهای خارجی و نقش آن‌ها در تضمین یکپارچگی داده‌ها پرداختیم. در نهایت، نرم‌افزار مدیریت پایگاه داده MySQL و ویژگی‌های کلیدی آن معرفی شد و نحوه استفاده از ابزارهای مرتبط برای مدیریت پایگاه داده‌ها توضیح داده شد.

با تکمیل این فصل، دیدگاه جامعی نسبت به ساختار داده‌ها و نحوه مدیریت آن‌ها در پروژه به دست آوردیم. این اطلاعات پایه‌ای ضروری برای مراحل بعدی پیاده‌سازی سیستم فراهم می‌کنند.

در فصل بعدی، به مرحله پیاده‌سازی سیستم خواهیم پرداخت. در این مرحله، به توسعه کدهای برنامه، پیاده‌سازی رابط‌های کاربری، و انجام تست‌های لازم برای اطمینان از عملکرد صحیح سیستم خواهیم پرداخت. پیاده‌سازی این سیستم گام مهمی در تحقق اهداف پروژه است و به ما امکان می‌دهد تا از تئوری به عمل برسیم و سیستم مدیریت شارژ ساختمان را به واقعیت تبدیل کنیم.

فصل چهار  
پیاده‌سازی

مقدمه

مقدمه فصل پیاده‌سازی

در این فصل، به جزئیات پیاده‌سازی سیستم مدیریت شارژ ساختمان خواهیم پرداخت. هدف اصلی این بخش، شرح فرآیندهای اجرایی و عملیاتی است که در طول توسعه سیستم انجام شده‌اند. این فصل شامل توضیحاتی در مورد ابزارها، نرم‌افزارها، چارچوب‌ها، زبان‌های برنامه‌نویسی و تنظیمات مورد استفاده در پروژه می‌باشد. همچنین، تصاویری از نمونه اجراهای سیستم و رخدادهای مختلف برای درک بهتر مراحل پیاده‌سازی ارائه خواهد شد.

ابتدا به معرفی ابزارها و محیط‌های توسعه‌ای که برای این پروژه انتخاب شده‌اند می‌پردازیم. در این پروژه از زبان‌های HTML، CSS، PHP و MySQL به عنوان پایه‌های اصلی توسعه استفاده شده است. HTML و CSS برای طراحی رابط کاربری، PHP برای منطق سرور و MySQL برای مدیریت پایگاه داده به کار گرفته شده‌اند. همچنین از نرم‌افزارهایی مانند phpMyAdmin برای مدیریت پایگاه داده و ابزارهای نسخه‌گذاری مانند Git برای مدیریت کدهای پروژه بهره برده‌ایم.

در بخش بعدی، جزئیات پیاده‌سازی شامل کدنویسی و توسعه ماژول‌های مختلف سیستم شرح داده خواهد شد. هر ماژول، شامل ثبت و مدیریت ساختمان، رزرو استفاده از امکانات، مدیریت پرداخت‌ها، شارژ ساختمان، تابلو اعلانات، درخواست‌های ساکنین و قرار ملاقات‌ها، به طور جداگانه مورد بررسی قرار گرفته و نحوه پیاده‌سازی آن‌ها توضیح داده خواهد شد.

در نهایت، تصاویری از نمونه اجراهای سیستم و رخدادهای مختلف، به منظور نمایش نحوه عملکرد سیستم و بررسی نتایج پیاده‌سازی، ارائه می‌شود. این تصاویر به درک بهتر و ملموس‌تر فرآیند پیاده‌سازی کمک کرده و نشان می‌دهند که چگونه سیستم توسعه یافته توانسته است نیازهای مطرح شده را برآورده سازد.

با تکمیل این فصل، دیدگاهی کامل از مراحل پیاده‌سازی و اجرای پروژه خواهیم داشت که می‌تواند به عنوان راهنمایی برای توسعه‌دهندگان و علاقه‌مندان به این حوزه مورد استفاده قرار گیرد.

این سیستم تحت وب با زبان برنامه نویسی و در یک سطح فقط مدیریت ساختمان نوشته شده است .

مروری کوتاه بر زبان برنامه نویسی بکند PHP :

PHP که مخفف "PHP: Hypertext Preprocessor" است، یک زبان برنامه‌نویسی سمت سرور است که به طور گسترده‌ای برای توسعه وب استفاده می‌شود. این زبان در ابتدا در سال 1994 توسط Rasmus Lerdorf ایجاد شد و به تدریج به یکی از پرکاربردترین زبان‌های برنامه‌نویسی وب تبدیل شده است. PHP به دلیل سادگی، انعطاف‌پذیری و کارایی بالا، مورد توجه توسعه‌دهندگان وب قرار گرفته است.

ویژگی‌های اصلی PHP

1. توسعه سریع و آسان:

- PHP به گونه‌ای طراحی شده که یادگیری و استفاده از آن آسان باشد. این ویژگی به توسعه‌دهندگان امکان می‌دهد تا به سرعت وب‌سایت‌ها و برنامه‌های کاربردی وب را ایجاد کنند.

2. پشتیبانی گسترده از پایگاه‌های داده:

- PHP به طور مستقیم با بسیاری از سیستم‌های مدیریت پایگاه داده (DBMS) مانند MySQL، PostgreSQL، Oracle و SQLite تعامل دارد. این ویژگی به توسعه‌دهندگان این امکان را می‌دهد تا به راحتی داده‌ها را ذخیره و بازیابی کنند.

3. یکپارچگی با HTML:

- PHP به راحتی با HTML ترکیب می‌شود، که این امکان را می‌دهد تا صفحات وب پویا ایجاد شوند. کد PHP می‌تواند در میان کد HTML قرار گیرد و در زمان اجرای صفحه توسط سرور پردازش شود.

4. قابلیت گسترش:

- PHP دارای کتابخانه‌های گسترده‌ای است که توسعه‌دهندگان می‌توانند از آن‌ها برای افزودن قابلیت‌های مختلف به برنامه‌های خود استفاده کنند. همچنین، افزونه‌ها و ماژول‌های متعددی وجود دارند که می‌توانند برای گسترش عملکرد PHP به کار گرفته شوند.

5. جامعه فعال و منابع فراوان:

- PHP دارای یک جامعه بزرگ و فعال از توسعه‌دهندگان است. این جامعه منابع فراوانی از جمله مستندات، انجمن‌ها، و آموزش‌های آنلاین را ارائه می‌دهد که می‌تواند به توسعه‌دهندگان در حل مشکلات و بهبود مهارت‌هایشان کمک کند.

کاربردهای PHP

- توسعه وب‌سایت‌ها و برنامه‌های کاربردی وب:

PHP به طور گسترده برای ساخت وب‌سایت‌های پویا و برنامه‌های کاربردی وب مورد استفاده قرار می‌گیرد. از نمونه‌های معروفی که با PHP توسعه یافته‌اند می‌توان به فیسبوک، وردپرس و ویکی‌پدیا اشاره کرد.

- مدیریت پایگاه‌های داده:

PHP با توانایی برقراری ارتباط مستقیم با پایگاه‌های داده، به توسعه‌دهندگان این امکان را می‌دهد تا به راحتی داده‌ها را مدیریت کنند.

- توسعه API:

PHP می‌تواند برای ایجاد و مدیریت APIهای وب استفاده شود که به برنامه‌های مختلف اجازه می‌دهد با یکدیگر تعامل داشته باشند.

در نهایت، PHP به عنوان یک زبان قدرتمند و همه‌کاره برای توسعه وب، به توسعه‌دهندگان این امکان را می‌دهد تا به سرعت و با کارایی بالا، برنامه‌های کاربردی پیچیده و متنوعی را ایجاد و مدیریت کنند.

کدنویسی

کد نویسی

add-apartment.php

<?php

require\_once "config.php";

if (!isset($\_SESSION['admin\_username'])) {

    header("Location: login.php");

    exit();

}

if ($\_SERVER["REQUEST\_METHOD"] == "POST") {

    $apartment\_number = $\_POST['apartment\_number'];

    $floor = $\_POST['floor'];

    $number\_of\_rooms = $\_POST['number\_of\_rooms'];

    $owner\_name = $\_POST['owner\_name'];

    $owner\_contact = $\_POST['owner\_contact'];

    $query = "INSERT INTO apartments (apartment\_number, floor, number\_of\_rooms, owner\_name, owner\_contact) VALUES ('$apartment\_number', '$floor', '$number\_of\_rooms', '$owner\_name', '$owner\_contact')";

    if (mysqli\_query($conn, $query)) {

        $addResponse = ['success', "افزودن آپارتمان با موفقیت انجام شد"];

    } else {

        $addResponse = ['danger', "Error: " . mysqli\_error($conn)];

    }

}

$adminSidebarMenuItems['add-apartment']['is-active'] = true;

?>

کد PHP برای مدیریت افزودن آپارتمان جدید

این کد PHP برای افزودن آپارتمان جدید به پایگاه داده و نمایش پیغام‌های موفقیت یا خطا استفاده می‌شود. در اینجا به شرح دقیق هر بخش می‌پردازیم.

توضیحات کد

1. اتصال به فایل پیکربندی (config.php):

require\_once "config.php";

- این خط کد فایل `config.php` را وارد می‌کند که شامل تنظیمات پایگاه داده و سایر تنظیمات ضروری است.

2. بررسی ورود مدیر:

if (!isset($\_SESSION['admin\_username'])) {

header("Location: login.php");

exit();

}

- این کد بررسی می‌کند که آیا مدیر وارد شده است یا خیر. اگر مدیر وارد نشده باشد، به صفحه `login.php` منتقل می‌شود.

3. بررسی درخواست POST:

if ($\_SERVER["REQUEST\_METHOD"] == "POST") {

$apartment\_number = $\_POST['apartment\_number'];

$floor = $\_POST['floor'];

$number\_of\_rooms = $\_POST['number\_of\_rooms'];

$owner\_name = $\_POST['owner\_name'];

$owner\_contact = $\_POST['owner\_contact'];

- اگر درخواست به روش `POST` ارسال شده باشد، اطلاعات آپارتمان از فرم دریافت می‌شود.

4. ایجاد و اجرای کوئری SQL:

$query = "INSERT INTO apartments (apartment\_number, floor, number\_of\_rooms, owner\_name, owner\_contact) VALUES ('$apartment\_number', '$floor', '$number\_of\_rooms', '$owner\_name', '$owner\_contact')";

- کوئری SQL برای وارد کردن اطلاعات آپارتمان جدید در جدول `apartments` ساخته می‌شود.

5. اجرای کوئری و بررسی نتیجه:

if (mysqli\_query($conn, $query)) {

$addResponse = ['success', "افزودن آپارتمان با موفقیت انجام شد"];

} else {

$addResponse = ['danger', "Error: " . mysqli\_error($conn)];

}

- کوئری اجرا می‌شود و نتیجه بررسی می‌گردد. اگر کوئری موفقیت‌آمیز باشد، پیامی مبنی بر موفقیت آمیز بودن اضافه کردن آپارتمان نمایش داده می‌شود و در غیر این صورت خطای مربوطه نمایش داده می‌شود.

