
بسم الله الرحمن الرحيم

چکیده

برای ثبت اطلاعات ورود و خروج افراد نیاز است قفلی بر روی درب نصب شود که دارای قابلیت مخابره کردن باشد. همچنان میبایست یک نرم‌افزاری در آن سو اجرا شود که بتوان این اطلاعات را پردازش کرده و به بهترین شکل نمایش دهد. همچنان میبایست این قفل و نرم‌افزار، اطلاعات افراد مطمئن را ذخیره کرده و از ورود افراد غیرمجاز جلوگیری کند. در عین حال نیز باید بتوان تنظیمات سیستم را با نرم‌افزار موجود بروزرسانی و تغییر داد.

کلید واژه: قفل الکترونیکی، برنامه اندرویدی، کنترل از راه دور، پیامک، اثرانگشت.

فهرست مطالب

عنوان **صفحه**

فصل اول

مقدمه

۱-۱- پیشگفتار

نیاز است برای امنیت بیشتر و کنترل عبور و مرور افراد، سیستمی طراحی شود که اطلاعات ورود و خروج افراد ثبت و ذخیره کند؛ این سیستم باید دارای قابلیت هویت‌شناسی باشد. برای اطلاع فوری، میبایست سیستم این داده‌ها را ارسال کرده و اطلاع دهد. همچنین قابلیتی داشته باشد که بتوان از راه دور مدیریت آن را انجام داد.

۱-۲- هدف پروژه

قرار است قفل الکترونیکی طراحی کنیم که از راه دور با کمک برنامه اندرویدی کنترل شود. این قفل میبایست ورود و خروج افراد را بوسیله پیامک اطلاع و در پایگاه داده برنامه اندرویدی ذخیره کند. اطلاعات افراد نیز در حافظه سیستم و همچنان در پایگاه داده برنامه اندرویدی نیز میبایست حفظ شود. این قفل قابلیت ورود با اثرانگشت و رمز عبور را دارد و باید بتوان کل تنظیمات سیستم را از طریق برنامه اندرویدی تغییر داد.

فصل دوم

ابعاد پروژه

۱-۲ حالت‌های مختلف عملکردی

این پروژه چندین حالت[1] های عملکردی دارد که رفتار سیستم را به چندین دسته تقسیم می‌کنند که در ادامه حالت‌های موجود را توضیح می‌دهیم

۱-۲-۱-حالت عادی

در حالت عادی[2]، سیستم برای باز شدن قفل، رمز عبور را از صفحه کلید دریافت می‌کند؛ همزمان ماژول[3] سیم‌کارت پیامک‌های دریافتی را پردازش می‌کند؛ در عین حال یک شمارنده نیز برای نمایش ثانیه‌های عبوری بر روی صفحه نمایش داده می‌شود. همچنان صفحه نمایش کلیدهای فرمان ورودی را نشان می‌دهد ولی به دلیل افزایش امنیت، بجای اعداد، کاراکتر ستاره نمایش داده خواهد شد. در این حالت برای فعالسازی ورودی اثرانگشت نیاز است کلید مشخصی فشار داده شود.

۱-۲-۲-حالت غیرفعال

در حالت غیرفعال[4]، ورودی از صفحه کلید و اثرانگشت غیرفعال می‌باشد. ماژول سیم‌کارت همچنان پیامک‌های دریافتی را پردازش می‌کند. صفحه نمایش فقط متن «حالت غیرفعال» را نشان می‌دهد. همچنان بصورت همزمان شمارنده سیستم در پس‌زمینه مشغول کار می‌باشد.

۱-۲-۳-حالت دریافت تنظیمات از صفحه کلید

در حالت دریافت تنظیمات از صفحه کلید[5]، صفحه نمایش منوی تنظیمات را نشان خواهد داد؛ با صفحه کلید می‌توان یک گزینه را انتخاب کرده و مقدار آن را تغییر داد. پس از تغییر تنظیمات، یک پیامک اعلان برای برنامه اندرویدی ارسال می‌شود.

۱-۲-۴-حالت ایجاد کاربر

در حالت ایجاد کاربر، صفحه نمایش در خدمت ماژول اثرانگشت بوده و پیغام های موردنیاز را نشان می‌دهد. ماژول اثرانگشت نیز در هر مرحله منتظر تایید آماده بودن کاربر از طریق صفحه کلید می‌باشد. در این حالت ماژول اثرانگشت، یک کاربر جدید به سیستم معرفی خواهد کرد. بعد از نمایش شناسه جدید

[1] Mode
[2] Normal Mode
[3] Module
[4] Disable Mode
[5] Get Settings from Keypad

کاربر، نیاز است با استفاده از برنامه اندرویدی نام کاربر را برای سخت‌افزار موجود ارسال کند. برای امنیت بیشتر، حتما نیاز است در ابتدا رمز عبور تایید شود.

شکل ۱-۰: دیاگرام نحوه اضافه کردن کاربر

۵-۲-۱-حالت خواندن کاربر

در حالت عادی پس از اینکه ماژول اثرانگشت فعال شود، سیستم به حالت خواندن کاربر می‌رود و اثرانگشت کاربر را گرفته و در صورت وجود قفل باز می‌شود و در غیر اینصورت با نمایش پیغامی اجازه ورود داده نمی‌شود. در هر دو حالت پیامک اعلان برای برنامه اندرویدی ارسال خواهد شد.

۶-۲-۱-حالت بروز رسانی کاربر

این قابلیت وجود دارد که با استفاده از حالت بروز رسانی کاربر، اثرانگشت کاربر ثبت شده موجود را تغییر و بروز رسانی کرد. نیاز است شناسه کاربر موردنظر و رمز عبور در ابتدا وارد شود. پس از موفقیت