6. فعال کردن آیتم منوی افزودن آپارتمان:

$adminSidebarMenuItems['add-apartment']['is-active'] = true;

- این کد آیتم منوی افزودن آپارتمان را به عنوان آیتم فعال در منوی جانبی تنظیم می‌کند.

add-late-penalty.php

<?php

require\_once "config.php";

if (!isset($\_SESSION['admin\_username'])) {

    header("Location: login.php");

    exit();

}

$query = "SELECT id, name FROM residents";

$result = mysqli\_query($conn, $query);

$residents = mysqli\_fetch\_all($result, MYSQLI\_ASSOC);

if ($\_SERVER["REQUEST\_METHOD"] == "POST") {

    $resident\_id = $\_POST['resident\_id'];

    $amount = $\_POST['amount'];

    $query = "INSERT INTO penalties (resident\_id, amount, date\_added) VALUES ('$resident\_id', '$amount', NOW())";

    if (mysqli\_query($conn, $query)) {

        $addResponse = ['success', "افزودن جریمه با موفقیت انجام شد"];

    } else {

        $addResponse = ['danger', "Error: " . mysqli\_error($conn)];

    }

}

$adminSidebarMenuItems['add-late-penalty']['is-active'] = true;

?>

توضیحات کد PHP برای افزودن جریمه جدید

این کد PHP برای افزودن جریمه جدید به ساکنان مجتمع مسکونی استفاده می‌شود و شامل بخش‌هایی برای دریافت اطلاعات ساکنان، دریافت اطلاعات فرم و اجرای کوئری SQL می‌باشد. در اینجا به شرح دقیق هر بخش می‌پردازیم.

توضیحات کد

1. اتصال به فایل پیکربندی (config.php):

require\_once "config.php";

- این خط کد فایل `config.php` را وارد می‌کند که شامل تنظیمات پایگاه داده و سایر تنظیمات ضروری است.

2. بررسی ورود مدیر:

if (!isset($\_SESSION['admin\_username'])) {

header("Location: login.php");

exit();

}

- این کد بررسی می‌کند که آیا مدیر وارد شده است یا خیر. اگر مدیر وارد نشده باشد، به صفحه `login.php` منتقل می‌شود.

3. دریافت اطلاعات ساکنان:

$query = "SELECT id, name FROM residents";

$result = mysqli\_query($conn, $query);

$residents = mysqli\_fetch\_all($result, MYSQLI\_ASSOC);

- این کد یک کوئری SQL برای دریافت اطلاعات ساکنان اجرا می‌کند و نتایج را در آرایه `$residents` ذخیره می‌کند. این اطلاعات برای نمایش در فرم استفاده خواهد شد.

4. بررسی درخواست POST و دریافت داده‌های فرم:

if ($\_SERVER["REQUEST\_METHOD"] == "POST") {

$resident\_id = $\_POST['resident\_id'];

$amount = $\_POST['amount'];

- اگر درخواست به روش `POST` ارسال شده باشد، اطلاعات مربوط به جریمه از فرم دریافت می‌شود.

5. ایجاد و اجرای کوئری SQL برای افزودن جریمه:

$query = "INSERT INTO penalties (resident\_id, amount, date\_added) VALUES ('$resident\_id', '$amount', NOW())";

if (mysqli\_query($conn, $query)) {

$addResponse = ['success', "افزودن جریمه با موفقیت انجام شد"];

} else {

$addResponse = ['danger', "Error: " . mysqli\_error($conn)];

}

- کوئری SQL برای وارد کردن اطلاعات جریمه جدید در جدول `penalties` ساخته می‌شود. در صورت موفقیت‌آمیز بودن کوئری، پیام موفقیت و در غیر این صورت، پیام خطا نمایش داده می‌شود.

6. فعال کردن آیتم منوی افزودن جریمه:

$adminSidebarMenuItems['add-late-penalty']['is-active'] = true;

- این کد آیتم منوی افزودن جریمه را به عنوان آیتم فعال در منوی جانبی تنظیم می‌کند.

admin-dashboard.php

<?php

require\_once "config.php";

if (!isset($\_SESSION['admin\_username'])) {

    header("Location: login.php");

    exit();

}

$comments = [

    ['name' => 'آرمان', 'img' => 'user3', 'text' => 'استفاده از این نرم افزار برای من کار را آسان کرده. هر کسی که ساختمانی را اداره می‌کند باید این نرم افزار را داشته باشد. 🏬', 'time' => '2 دقیقه قبل',],

    ['name' => 'مریم', 'img' => 'user1', 'text' => 'رابط کاربری خیلی راحتی داره و واقعا کمک زیادی به من کرده. تشکر! 👌', 'time' => '2 دقیقه قبل',],

    ['name' => 'کامران', 'img' => 'user', 'text' => 'عالیه! از زمانی که از این نرم افزار استفاده می‌کنم، بهتر شده. 🏢', 'time' => '2 دقیقه قبل',],

    ['name' => 'سارا', 'img' => 'user4', 'text' => 'واقعا از این اپلیکیشن لذت می‌برم. به همه توصیه می‌کنم! 😊', 'time' => '2 دقیقه قبل',],

    ['name' => 'علی', 'img' => 'user2', 'text' => 'با استفاده از این نرم افزار، مدیریت شارژ ساختمان به راحتی انجام می‌شود. 👍', 'time' => '2 دقیقه قبل',],

    ['name' => 'نیلوفر', 'img' => 'user5', 'text' => 'این نرم افزار فوق العاده است! 🌟', 'time' => '2 دقیقه قبل',]

];

$notifications = [

    ['color' => 'info', 'text' => 'خوش آمدید! از اکنون می‌توانید شارژ خود را به راحتی مدیریت کنید.', 'time' => '2 دقیقه قبل', 'icon' => 'envelope-open',],

    ['color' => 'success', 'text' => 'آخرین تخفیفات ویژه برای شارژهای بالاتر از 100 واحد را چک کنید!', 'time' => '2 دقیقه قبل', 'icon' => 'envelope-open',],

    ['color' => 'warning', 'text' => ' تغییرات جدید در روش‌های پرداخت شارژ به زودی در دسترس خواهد بود.', 'time' => '2 دقیقه قبل', 'icon' => 'envelope-open',],

    ['color' => '', 'text' => 'آموزش کامل استفاده از امکانات جدید به آسانی با یک کلیک!', 'time' => '2 دقیقه قبل', 'icon' => 'envelope-open',],

    ['color' => 'success', 'text' => 'برنامه‌ی وفاداری: با هر شارژ، امکانات اختصاصی بیشتری را کسب کنید.', 'time' => '2 دقیقه قبل', 'icon' => 'envelope-open',],

    ['color' => 'danger', 'text' => 'مشکلات رایج و راه‌حل‌های آن‌ها: چگونه می‌توانم از بهترین راهکارها برای مدیریت شارژ استفاده کنم؟', 'time' => '2 دقیقه قبل', 'icon' => 'envelope-open',],

];

$adminSidebarMenuItems['admin-dashboard']['is-active'] = true;

?>

توضیحات کد PHP برای داشبورد مدیر

این کد PHP شامل بخش‌های مختلفی است که برای مدیریت نمایش نظرات کاربران، اعلان‌ها و منوی جانبی در داشبورد مدیر استفاده می‌شود. در اینجا به شرح دقیق هر بخش می‌پردازیم.

توضیحات کد

1. اتصال به فایل پیکربندی (config.php):

require\_once "config.php";

- این خط کد فایل `config.php` را وارد می‌کند که شامل تنظیمات پایگاه داده و سایر تنظیمات ضروری است.

2. بررسی ورود مدیر:

if (!isset($\_SESSION['admin\_username'])) {

header("Location: login.php");

exit();

}

- این کد بررسی می‌کند که آیا مدیر وارد شده است یا خیر. اگر مدیر وارد نشده باشد، به صفحه `login.php` منتقل می‌شود.

3. تعریف نظرات کاربران:

$comments = [

['name' => 'آرمان', 'img' => 'user3', 'text' => 'استفاده از این نرم افزار برای من کار را آسان کرده. هر کسی که ساختمانی را اداره می‌کند باید این نرم افزار را داشته باشد. 🏬', 'time' => '2 دقیقه قبل',],

['name' => 'مریم', 'img' => 'user1', 'text' => 'رابط کاربری خیلی راحتی داره و واقعا کمک زیادی به من کرده. تشکر! 👌', 'time' => '2 دقیقه قبل',],

['name' => 'کامران', 'img' => 'user', 'text' => 'عالیه! از زمانی که از این نرم افزار استفاده می‌کنم، بهتر شده. 🏢', 'time' => '2 دقیقه قبل',],

['name' => 'سارا', 'img' => 'user4', 'text' => 'واقعا از این اپلیکیشن لذت می‌برم. به همه توصیه می‌کنم! 😊', 'time' => '2 دقیقه قبل',],

['name' => 'علی', 'img' => 'user2', 'text' => 'با استفاده از این نرم افزار، مدیریت شارژ ساختمان به راحتی انجام می‌شود. 👍', 'time' => '2 دقیقه قبل',],

['name' => 'نیلوفر', 'img' => 'user5', 'text' => 'این نرم افزار فوق العاده است! 🌟', 'time' => '2 دقیقه قبل',]

];

- یک آرایه از نظرات کاربران تعریف شده است که شامل نام کاربر، تصویر کاربر، متن نظر و زمان ارسال نظر می‌باشد.

4. تعریف اعلان‌ها:

$notifications = [

['color' => 'info', 'text' => 'خوش آمدید! از اکنون می‌توانید شارژ خود را به راحتی مدیریت کنید.', 'time' => '2 دقیقه قبل', 'icon' => 'envelope-open',],

['color' => 'success', 'text' => 'آخرین تخفیفات ویژه برای شارژهای بالاتر از 100 واحد را چک کنید!', 'time' => '2 دقیقه قبل', 'icon' => 'envelope-open',],

['color' => 'warning', 'text' => ' تغییرات جدید در روش‌های پرداخت شارژ به زودی در دسترس خواهد بود.', 'time' => '2 دقیقه قبل', 'icon' => 'envelope-open',],

['color' => '', 'text' => 'آموزش کامل استفاده از امکانات جدید به آسانی با یک کلیک!', 'time' => '2 دقیقه قبل', 'icon' => 'envelope-open',],

['color' => 'success', 'text' => 'برنامه‌ی وفاداری: با هر شارژ، امکانات اختصاصی بیشتری را کسب کنید.', 'time' => '2 دقیقه قبل', 'icon' => 'envelope-open',],

['color' => 'danger', 'text' => 'مشکلات رایج و راه‌حل‌های آن‌ها: چگونه می‌توانم از بهترین راهکارها برای مدیریت شارژ استفاده کنم؟', 'time' => '2 دقیقه قبل', 'icon' => 'envelope-open',],

];

- یک آرایه از اعلان‌ها تعریف شده است که شامل رنگ اعلان، متن اعلان، زمان ارسال اعلان و آیکون مربوطه می‌باشد.

5. فعال کردن آیتم منوی داشبورد مدیر:

$adminSidebarMenuItems['admin-dashboard']['is-active'] = true;

- این کد آیتم منوی داشبورد مدیر را به عنوان آیتم فعال در منوی جانبی تنظیم می‌کند.

calculation-of-late-penalty.php

<?php

require\_once "config.php";

if (!isset($\_SESSION['admin\_username'])) {

    header("Location: login.php");

    exit();

}

$query = "SELECT id, name FROM residents";

$result = mysqli\_query($conn, $query);

$residents = mysqli\_fetch\_all($result, MYSQLI\_ASSOC);

if ($\_SERVER["REQUEST\_METHOD"] == "POST") {

    $resident\_id = $\_POST['resident\_id'];

    $amount\_due = $\_POST['amount\_due'];

    $due\_date = $\_POST['due\_date'];

    $current\_date = date('Y-m-d');

    $penalty\_amount = 0;

    if ($current\_date > $due\_date) {

        $diff = strtotime($current\_date) - strtotime($due\_date);

        $days\_late = ceil($diff / (60 \* 60 \* 24));

        $penalty\_amount = $days\_late \* 1500; // Example: $1.5 penalty per day

    } else {

        $penalty\_amount = 0;

    }

    $selected\_resident = '';

    foreach ($residents as $resident) {

        if ($resident['id'] == $resident\_id) {

            $selected\_resident = $resident['name'];

            break;

        }

    }

    $penaltyAmount = "جریمه دیرکرد " . $selected\_resident . " برابر: " . number\_format($penalty\_amount) . " تومان است.";

}

$adminSidebarMenuItems['calculation-of-late-penalty']['is-active'] = true;

?>

توضیحات کد PHP برای محاسبه جریمه دیرکرد

این کد PHP برای محاسبه جریمه دیرکرد پرداخت شارژ ساختمان طراحی شده است. در اینجا به صورت جزئی هر بخش از کد توضیح داده می‌شود.

توضیحات کد

1. اتصال به فایل پیکربندی (config.php):

require\_once "config.php";

- این خط کد فایل `config.php` را وارد می‌کند که شامل تنظیمات پایگاه داده و سایر تنظیمات ضروری است.

2. بررسی ورود مدیر:

if (!isset($\_SESSION['admin\_username'])) {

header("Location: login.php");

exit();

}

- این کد بررسی می‌کند که آیا مدیر وارد شده است یا خیر. اگر مدیر وارد نشده باشد، به صفحه `login.php` منتقل می‌شود.

3. دریافت اطلاعات ساکنان:

$query = "SELECT id, name FROM residents";

$result = mysqli\_query($conn, $query);

$residents = mysqli\_fetch\_all($result, MYSQLI\_ASSOC);

- این بخش یک کوئری SQL برای دریافت شناسه و نام تمامی ساکنان از جدول `residents` ارسال می‌کند.

- نتیجه‌ی کوئری در متغیر `$result` ذخیره می‌شود و سپس به یک آرایه‌ی انجمنی تبدیل می‌شود که در متغیر `$residents` قرار می‌گیرد.

4. پردازش فرم ارسالی:

if ($\_SERVER["REQUEST\_METHOD"] == "POST") {

$resident\_id = $\_POST['resident\_id'];

$amount\_due = $\_POST['amount\_due'];

$due\_date = $\_POST['due\_date'];

$current\_date = date('Y-m-d');

$penalty\_amount = 0;

if ($current\_date > $due\_date) {

$diff = strtotime($current\_date) - strtotime($due\_date);

$days\_late = ceil($diff / (60 \* 60 \* 24));

$penalty\_amount = $days\_late \* 1500; // Example: $1.5 penalty per day

} else {

$penalty\_amount = 0;

}

$selected\_resident = '';

foreach ($residents as $resident) {

if ($resident['id'] == $resident\_id) {

$selected\_resident = $resident['name'];

break;

}

}

$penaltyAmount = "جریمه دیرکرد " . $selected\_resident . " برابر: " . number\_format($penalty\_amount) . " تومان است.";

}

- این بخش چک می‌کند که آیا فرم ارسال شده است یا خیر (یعنی آیا درخواست POST است).

- اگر فرم ارسال شده باشد، اطلاعاتی که کاربر در فرم وارد کرده (شناسه ساکن، مبلغ بدهی و تاریخ سررسید) از طریق `$\_POST` دریافت می‌شود.

- تاریخ جاری نیز دریافت می‌شود.

- سپس محاسبه جریمه انجام می‌شود. اگر تاریخ جاری بعد از تاریخ سررسید باشد، تعداد روزهای تاخیر محاسبه و جریمه بر اساس 1500 تومان برای هر روز تاخیر محاسبه می‌شود.

- در ادامه، نام ساکن مورد نظر از آرایه‌ی `residents` جستجو و یافت می‌شود.

- در نهایت، یک پیام شامل مبلغ جریمه به زبان فارسی ساخته می‌شود.

5. فعال کردن آیتم منوی محاسبه جریمه دیرکرد:

$adminSidebarMenuItems['calculation-of-late-penalty']['is-active'] = true;

- این کد آیتم منوی محاسبه جریمه دیرکرد را به عنوان آیتم فعال در منوی جانبی تنظیم می‌کند.

نتیجه‌گیری

این کد برای محاسبه جریمه دیرکرد پرداخت شارژ ساختمان طراحی شده است. با استفاده از این کد، مدیر می‌تواند با وارد کردن شناسه ساکن، مبلغ بدهی و تاریخ سررسید، جریمه تاخیر را محاسبه کرده و به ساکن اطلاع دهد.

config.php

<?php

session\_start();

$servername = "localhost";

$username = "root"; // Replace with your database username

$password = ""; // Replace with your database password

$dbname = "kamizi"; // Replace with your database name

// Create connection

$conn = mysqli\_connect($servername, $username, $password, $dbname);

// Check connection

if (!$conn) {

    die("Connection failed: " . mysqli\_connect\_error());

}

// Set UTF-8 encoding

mysqli\_set\_charset($conn, "utf8");

// Optionally, set collation (if needed)

mysqli\_query($conn, "SET collation\_connection = 'utf8\_general\_ci'");

define('SITE\_TITLE', 'مدیریت شارژ ساختمان کمیزی');

$sidebarMenuItems = [

    'resident-dashboard' => ['text' => "داشبورد", 'icon' => 'pie-chart', 'is-active' => false],

    'pay-elevator-charge' => ['text' => "پرداخت شارژ آسانسور", 'icon' => 'grip-vertical', 'is-active' => false],

    'pay-parking-charges' => ['text' => "پرداخت شارژ پارکینگ", 'icon' => 'p-circle', 'is-active' => false],

    'pay-charge-monthly-apartment' => ['text' => "پرداخت شارژ ماهانه", 'icon' => 'house', 'is-active' => false],

    'logout' => ['text' => "خروج", 'icon' => 'box-arrow-left', 'is-active' => false],

];

$adminSidebarMenuItems = [

    'admin-dashboard' => ['text' => "داشبورد", 'icon' => 'pie-chart', 'is-active' => false],

    'add-apartment' => ['text' => "اضافه کردن آپارتمان", 'icon' => 'plus-circle', 'is-active' => false],

    'view-apartment' => ['text' => "مشاهده آپارتمان‌ها", 'icon' => 'eye', 'is-active' => false],

    'view-resident' => ['text' => "مشاهده ساکنین", 'icon' => 'list', 'is-active' => false],

    'add-late-penalty' => ['text' => "اضافه کردن جریمه دیرکرد", 'icon' => 'plus-circle', 'is-active' => false],

    'calculation-of-late-penalty' => ['text' => "محاسبه جریمه دیرکرد", 'icon' => 'calculator', 'is-active' => false],

    'view-late-penalty' => ['text' => "مشاهده جریمه‌های دیرکرد", 'icon' => 'list', 'is-active' => false],

    // 'add-resident' => ['text' => "اضافه کردن ساکن", 'icon' => 'plus-circle', 'is-active' => false],

    'view-charges' => ['text' => "مشاهده شارژها", 'icon' => 'receipt', 'is-active' => false],

    'logout' => ['text' => "خروج", 'icon' => 'box-arrow-left', 'is-active' => false],

];

require\_once 'jdf.php';

توضیحات کد برای مدیریت شارژ ساختمان کمیزی

این کد برای مدیریت شارژ ساختمان کمیزی طراحی شده است. در ادامه به توضیح بخش‌های مختلف کد می‌پردازیم.

توضیحات کد

1. شروع جلسه (Session)

session\_start();

- این خط کد جلسه‌ی PHP را شروع می‌کند که برای مدیریت اطلاعات ورود کاربران مورد استفاده قرار می‌گیرد.

2. تنظیمات اتصال به پایگاه داده

$servername = "localhost";

$username = "root"; // Replace with your database username

$password = ""; // Replace with your database password

$dbname = "kamizi"; // Replace with your database name

// Create connection

$conn = mysqli\_connect($servername, $username, $password, $dbname);

// Check connection

if (!$conn) {

die("Connection failed: " . mysqli\_connect\_error());

}

- این بخش برای اتصال به پایگاه داده‌ی MySQL استفاده می‌شود.

- اطلاعات اتصال شامل نام سرور، نام کاربری، رمز عبور و نام پایگاه داده تعریف شده‌اند.

- سپس اتصال با استفاده از `mysqli\_connect` ایجاد می‌شود و اگر اتصال موفقیت‌آمیز نباشد، پیام خطا نمایش داده می‌شود.

3. تنظیمات کدگذاری و اتصال

// Set UTF-8 encoding

mysqli\_set\_charset($conn, "utf8");

// Optionally, set collation (if needed)

mysqli\_query($conn, "SET collation\_connection = 'utf8\_general\_ci'");

- این بخش برای تنظیم کدگذاری UTF-8 جهت پشتیبانی از زبان فارسی استفاده می‌شود.

- تنظیم collation به صورت اختیاری برای مدیریت ترتیب بندی کاراکترها.

4. تعریف ثابت‌ها و منوها

define('SITE\_TITLE', 'مدیریت شارژ ساختمان کمیزی');

$sidebarMenuItems = [

'resident-dashboard' => ['text' => "داشبورد", 'icon' => 'pie-chart', 'is-active' => false],

'pay-elevator-charge' => ['text' => "پرداخت شارژ آسانسور", 'icon' => 'grip-vertical', 'is-active' => false],

'pay-parking-charges' => ['text' => "پرداخت شارژ پارکینگ", 'icon' => 'p-circle', 'is-active' => false],

'pay-charge-monthly-apartment' => ['text' => "پرداخت شارژ ماهانه", 'icon' => 'house', 'is-active' => false],

'logout' => ['text' => "خروج", 'icon' => 'box-arrow-left', 'is-active' => false],

];

$adminSidebarMenuItems = [

'admin-dashboard' => ['text' => "داشبورد", 'icon' => 'pie-chart', 'is-active' => false],

'add-apartment' => ['text' => "اضافه کردن آپارتمان", 'icon' => 'plus-circle', 'is-active' => false],

'view-apartment' => ['text' => "مشاهده آپارتمان‌ها", 'icon' => 'eye', 'is-active' => false],

'view-resident' => ['text' => "مشاهده ساکنین", 'icon' => 'list', 'is-active' => false],

'add-late-penalty' => ['text' => "اضافه کردن جریمه دیرکرد", 'icon' => 'plus-circle', 'is-active' => false],

'calculation-of-late-penalty' => ['text' => "محاسبه جریمه دیرکرد", 'icon' => 'calculator', 'is-active' => false],

'view-late-penalty' => ['text' => "مشاهده جریمه‌های دیرکرد", 'icon' => 'list', 'is-active' => false],

// 'add-resident' => ['text' => "اضافه کردن ساکن", 'icon' => 'plus-circle', 'is-active' => false],

'view-charges' => ['text' => "مشاهده شارژها", 'icon' => 'receipt', 'is-active' => false],

'logout' => ['text' => "خروج", 'icon' => 'box-arrow-left', 'is-active' => false],

];

- ثابت `SITE\_TITLE` برای تعریف عنوان سایت استفاده می‌شود.

- دو آرایه‌ی `$sidebarMenuItems` و `$adminSidebarMenuItems` برای تعریف آیتم‌های منوی کاربر و مدیر تعریف شده‌اند.

- هر آیتم منو شامل متن، آیکون و وضعیت فعال/غیرفعال بودن است.

5. وارد کردن فایل jdf.php

require\_once 'jdf.php';

- این خط کد فایل `jdf.php` را وارد می‌کند که احتمالاً برای مدیریت تاریخ جلالی (هجری شمسی) استفاده می‌شود.

نتیجه‌گیری

این کد برای مدیریت سیستم شارژ ساختمان طراحی شده است و شامل بخش‌هایی برای اتصال به پایگاه داده، تنظیمات کدگذاری و تعریف منوهای کاربر و مدیر است. همچنین بررسی ورود مدیر و وارد کردن فایل‌های مورد نیاز نیز در این کد انجام می‌شود.

edit-resident.php

<?php

require\_once 'config.php';

if (!isset($\_SESSION['admin\_username'])) {

    header("Location: login.php");

    exit();

} elseif (!isset($\_GET['rId'])) {

    header("Location: view-resident.php?action=danger&text=شناسه ساکن برای ویرایش یافت نشد.");

    exit();

}

if ($\_SERVER["REQUEST\_METHOD"] == "POST") {

    $resident\_id = $\_POST['resident\_id'];

    $name = $\_POST['name'];

    $apartment\_id = $\_POST['apartment\_id'];

    $contact = $\_POST['contact'];

    $is\_primary\_resident = isset($\_POST['is\_primary\_resident']) ? 1 : 0;

    $password = (empty($\_POST['password'])) ? '' : ", password='$\_POST[password]'";

    $query = "UPDATE residents SET name = '$name', apartment\_id = '$apartment\_id', contact = '$contact', is\_primary\_resident = '$is\_primary\_resident'$password WHERE id = '$resident\_id'";

    if (mysqli\_query($conn, $query)) {

        header("Location: view-resident.php?action=success&text=اطلاعات $name با موفقیت بروزرسانی شد.");

    } else {

        header("Location: view-resident.php?action=danger&text=" . mysqli\_error($conn));

    }

}

$resident\_id = $\_GET['rId'];

$query = "SELECT \* FROM residents WHERE id = '$resident\_id'";

$result = mysqli\_query($conn, $query);

$resident = mysqli\_fetch\_assoc($result);

$query\_apartments = "SELECT id, apartment\_number FROM apartments";

$result\_apartments = mysqli\_query($conn, $query\_apartments);

$apartment\_options = '';

while ($row = mysqli\_fetch\_assoc($result\_apartments)) {

    $apCheck = ($resident['apartment\_id'] == $row['id']) ? 'selected' : '';

    $apartment\_options .= '<option value="' . $row['id'] . '" ' . $apCheck . '>' . $row['apartment\_number'] . '</option>';

}

?>

توضیحات کد برای ویرایش اطلاعات ساکنین

این کد برای ویرایش اطلاعات ساکنین در سیستم مدیریت شارژ ساختمان کمیزی طراحی شده است. در ادامه به توضیح بخش‌های مختلف کد می‌پردازیم.

توضیحات کد

1. وارد کردن فایل‌های پیکربندی و بررسی‌های اولیه

require\_once 'config.php';

if (!isset($\_SESSION['admin\_username'])) {

header("Location: login.php");

exit();

} elseif (!isset($\_GET['rId'])) {

header("Location: view-resident.php?action=danger&text=شناسه ساکن برای ویرایش یافت نشد.");

exit();

}

- در ابتدا فایل پیکربندی `config.php` وارد می‌شود.

- سپس بررسی می‌شود که آیا کاربر مدیر وارد سیستم شده است یا خیر. اگر وارد نشده باشد، به صفحه‌ی ورود هدایت می‌شود.

- همچنین بررسی می‌شود که آیا شناسه‌ی ساکن (`rId`) در پارامترهای GET موجود است یا خیر. اگر موجود نباشد، به صفحه‌ی مشاهده ساکنین هدایت می‌شود و پیام خطا نمایش داده می‌شود.

2. پردازش فرم ویرایش ساکن

if ($\_SERVER["REQUEST\_METHOD"] == "POST") {

$resident\_id = $\_POST['resident\_id'];

$name = $\_POST['name'];

$apartment\_id = $\_POST['apartment\_id'];

$contact = $\_POST['contact'];

$is\_primary\_resident = isset($\_POST['is\_primary\_resident']) ? 1 : 0;

$password = (empty($\_POST['password'])) ? '' : ", password='$\_POST[password]'";

$query = "UPDATE residents SET name = '$name', apartment\_id = '$apartment\_id', contact = '$contact', is\_primary\_resident = '$is\_primary\_resident'$password WHERE id = '$resident\_id'";

if (mysqli\_query($conn, $query)) {

header("Location: view-resident.php?action=success&text=اطلاعات $name با موفقیت بروزرسانی شد.");

} else {

header("Location: view-resident.php?action=danger&text=" . mysqli\_error($conn));

}

}

- اگر درخواست به صورت POST ارسال شده باشد، اطلاعات فرم دریافت می‌شود.

- اطلاعات ساکن از جمله نام، شناسه آپارتمان، تماس، وضعیت ساکن اصلی بودن و رمز عبور دریافت می‌شود.

- اگر رمز عبور خالی باشد، هیچ تغییری در رمز عبور ایجاد نمی‌شود.

- دستور SQL برای به‌روزرسانی اطلاعات ساکن ایجاد می‌شود و در صورت موفقیت‌آمیز بودن، کاربر به صفحه‌ی مشاهده ساکنین با پیام موفقیت هدایت می‌شود و در صورت خطا، پیام خطا نمایش داده می‌شود.

3. دریافت اطلاعات ساکن برای ویرایش

$resident\_id = $\_GET['rId'];

$query = "SELECT \* FROM residents WHERE id = '$resident\_id'";

$result = mysqli\_query($conn, $query);

$resident = mysqli\_fetch\_assoc($result);

- شناسه‌ی ساکن از پارامتر GET دریافت می‌شود.

- اطلاعات ساکن از پایگاه داده با استفاده از شناسه‌ی ساکن دریافت می‌شود.

4. دریافت لیست آپارتمان‌ها

$query\_apartments = "SELECT id, apartment\_number FROM apartments";

$result\_apartments = mysqli\_query($conn, $query\_apartments);

$apartment\_options = '';

while ($row = mysqli\_fetch\_assoc($result\_apartments)) {

$apCheck = ($resident['apartment\_id'] == $row['id']) ? 'selected' : '';

$apartment\_options .= '<option value="' . $row['id'] . '" ' . $apCheck . '>' . $row['apartment\_number'] . '</option>';

}

- لیست آپارتمان‌ها از پایگاه داده دریافت می‌شود.

- یک لیست کشویی (dropdown) از آپارتمان‌ها ایجاد می‌شود و آپارتمان فعلی ساکن به عنوان گزینه‌ی انتخاب شده (selected) تنظیم می‌شود.

نتیجه‌گیری

این کد برای ویرایش اطلاعات ساکنین در سیستم مدیریت شارژ ساختمان طراحی شده است. این کد شامل بخش‌هایی برای بررسی ورود کاربر مدیر، دریافت اطلاعات فرم و به‌روزرسانی اطلاعات ساکن در پایگاه داده است. همچنین اطلاعات ساکن و لیست آپارتمان‌ها برای نمایش در فرم ویرایش دریافت می‌شود.

index.php

<?php

if (isset($\_SESSION['resident\_username'])) {

    header("Location: resident-dashboard.php");

    exit();

} elseif (isset($\_SESSION['admin\_username'])) {

    header("Location: admin-dashboard.php");

    exit();

} else {

    header("Location: login.php");

    exit();

}

این کد PHP برای اتصال کاربران به صفحات مختلف بر اساس نقش آن‌ها در سیستم طراحی شده است. در ادامه به توضیح هر بخش می‌پردازیم:

1. بررسی کاربر ساکن

if (isset($\_SESSION['resident\_username'])) {

header("Location: resident-dashboard.php");

exit();

}

- اگر نام کاربری ساکن در sesion (`$\_SESSION['resident\_username']`) تنظیم شده باشد، کاربر به صفحه‌ی `resident-dashboard.php` هدایت می‌شود.

- `exit()` برای جلوگیری از اجرای کد بیشتر در این فایل استفاده می‌شود.

2. بررسی کاربر مدیر

elseif (isset($\_SESSION['admin\_username'])) {

header("Location: admin-dashboard.php");

exit();

}

- اگر نام کاربری مدیر در session (`$\_SESSION['admin\_username']`) تنظیم شده باشد، کاربر به صفحه‌ی `admin-dashboard.php` هدایت می‌شود.

- `exit()` برای جلوگیری از اجرای کد بیشتر در این فایل استفاده می‌شود.

3. در صورت عدم ورود کاربر

else {

header("Location: login.php");

exit();

}

- اگر هیچ یک از نام کاربری‌ها (ساکن یا مدیر) در session تنظیم نشده باشد، کاربر به صفحه‌ی ورود (`login.php`) هدایت می‌شود.

- `exit()` برای جلوگیری از اجرای کد بیشتر در این فایل استفاده می‌شود.

نکته مهم

- این کد از `header("Location: ...")` برای هدایت کاربر به صفحات دیگر استفاده می‌کند. این دستور باعث می‌شود که مرورگر به طور خودکار به صفحه‌ی مورد نظر هدایت شود و کاربر نمی‌تواند در همان صفحه باقی بماند.

- `exit()` پس از هر دستور `header()` برای جلوگیری از اجرای کد بیشتر در فایل استفاده می‌شود، زیرا که اجرای کد بعد از این دستورات معمولاً معنی ندارد و اجرای آن اضافی است.

invoice.php

<?php

require\_once "config.php";

if (!isset($\_SESSION['resident\_username'])) {

    header("Location: login.php");

    exit();

} elseif (!isset($\_GET['id']) || !is\_numeric($\_GET['id'])) {

    header("Location: resident-dashboard.php");

    exit();

}

$query = "SELECT \* FROM `charges` WHERE `id`='$\_GET[id]'";

$result = mysqli\_query($conn, $query);

$charge = mysqli\_fetch\_assoc($result);

switch ($charge['charge\_type']) {

    case 'parking':

        $chargeType = 'pay-parking-charges';

        $charge['typeText'] = 'شارژ پارکینگ';

        break;

    case 'monthly':

        $chargeType = 'pay-charge-monthly-apartment';

        $charge['typeText'] = 'شارژ ماهانه ساختمان';

        break;

    case 'elevator':

        $chargeType = 'pay-elevator-charge';

        $charge['typeText'] = 'شارژ آسانسور';

        break;

}

$timestamp = strtotime($charge['date\_added']);

$charge['jdate'] = jdate("j F Y", $timestamp);

$sidebarMenuItems[$chargeType]['is-active'] = true;

?>

این کد PHP برای نمایش جزئیات یک شارژ خاص به کاربر ساکن استفاده می‌شود و شامل بررسی‌های زیر است:

1. بررسی وجود ورود کاربر:

require\_once "config.php";

if (!isset($\_SESSION['resident\_username'])) {

header("Location: login.php");

exit();

}

- ابتدا بررسی می‌شود که آیا کاربر وارد شده است یا خیر. اگر نام کاربری ساکن در session (`$\_SESSION['resident\_username']`) تنظیم نشده باشد، کاربر به صفحه‌ی ورود (`login.php`) هدایت می‌شود.

2. بررسی وجود پارامتر `id` در URL و اعتبارسنجی آن:

elseif (!isset($\_GET['id']) || !is\_numeric($\_GET['id'])) {

header("Location: resident-dashboard.php");

exit();

}

- در این قسمت بررسی می‌شود که آیا پارامتر `id` در URL وجود دارد و آیا مقدار آن عددی است یا خیر. اگر پارامتر `id` وجود نداشته یا مقدار آن عددی نبود، کاربر به صفحه‌ی داشبورد ساکن (`resident-dashboard.php`) هدایت می‌شود.

3. دریافت اطلاعات شارژ از پایگاه داده:

$query = "SELECT \* FROM `charges` WHERE `id`='$\_GET[id]'";

$result = mysqli\_query($conn, $query);

$charge = mysqli\_fetch\_assoc($result);

- در این بخش، با استفاده از پارامتر `id` در URL، اطلاعات شارژ مورد نظر از جدول `charges` در پایگاه داده استخراج می‌شود و در آرایه‌ی `$charge` قرار می‌گیرد.

4. تنظیم نوع شارژ و تاریخ به شمسی:

switch ($charge['charge\_type']) {

case 'parking':

$chargeType = 'pay-parking-charges';

$charge['typeText'] = 'شارژ پارکینگ';

break;

case 'monthly':

$chargeType = 'pay-charge-monthly-apartment';

$charge['typeText'] = 'شارژ ماهانه ساختمان';

break;

case 'elevator':

$chargeType = 'pay-elevator-charge';

$charge['typeText'] = 'شارژ آسانسور';

break;

}

- با استفاده از `switch`، نوع شارژ بر اساس مقدار فیلد `charge\_type` در آرایه `$charge` تعیین می‌شود و در متغیر `$chargeType` ذخیره می‌شود. همچنین متن مربوط به نوع شارژ (`$charge['typeText']`) نیز تنظیم می‌شود.

5. تبدیل تاریخ به فرمت شمسی:

$timestamp = strtotime($charge['date\_added']);

$charge['jdate'] = jdate("j F Y", $timestamp);

- تاریخ اضافه شدن شارژ (موجود در `$charge['date\_added']`) ابتدا به فرمت timestamp تبدیل شده و سپس با استفاده از تابع `jdate` به فرمت شمسی تبدیل می‌شود و در `$charge['jdate']` ذخیره می‌شود.

6. تنظیم فعال بودن آیتم منوی سایدبار:

$sidebarMenuItems[$chargeType]['is-active'] = true;

- بر اساس نوع شارژ (`$chargeType`)، متغیر `$sidebarMenuItems` که شامل آیتم‌های منوی سایدبار است، برای آیتم مربوطه (`$chargeType`) به وضعیت فعال (`true`) تنظیم می‌شود.

این کد به کاربر اجازه می‌دهد تا جزئیات شارژ مورد نظر خود را مشاهده کرده و سایدبار را بر اساس نوع شارژ انتخاب شده فعال کند.

login.php

<?php

require\_once "config.php";

if ($\_SERVER["REQUEST\_METHOD"] == "POST") {

    if ($\_POST['user-role'] == 'admin') {

        $username = $\_POST['username'];

        $password = $\_POST['password'];

        $query = "SELECT \* FROM admins WHERE username = '$username' AND password = '$password'";

        $result = mysqli\_query($conn, $query);

        $admin = mysqli\_fetch\_assoc($result);

        if ($admin) {

            $\_SESSION['admin\_username'] = $username;

            $\_SESSION['admin\_name'] = $admin['name'];

            header("Location: admin-dashboard.php");

            exit();

        }

        $loginErr = "نام کاربری یا رمز عبور صحیح نیست.";

    } elseif ($\_POST['user-role'] == 'resident') {

        $contact = $\_POST['username'];

        $password = $\_POST['password'];

        // Use prepared statement to prevent SQL injection

        $query = "SELECT \* FROM residents WHERE contact = ? AND password = ?";

        $stmt = mysqli\_prepare($conn, $query);

        mysqli\_stmt\_bind\_param($stmt, "ss", $contact, $password);

        mysqli\_stmt\_execute($stmt);

        $result = mysqli\_stmt\_get\_result($stmt);

        $resident = mysqli\_fetch\_assoc($result);

        if ($resident) {

            $\_SESSION['resident\_username'] = $resident['name']; // Store name in session

            $\_SESSION['resident\_id'] = $resident['id']; // Store resident ID in session

            header("Location: resident-dashboard.php");

            exit();

        }

        $loginErr = "نام کاربری یا رمز عبور صحیح نیست.";

    } else {

        $loginErr = "نقش کاربری به درستی انتخاب نشده است.";

    }

}

این کد PHP برای احراز هویت کاربران در سیستم استفاده می‌شود و شامل موارد زیر است:

1. بررسی درخواست فرم ارسالی (POST):

require\_once "config.php";

if ($\_SERVER["REQUEST\_METHOD"] == "POST") {

- کد ابتدا بررسی می‌کند که آیا درخواست ارسالی از نوع POST است یا خیر.

2. احراز هویت برای نقش مدیر (admin):

if ($\_POST['user-role'] == 'admin') {

$username = $\_POST['username'];

$password = $\_POST['password'];

$query = "SELECT \* FROM admins WHERE username = '$username' AND password = '$password'";

$result = mysqli\_query($conn, $query);

$admin = mysqli\_fetch\_assoc($result);

if ($admin) {

$\_SESSION['admin\_username'] = $username;

$\_SESSION['admin\_name'] = $admin['name'];

header("Location: admin-dashboard.php");

exit();

}

$loginErr = "نام کاربری یا رمز عبور صحیح نیست.";

}

- اگر نقش کاربری `admin` انتخاب شده باشد، نام کاربری و رمز عبور از فرم POST گرفته می‌شود. سپس با استفاده از یک کوئری SQL، اطلاعات مدیر متناظر با نام کاربری و رمز عبور وارد شده در پایگاه داده جستجو می‌شود. اگر اطلاعات مدیر معتبر باشد، نام کاربری مدیر و نام او در session ذخیره شده و کاربر به صفحه‌ی داشبورد مدیر (`admin-dashboard.php`) هدایت می‌شود.

3. احراز هویت برای نقش ساکن (resident):

elseif ($\_POST['user-role'] == 'resident') {

$contact = $\_POST['username'];

$password = $\_POST['password'];

// Use prepared statement to prevent SQL injection

$query = "SELECT \* FROM residents WHERE contact = ? AND password = ?";

$stmt = mysqli\_prepare($conn, $query);

mysqli\_stmt\_bind\_param($stmt, "ss", $contact, $password);

mysqli\_stmt\_execute($stmt);

$result = mysqli\_stmt\_get\_result($stmt);

$resident = mysqli\_fetch\_assoc($result);

if ($resident) {

$\_SESSION['resident\_username'] = $resident['name']; // Store name in session

$\_SESSION['resident\_id'] = $resident['id']; // Store resident ID in session

header("Location: resident-dashboard.php");

exit();

}

$loginErr = "نام کاربری یا رمز عبور صحیح نیست.";

}

- اگر نقش کاربری `resident` انتخاب شده باشد، شماره تماس (contact) و رمز عبور از فرم POST گرفته می‌شود. در اینجا از استفاده از prepared statement استفاده می‌شود تا از حملات SQL injection جلوگیری شود. سپس با استفاده از کوئری آماده شده، اطلاعات ساکن متناظر با شماره تماس و رمز عبور وارد شده در پایگاه داده جستجو می‌شود. اگر اطلاعات ساکن معتبر باشد، نام ساکن و شناسه او در session ذخیره شده و کاربر به صفحه‌ی داشبورد ساکن (`resident-dashboard.php`) هدایت می‌شود.

4. بررسی نقش کاربری نامعتبر:

else {

$loginErr = "نقش کاربری به درستی انتخاب نشده است.";

}

- اگر نقش کاربری ارسال شده (`admin` یا `resident`) معتبر نباشد، یک پیام خطا (`$loginErr`) تنظیم می‌شود که نقش کاربری به درستی انتخاب نشده است.

این کد PHP به کمک پایگاه داده MySQL احراز هویت کاربران را انجام می‌دهد و بسته به نقش کاربری انتخاب شده، کاربر را به صفحه‌ی مناسب هدایت می‌کند یا پیام خطا مناسب را نمایش می‌دهد.

logout.php

<?php

session\_start();

// Unset all of the session variables

$\_SESSION = array();

// Destroy the session

session\_destroy();

// Redirect to login page

header("Location: login.php");

exit();

?>

این کد PHP برای خروج از سیستم کاربران و حذف همه‌ی متغیرهای جلسه (session variables) مورد استفاده قرار می‌گیرد. زمانی که کاربر این فایل را اجرا می‌کند، کلیه‌ی متغیرهای جلسه‌ای که در حال حاضر تنظیم شده‌اند، حذف می‌شوند و سپس جلسه نابود می‌شود. سپس کاربر به صفحه‌ی ورود (`login.php`) هدایت می‌شود.

به طور دقیق، کد عملیات زیر را انجام می‌دهد:

1. session\_start():

- این تابع جلسه را شروع می‌کند یا اگر قبلاً شروع شده بود، جلسه فعلی را باز می‌کند.

2. $\_SESSION = array():

- این دستور تمامی متغیرهای جلسه را با یک آرایه‌ی خالی جایگزین می‌کند. این کار باعث حذف همه‌ی متغیرهای جلسه می‌شود.

3. session\_destroy():

- این تابع جلسه را نابود می‌کند، به این معنی که تمام داده‌های جلسه شامل همه‌ی متغیرها و تنظیمات جلسه‌ای حذف می‌شوند.

4. header("Location: login.php"):

- این دستور به کاربر می‌گوید که به صفحه‌ی ورود (`login.php`) هدایت شود.

5. exit():

- این دستور جلوگیری می‌کند که هر دستور دیگری بر روی صفحه‌ی فعلی اجرا شود، به طوری که بلافاصله پس از اجرای این دستور، کنترل به صفحه‌ی ورود هدایت می‌شود.

این کد به طور کامل از جلسه استفاده کرده و با حذف متغیرهای جلسه و نابود کردن جلسه، از کاربر خروج را انجام می‌دهد و اطمینان حاصل می‌شود که دیگری به اطلاعات یا صفحات حساس دسترسی ندارد.

pay-elevator-charge.php

<?php

require\_once "config.php";

if (!isset($\_SESSION['resident\_username'])) {

    header("Location: login.php");

    exit();

}

if ($\_SERVER["REQUEST\_METHOD"] == "POST") {

    $resident\_id = $\_SESSION['resident\_id']; // Store resident ID in session upon login

    $amount = $\_POST['amount'];

    $query = "INSERT INTO charges (charge\_type, amount, date\_added, resident\_id) VALUES ('elevator', '$amount', NOW(), '$resident\_id')";

    if (mysqli\_query($conn, $query)) {

        $chargesId = mysqli\_insert\_id($conn);

        header("Location: invoice.php?id=" . $chargesId);

        exit();

    } else {

        echo "Error: " . mysqli\_error($conn);

    }

}

$sidebarMenuItems['pay-elevator-charge']['is-active'] = true;

?>

این کد PHP برای افزودن شارژ آسانسور به حساب ساکن استفاده می‌شود. زمانی که یک ساکن وارد حساب خود در سیستم شود و فرمی را پر کند و ارسال کند، شارژ جدید آسانسور به پایگاه داده اضافه می‌شود. سپس کاربر به صفحه‌ی فاکتور (`invoice.php`) هدایت می‌شود تا جزئیات شارژ اضافه شده را ببیند.

توضیحات کد:

1. اعتبارسنجی ورودی:

- کد ابتدا از وجود یک جلسه برای کنترل دسترسی مطمئن می‌شود. اگر هیچ کاربری در حال حاضر وارد نشده باشد (`$\_SESSION['resident\_username']` وجود نداشته باشد)، کاربر به صفحه‌ی ورود (`login.php`) هدایت می‌شود.

2. پردازش فرم:

- اگر درخواست از نوع POST باشد، اطلاعات ارسالی از فرم دریافت می‌شود.

- `$\_SESSION['resident\_id']` که هویت ساکن را نگه می‌دارد، برای افزودن شارژ به حساب ساکن استفاده می‌شود.

- اطلاعات شارژ شامل نوع شارژ (elevator)، مبلغ، تاریخ و id ساکن در جدول charges ذخیره می‌شود.

3. اضافه کردن شارژ:

- اگر افزودن شارژ با موفقیت انجام شود (`mysqli\_query($conn, $query)` بازگشت true داشته باشد)، id جدید شارژ (charges\_id) با `mysqli\_insert\_id($conn)` گرفته می‌شود.

- سپس کاربر به صفحه‌ی فاکتور (`invoice.php?id=chargesId`) هدایت می‌شود تا جزئیات شارژ اضافه شده را ببیند.

4. نمایش خطا:

- اگر افزودن شارژ با مشکل مواجه شود، پیغام خطا به کاربر نمایش داده می‌شود.

5. تنظیمات منوی کناری:

- این بخش از کد برای تنظیم منوی کناری صفحه‌ی پرداخت شارژ آسانسور (`pay-elevator-charge`) به وضعیت فعال تنظیم شده است.

pay-parking-charges.php

<?php

require\_once "config.php";

if (!isset($\_SESSION['resident\_username'])) {

    header("Location: login.php");

    exit();

}

if ($\_SERVER["REQUEST\_METHOD"] == "POST") {

    $resident\_id = $\_SESSION['resident\_id']; // Store resident ID in session upon login

    $amount = $\_POST['amount'];

    $query = "INSERT INTO charges (charge\_type, amount, date\_added, resident\_id) VALUES ('parking', '$amount', NOW(), '$resident\_id')";

    if (mysqli\_query($conn, $query)) {

        $chargesId = mysqli\_insert\_id($conn);

        header("Location: invoice.php?id=" . $chargesId);

        exit();

    } else {

        echo "Error: " . mysqli\_error($conn);

    }

}

$sidebarMenuItems['pay-parking-charges']['is-active'] = true;

?>

کد PHP که شما ارائه داده‌اید، برای افزودن شارژ پارکینگ به حساب ساکن استفاده می‌شود. زمانی که یک ساکن وارد حساب کاربری خود شده و فرم پرداخت شارژ پارکینگ را تکمیل کند و ارسال کند، اطلاعات شارژ جدید به پایگاه داده اضافه می‌شود و سپس کاربر به صفحه‌ی فاکتور (`invoice.php`) هدایت می‌شود تا جزئیات شارژ اضافه شده را ببیند.

توضیحات کد:

1. اعتبارسنجی ورودی:

- کد ابتدا از وجود یک جلسه برای کنترل دسترسی مطمئن می‌شود. اگر هیچ کاربری در حال حاضر وارد نشده باشد (`$\_SESSION['resident\_username']` وجود نداشته باشد)، کاربر به صفحه‌ی ورود (`login.php`) هدایت می‌شود.

2. پردازش فرم:

- اگر درخواست از نوع POST باشد، اطلاعات ارسالی از فرم دریافت می‌شود.

- `$\_SESSION['resident\_id']` که هویت ساکن را نگه می‌دارد، برای افزودن شارژ به حساب ساکن استفاده می‌شود.

- اطلاعات شارژ شامل نوع شارژ (parking)، مبلغ، تاریخ و id ساکن در جدول charges ذخیره می‌شود.

3. اضافه کردن شارژ:

- اگر افزودن شارژ با موفقیت انجام شود (`mysqli\_query($conn, $query)` بازگشت true داشته باشد)، id جدید شارژ (charges\_id) با `mysqli\_insert\_id($conn)` گرفته می‌شود.

- سپس کاربر به صفحه‌ی فاکتور (`invoice.php?id=chargesId`) هدایت می‌شود تا جزئیات شارژ اضافه شده را ببیند.

4. نمایش خطا:

- اگر افزودن شارژ با مشکل مواجه شود، پیغام خطا به کاربر نمایش داده می‌شود.

5. تنظیمات منوی کناری:

- این بخش از کد برای تنظیم منوی کناری صفحه‌ی پرداخت شارژ پارکینگ (`pay-parking-charges`) به وضعیت فعال تنظیم شده است.

resident-register.php

<?php

require\_once 'config.php'; // Database connection

// Fetch apartment numbers for dropdown

$query\_apartments = "SELECT id, apartment\_number FROM apartments";

$result\_apartments = mysqli\_query($conn, $query\_apartments);

$apartment\_options = '';

while ($row = mysqli\_fetch\_assoc($result\_apartments)) {

    $apartment\_options .= '<option value="' . $row['id'] . '">' . $row['apartment\_number'] . '</option>';

}

if ($\_SERVER["REQUEST\_METHOD"] == "POST") {

    $name = $\_POST['name'];

    $apartment\_id = $\_POST['apartment\_id'];

    $contact = $\_POST['contact'];

    $password = $\_POST['password']; // New password field

    $is\_primary\_resident = isset($\_POST['is\_primary\_resident']) ? 1 : 0;

    // Use prepared statement to prevent SQL injection

    $query = "INSERT INTO residents (name, apartment\_id, contact, password, is\_primary\_resident) VALUES (?, ?, ?, ?, ?)";

    $stmt = mysqli\_prepare($conn, $query);

    mysqli\_stmt\_bind\_param($stmt, "sissi", $name, $apartment\_id, $contact, $password, $is\_primary\_resident);

    if (mysqli\_stmt\_execute($stmt)) {

        session\_start();

        $\_SESSION['resident\_username'] = $name;

        $\_SESSION['resident\_id'] = mysqli\_insert\_id($conn); // Store resident ID in session

        header("Location: login.php");

        exit();

    } else {

        echo "Error: " . mysqli\_error($conn);

    }

}

?>

این قطعه کد PHP برای یک فرم ثبت‌نام ساکنان در یک سیستم مدیریت ساختمان نوشته شده است. حالا به صورت جزئی توضیحاتی راجع به هر بخش ارائه می‌دهم:

1. require\_once 'config.php';:

این خط کد برای اتصال به پایگاه داده استفاده می‌شود. فایل `config.php` حاوی تنظیمات اتصال به پایگاه داده (مانند نام کاربری، رمز عبور، نام دیتابیس و ...) است.

2. $query\_apartments:

در این بخش، یک پرس و جو SQL نوشته شده است که همه شماره‌های آپارتمان را برای استفاده در یک لیست انتخابی (dropdown) بازیابی می‌کند. این پرس‌وجو از جدول `apartments` اطلاعات را بازیابی می‌کند.

3. $result\_apartments:

نتیجه اجرای پرس‌وجو به کمک تابع `mysqli\_query` در اینجا ذخیره می‌شود.

4. $apartment\_options:

یک رشته است که به آرایه‌ای از آپشن‌های HTML برای dropdown تبدیل می‌شود. هر مورد با استفاده از داده‌هایی که از جدول بازیابی شده است، ساخته می‌شود.

5. if ($\_SERVER["REQUEST\_METHOD"] == "POST"):

این بخش بررسی می‌کند که آیا فرم توسط HTTP POST ارسال شده است یا نه.

6. $name, $apartment\_id, $contact, $password, $is\_primary\_resident:

این متغیرها از داده‌هایی که توسط کاربر از طریق فرم ارسال شده‌اند، پر می‌شوند.

7. $query:

یک پرس‌وجو SQL برای درج اطلاعات ساکنان جدید به جدول `residents` تعریف می‌کند. اینجا از prepared statement برای جلوگیری از حملات SQL injection استفاده می‌شود.

8. mysqli\_stmt\_bind\_param:

این تابع برای پیوند دادن متغیرهای PHP به پارامترهای پرس‌وجوی SQL در prepared statement استفاده می‌شود.

9. mysqli\_stmt\_execute:

این تابع پرس‌وجو را اجرا می‌کند و اگر اجرای آن موفق باشد، جلسه‌ای برای کاربر شروع می‌کند، اطلاعات ساکن جدید را در جلسه ذخیره می‌کند و کاربر را به صفحه ورود (`login.php`) هدایت می‌کند.

10. session\_start() و $\_SESSION:

این بخش‌ها برای مدیریت جلسات در PHP استفاده می‌شود، تا اطلاعات کاربر (مانند نام کاربری و شناسه) در سرور نگهداری شود.

11. header("Location: login.php"):

با این دستور، کاربر به صفحه ورود هدایت می‌شود.

12. echo "Error: " . mysqli\_error($conn):

اگر در اجرای پرس‌وجو مشکلی پیش‌آید، این پیام خطا نمایش داده می‌شود.

این کد به طور کلی برای اضافه کردن اطلاعات ساکنان جدید به پایگاه داده و مدیریت جلسات در PHP طراحی شده است.

view-apartment.php

<?php

require\_once "config.php";

if (!isset($\_SESSION['admin\_username'])) {

    header("Location: login.php");

    exit();

}

$query = "SELECT \* FROM apartments";

$result = mysqli\_query($conn, $query);

$adminSidebarMenuItems['view-apartment']['is-active'] = true;

?>

این قطعه کد PHP به زبان فارسی به صورت جزئی توضیح داده شده است:

php

<?php

require\_once "config.php";

// بررسی می‌کنیم که آیا کاربر ادمین وارد شده است یا نه

if (!isset($\_SESSION['admin\_username'])) {

// اگر کاربر ادمین وارد نشده باشد، به صفحه ورود هدایت می‌شود

header("Location: login.php");

exit(); // پس از ارسال header، برنامه خاتمه می‌یابد

}

// درخواست برای بازیابی اطلاعات از جدول آپارتمان‌ها

$query = "SELECT \* FROM apartments";

$result = mysqli\_query($conn, $query);

// تنظیم گزینه‌های منوی کناری برای منوی مشاهده آپارتمان

$adminSidebarMenuItems['view-apartment']['is-active'] = true;

?>

توضیحات:

1. require\_once "config.php";:

این خط کد برای اتصال به پایگاه داده با استفاده از فایل `config.php` است. این فایل حاوی تنظیمات اتصال به پایگاه داده است که در اینجا به صورت یکبار فقط وارد می‌شود.

2. if (!isset($\_SESSION['admin\_username'])) { ... }:

این شرط بررسی می‌کند که آیا کاربر ادمین وارد شده است یا خیر. اگر کاربر ادمین وارد نشده باشد، با استفاده از `header("Location: login.php");` به صفحه ورود هدایت می‌شود و پس از آن اجرای برنامه خاتمه می‌یابد.

3. $query = "SELECT \* FROM apartments";:

این پرس و جو SQL برای بازیابی همه رکوردهای موجود در جدول آپارتمان‌ها تعریف شده است.

4. $result = mysqli\_query($conn, $query);:

اینجا پرس و جو اجرا می‌شود و نتیجه آن در متغیر `$result` ذخیره می‌شود. این نتیجه شامل تمام ردیف‌های بازگردانده شده از پایگاه داده است.

5. $adminSidebarMenuItems['view-apartment']['is-active'] = true;:

این خط کد تنظیم می‌کند که گزینه منوی کناری با نام "view-apartment" فعال باشد. این معمولا برای نشان دادن کدام بخش از منو انتخاب شده است یا فعال است.

این قطعه کد به طور کلی برای اطمینان از ورود کاربر ادمین و سپس نمایش اطلاعات آپارتمان‌ها در یک سیستم مدیریتی طراحی شده است.

view-charges.php

<?php

require\_once "config.php";

if (!isset($\_SESSION['admin\_username'])) {

    header("Location: login.php");

    exit();

}

$resident\_id = 0;

if ($\_SERVER["REQUEST\_METHOD"] == "POST") {

    $resident\_id = $\_POST['resident\_id'];

    // Query to fetch charges for the selected resident

    $query\_charges = "SELECT \* FROM charges WHERE resident\_id = $resident\_id";

    $result\_charges = mysqli\_query($conn, $query\_charges);

}

// Fetch residents for dropdown

$query\_residents = "SELECT id, name FROM residents";

$result\_residents = mysqli\_query($conn, $query\_residents);

$resident\_options = '';

while ($row = mysqli\_fetch\_assoc($result\_residents)) {

    $reCheck = ($row['id'] == $resident\_id) ? 'selected' : '';

    $resident\_options .= '<option value="' . $row['id'] . '" ' . $reCheck . '>' . $row['name'] . '</option>';

}

$adminSidebarMenuItems['view-charges']['is-active'] = true;

?>

این قطعه کد PHP به زبان فارسی به صورت جزئی توضیح داده شده است:

php

<?php

require\_once "config.php";

// بررسی می‌کنیم که آیا کاربر ادمین وارد شده است یا نه

if (!isset($\_SESSION['admin\_username'])) {

// اگر کاربر ادمین وارد نشده باشد، به صفحه ورود هدایت می‌شود

header("Location: login.php");

exit(); // پس از ارسال header، برنامه خاتمه می‌یابد

}

$resident\_id = 0;

if ($\_SERVER["REQUEST\_METHOD"] == "POST") {

// اگر درخواست ارسال فرم POST باشد، آنگاه شناسه ساکن از فرم دریافت می‌شود

$resident\_id = $\_POST['resident\_id'];

// پرس‌وجو برای بازیابی هزینه‌ها برای ساکن انتخاب شده

$query\_charges = "SELECT \* FROM charges WHERE resident\_id = $resident\_id";

$result\_charges = mysqli\_query($conn, $query\_charges);

}

// درخواست برای بازیابی اطلاعات ساکنان برای dropdown

$query\_residents = "SELECT id, name FROM residents";

$result\_residents = mysqli\_query($conn, $query\_residents);

$resident\_options = '';

while ($row = mysqli\_fetch\_assoc($result\_residents)) {

// بررسی می‌کنیم که آیا این ساکن انتخاب شده است یا خیر

$reCheck = ($row['id'] == $resident\_id) ? 'selected' : '';

// اضافه کردن آپشن‌های dropdown بر اساس اطلاعات بازیابی شده

$resident\_options .= '<option value="' . $row['id'] . '" ' . $reCheck . '>' . $row['name'] . '</option>';

}

// تنظیم گزینه‌های منوی کناری برای منوی مشاهده هزینه‌ها

$adminSidebarMenuItems['view-charges']['is-active'] = true;

?>

توضیحات:

1. require\_once "config.php";:

این خط کد برای اتصال به پایگاه داده با استفاده از فایل `config.php` است. این فایل حاوی تنظیمات اتصال به پایگاه داده است که در اینجا به صورت یکبار فقط وارد می‌شود.

2. if (!isset($\_SESSION['admin\_username'])) { ... }:

این شرط بررسی می‌کند که آیا کاربر ادمین وارد شده است یا خیر. اگر کاربر ادمین وارد نشده باشد، با استفاده از `header("Location: login.php");` به صفحه ورود هدایت می‌شود و پس از آن اجرای برنامه خاتمه می‌یابد.

3. $resident\_id = 0;:

این متغیر برای ذخیره کردن شناسه ساکنی است که از فرم POST دریافت می‌شود. در ابتدا به صفر تنظیم شده است.

4. if ($\_SERVER["REQUEST\_METHOD"] == "POST") { ... }:

این بخش بررسی می‌کند که آیا فرم با استفاده از درخواست POST ارسال شده است یا خیر. اگر ارسال شده باشد، شناسه ساکن از فرم دریافت می‌شود و پرس‌وجویی برای بازیابی هزینه‌ها برای آن ساکن انجام می‌شود.

5. $query\_charges = "SELECT \* FROM charges WHERE resident\_id = $resident\_id";:

این پرس‌وجو SQL برای بازیابی تمام هزینه‌هایی که به شناسه ساکن مرتبط باشد، تعریف شده است.

6. $result\_residents = mysqli\_query($conn, $query\_residents);:

اینجا پرس‌وجوی SQL برای بازیابی همه اطلاعات ساکنان (شامل شناسه و نام) برای ساختن یک dropdown تعریف شده است.

7. while ($row = mysqli\_fetch\_assoc($result\_residents)) { ... }:

در این حلقه، اطلاعات هر ساکن بازیابی شده از پایگاه داده را بررسی می‌کند و آپشن‌های HTML برای dropdown را بر اساس اطلاعات ساخته می‌شود. همچنین بررسی می‌شود که آیا ساکن انتخاب شده است یا خیر تا بر اساس آن، آپشن موردنظر به عنوان `selected` تنظیم شود.

8. $adminSidebarMenuItems['view-charges']['is-active'] = true;:

این خط کد تنظیم می‌کند که گزینه منوی کناری با نام "view-charges" فعال باشد. این معمولا برای نشان دادن کدام بخش از منو انتخاب شده است یا فعال است.

این قطعه کد به طور کلی برای بازیابی هزینه‌های مربوط به یک ساکن انتخاب شده از طریق یک فرم و نمایش انتخاب ساکنان در یک dropdown در یک سیستم مدیریتی استفاده می‌شود.

view-late-penalty.php

<?php

require\_once "config.php";

if (!isset($\_SESSION['admin\_username'])) {

    header("Location: login.php");

    exit();

}

$query = "SELECT penalties.id, residents.name AS resident\_name, penalties.amount, penalties.date\_added

          FROM penalties

          INNER JOIN residents ON penalties.resident\_id = residents.id";

$result = mysqli\_query($conn, $query);

$adminSidebarMenuItems['view-late-penalty']['is-active'] = true;

?>

این قطعه کد PHP به زبان فارسی به صورت جزئی توضیح داده شده است:

php

<?php

require\_once "config.php";

// بررسی می‌کنیم که آیا کاربر ادمین وارد شده است یا نه

if (!isset($\_SESSION['admin\_username'])) {

// اگر کاربر ادمین وارد نشده باشد، به صفحه ورود هدایت می‌شود

header("Location: login.php");

exit(); // پس از ارسال header، برنامه خاتمه می‌یابد

}

// پرس‌وجو برای بازیابی اطلاعات جریمه‌ها همراه با نام ساکن و تاریخ واریز

$query = "SELECT penalties.id, residents.name AS resident\_name, penalties.amount, penalties.date\_added

FROM penalties

INNER JOIN residents ON penalties.resident\_id = residents.id";

$result = mysqli\_query($conn, $query);

// تنظیم گزینه‌های منوی کناری برای منوی مشاهده جریمه‌های تأخیر

$adminSidebarMenuItems['view-late-penalty']['is-active'] = true;

?>

توضیحات:

1. require\_once "config.php";:

این خط کد برای اتصال به پایگاه داده با استفاده از فایل `config.php` است. این فایل حاوی تنظیمات اتصال به پایگاه داده است که در اینجا به صورت یکبار فقط وارد می‌شود.

2. if (!isset($\_SESSION['admin\_username'])) { ... }:

این شرط بررسی می‌کند که آیا کاربر ادمین وارد شده است یا خیر. اگر کاربر ادمین وارد نشده باشد، با استفاده از `header("Location: login.php");` به صفحه ورود هدایت می‌شود و پس از آن اجرای برنامه خاتمه می‌یابد.

3. $query:

این پرس‌وجو SQL برای بازیابی اطلاعات جریمه‌ها از جدول `penalties` استفاده می‌کند. با استفاده از JOIN با جدول `residents`، نام ساکن (که با نام `residents.name` نمایش داده می‌شود) همراه با هزینه جریمه (`penalties.amount`) و تاریخ اضافه شدن جریمه (`penalties.date\_added`) را بازیابی می‌کند.

4. $result = mysqli\_query($conn, $query);:

اینجا پرس‌وجو اجرا می‌شود و نتیجه آن در متغیر `$result` ذخیره می‌شود. این نتیجه شامل تمام ردیف‌های بازگردانده شده از پایگاه داده است که هر ردیف شامل شناسه جریمه، نام ساکن، مبلغ جریمه و تاریخ اضافه شدن است.

5. $adminSidebarMenuItems['view-late-penalty']['is-active'] = true;:

این خط کد تنظیم می‌کند که گزینه منوی کناری با نام "view-late-penalty" فعال باشد. این معمولا برای نشان دادن کدام بخش از منو انتخاب شده است یا فعال است.

این قطعه کد به طور کلی برای بازیابی و نمایش جریمه‌های مرتبط با ساکنان در یک سیستم مدیریتی استفاده می‌شود، و از JOIN برای ارتباط دادن جداول استفاده می‌کند تا اطلاعات مربوطه را با هم مرتبط کند.

view-resident.php

<?php

require\_once "config.php";

if (!isset($\_SESSION['admin\_username'])) {

    header("Location: login.php");

    exit();

}

if (isset($\_GET['action'])) {

    if ($\_GET['action'] == 'delete') {

        $resident\_id = $\_GET['rId'];

        $query = "DELETE FROM residents WHERE id = '$resident\_id'";

        if (mysqli\_query($conn, $query)) {

            header("Location: view-resident.php?action=success&text=حذف ساکن با موفقیت انجام شد");

        } else {

            header("Location: view-resident.php?action=danger&text=" . mysqli\_error($conn));

        }

    } elseif ($\_GET['action'] == 'success') {

        $deleteResponse = ['success', $\_GET['text']];

    } elseif ($\_GET['action'] == 'danger') {

        $deleteResponse = ['danger', "Error: " . $\_GET['text']];

    }

}

$query = "SELECT residents.\*, apartments.apartment\_number

          FROM residents

          INNER JOIN apartments ON residents.apartment\_id = apartments.id";

$result = mysqli\_query($conn, $query);

$adminSidebarMenuItems['view-resident']['is-active'] = true;

?>

این قطعه کد PHP به زبان فارسی به صورت جزئی توضیح داده شده است:

php

<?php

require\_once "config.php";

// بررسی می‌کنیم که آیا کاربر ادمین وارد شده است یا نه

if (!isset($\_SESSION['admin\_username'])) {

// اگر کاربر ادمین وارد نشده باشد، به صفحه ورود هدایت می‌شود

header("Location: login.php");

exit(); // پس از ارسال header، برنامه خاتمه می‌یابد

}

// اگر پارامتر action در URL وجود داشته باشد، آن را بررسی می‌کنیم

if (isset($\_GET['action'])) {

if ($\_GET['action'] == 'delete') {

// اگر عملیات حذف درخواست شده باشد، شناسه ساکن را از URL دریافت می‌کنیم

$resident\_id = $\_GET['rId'];

// پرس‌وجو برای حذف ساکن با استفاده از شناسه مورد نظر

$query = "DELETE FROM residents WHERE id = '$resident\_id'";

// اگر پرس‌وجو با موفقیت اجرا شود، کاربر به صفحه مشاهده ساکنان هدایت می‌شود

if (mysqli\_query($conn, $query)) {

header("Location: view-resident.php?action=success&text=حذف ساکن با موفقیت انجام شد");

} else {

// اگر خطایی در اجرای پرس‌وجو رخ دهد، کاربر به صفحه مشاهده ساکنان با پیام خطا هدایت می‌شود

header("Location: view-resident.php?action=danger&text=" . mysqli\_error($conn));

}

} elseif ($\_GET['action'] == 'success') {

// اگر عملیات با موفقیت انجام شده باشد، پیام موفقیت آمیز را نمایش می‌دهیم

$deleteResponse = ['success', $\_GET['text']];

} elseif ($\_GET['action'] == 'danger') {

// اگر خطایی رخ داده باشد، پیام خطا را نمایش می‌دهیم

$deleteResponse = ['danger', "Error: " . $\_GET['text']];

}

}

// پرس‌وجو برای بازیابی اطلاعات ساکنان همراه با شماره آپارتمان آن‌ها

$query = "SELECT residents.\*, apartments.apartment\_number

FROM residents

INNER JOIN apartments ON residents.apartment\_id = apartments.id";

$result = mysqli\_query($conn, $query);

// تنظیم گزینه‌های منوی کناری برای منوی مشاهده ساکنان

$adminSidebarMenuItems['view-resident']['is-active'] = true;

?>

توضیحات:

1. require\_once "config.php";:

این خط کد برای اتصال به پایگاه داده با استفاده از فایل `config.php` است. این فایل حاوی تنظیمات اتصال به پایگاه داده است که در اینجا به صورت یکبار فقط وارد می‌شود.

2. if (!isset($\_SESSION['admin\_username'])) { ... }:

این شرط بررسی می‌کند که آیا کاربر ادمین وارد شده است یا خیر. اگر کاربر ادمین وارد نشده باشد، با استفاده از `header("Location: login.php");` به صفحه ورود هدایت می‌شود و پس از آن اجرای برنامه خاتمه می‌یابد.

3. if (isset($\_GET['action'])) { ... }:

این بخش بررسی می‌کند که آیا پارامتر action در URL وجود دارد یا خیر. اگر وجود داشته باشد، می‌تواند اعمالی مانند حذف یا نمایش پیام‌های موفقیت و خطا را انجام دهد.

4. $query = "DELETE FROM residents WHERE id = '$resident\_id'";:

این پرس‌وجو SQL برای حذف ساکن با استفاده از شناسه (`id`) مورد نظر تعریف شده است.

5. header("Location: ...");:

این دستورات `header` برای انتقال کاربر به صفحات مختلف به توجه به نتایج اجرای پرس‌وجوها (مانند حذف موفق یا خطا) استفاده می‌شود.

6. $adminSidebarMenuItems['view-resident']['is-active'] = true;:

این خط کد تنظیم می‌کند که گزینه منوی کناری با نام "view-resident" فعال باشد. این معمولا برای نشان دادن کدام بخش از منو انتخاب شده است یا فعال است.

اجرای نرم­افزار

صفحه ی پرتال کاربری:

https://ams.ar100demo.ir/dashboard

نقش مدیر:

admin

admin123

—-

نقش ساکن :

0912123214

12345

صفحه معرفی:

https://ams.ar100demo.ir/

نتیجه‌گیری

نتیجه‌گیری از پیاده‌سازی سیستم مدیریت و شارژ ساختمان

در این پروژه، یک سیستم مدیریت و شارژ ساختمان طراحی و پیاده‌سازی شد که شامل امکانات گوناگونی برای مدیریت ساکنین و امکانات ساختمان است. ماژول‌های اصلی سیستم عبارت بودند از:

1. مدیریت ساکنین:

- افزودن و مشاهده خانواده‌ها.

جمع‌بندی و پیشنهادها (فصل 5: جمع‌بندی و پیشنهادها)

در فصل بعدی از گزارش، که فصل 5 با عنوان "جمع‌بندی و پیشنهادها" است، ما قصد داریم به بررسی عملکرد سیستم پرداخته و نتایج حاصل از پیاده‌سازی را مورد بررسی قرار دهیم. در این فصل، به موارد زیر پرداخته خواهد شد:

- جمع‌بندی عملیات اصلی سیستم: بررسی کلی عملکرد ماژول‌ها و کارایی آن‌ها در مدیریت ساختمان و خدمات مربوطه.

- ارزیابی کیفیت و قابلیت اطمینان سیستم: موردی که می‌توان به آن پرداخت، ارزیابی عملکرد سیستم در شرایط مختلف و مدت زمان طولانی.

- پیشنهادها برای بهبود سیستم: ارائه پیشنهادات و راهکارهایی برای بهبود عملکرد، افزایش امنیت و کارایی سیستم.

- نتیجه‌گیری و تأثیرات مثبت سیستم بر کاربران: ارائه نتایج نهایی پروژه و تأثیرات مثبت آن بر ساکنین و مدیران ساختمان.

این فصل به عنوان نتیجه‌گیری نهایی پروژه، امکان می‌دهد تا جوانب مختلف پیاده‌سازی و عملکرد سیستم را به طور جامع بررسی کرده و مسیرهای بهبود و پیشنهادات مفید را برای آینده تدارک دید.

فصل پنجم :  
جمع‌بندی و پیشنهادها

**5-1- نتیجه گیری**

در این پروژه، سیستم مدیریت شارژ ساختمان با هدف بهبود روند مدیریت مالی و اداری ساکنان ساختمان طراحی و پیاده‌سازی شد. ماژول‌های اصلی این سیستم شامل مدیریت ساکنان، امکانات ساختمان، تابلو اعلانات، درخواست‌ها، قرار ملاقات‌ها و پرداخت‌ها می‌باشد. با پیاده‌سازی این سیستم، امکان مشاهده و مدیریت دقیق‌تر و شفاف‌تر امور مالی و اطلاع‌رسانی به ساکنان فراهم شده است. تست‌ها و ارزیابی‌های انجام‌شده نشان‌دهنده کارایی و قابل‌اعتماد بودن این سیستم در شرایط مختلف بوده است

پیشنهادهایی برای کارهای آتی

گسترش ماژول‌های سیستم: اضافه کردن ماژول‌های جدید برای پوشش نیازهای بیشتر ساکنان و مدیریت بهتر امکانات ساختمان.

بهبود رابط کاربری: ارتقاء رابط کاربری برای افزایش تجربه کاربری و سهولت استفاده از سیستم.

ارتقاء امنیت: پیاده‌سازی مکانیزم‌های امنیتی پیشرفته‌تر برای حفاظت از داده‌های ساکنان و اطلاعات مالی.

یکپارچه‌سازی با سیستم‌های خارجی: توسعه قابلیت‌های یکپارچه‌سازی سیستم با درگاه‌های پرداخت و بانک‌ها برای تسهیل بیشتر در پرداخت‌ها و مدیریت مالی.

تحلیل داده‌ها: افزودن امکانات تحلیل و گزارش‌گیری پیشرفته برای مدیریت بهتر اطلاعات و تصمیم‌گیری‌های استراتژی

منابع

1. کتاب‌ها و مقالات علمی:

- کتاب "مهندسی نرم‌افزار"، نویسنده: ایان سامرویل، مترجم: جعفرنژاد قمی، ناشر: دانشگاه صنعتی شریف.

- کتاب "سیستم‌های مدیریت پایگاه داده"، نویسنده: ابراهیم عامری، ناشر: دانشگاه تهران.

- مقاله "طراحی و پیاده‌سازی سیستم‌های اطلاعاتی"، نویسنده: محمدرضا نیلچیان، مجله علمی پژوهشی دانشگاه تهران.

2. وب‌سایت‌ها:

- وب‌سایت "آموزش برنامه‌نویسی"، عنوان مطلب: "آموزش PHP"، [www.programmingiran.com](http://www.programmingiran.com)، دسترسی در تاریخ 2024/06/27.

- وب‌سایت "راهنمای جامع برنامه‌نویسی"، عنوان مطلب: "آموزش SQL"، [www.codestan.com](http://www.codestan.com)، دسترسی در تاریخ 2024/06/27.

- وب‌سایت "مرجع تخصصی برنامه‌نویسی"، عنوان مطلب: "آموزش طراحی سیستم‌های مدیریت مالی"، [www.itpro.ir](http://www.itpro.ir)، دسترسی در تاریخ 2024/06/27.

- وب‌سایت "آموزش‌های دانشگاهی"، عنوان مطلب: "مبانی برنامه‌نویسی وب"، [www.universitytutorials.ir](http://www.universitytutorials.ir)، دسترسی در تاریخ 2024/06/27.

- وب‌سایت "فرادرس"، عنوان مطلب: "آموزش PHP و MySQL"، [www.faradars.org](http://www.faradars.org)، دسترسی در تاریخ 2024/06/27.

- وب‌سایت "دانشجویار"، عنوان مطلب: "آموزش پروژه محور PHP"، [www.daneshjooyar.com](http://www.daneshjooyar.com)، دسترسی در تاریخ 2024/06/27.

- وب‌سایت "توتوریالز پلاس"، عنوان مطلب: "آموزش PHP و SQL"، [www.tutorialsplus.ir](http://www.tutorialsplus.ir)، دسترسی در تاریخ 2024/06/27.

3. مستندات پروژه:

- گزارش طراحی و پیاده‌سازی سیستم مدیریت شارژ ساختمان، تیم توسعه، 2024.

پیوست‌ها

لینک دانلود کامل پروژه در لینک زیر موجود می باشد :

<https://github.com/hamidkamizi/apartmentcrm>

**چکیده پروژه :**

این پروژه به طراحی و پیاده‌سازی سیستم مدیریت شارژ ساختمان با هدف بهبود فرآیندهای مالی و اداری ساکنان اختصاص دارد. سیستم شامل ماژول‌های مختلفی مانند مدیریت ساکنان، امکانات ساختمان، تابلو اعلانات، درخواست‌ها، قرار ملاقات‌ها و پرداخت‌ها می‌باشد. هدف اصلی این سیستم، فراهم کردن یک پلتفرم جامع و یکپارچه برای مدیریت بهتر و شفاف‌تر امور مالی و اطلاع‌رسانی به ساکنان است.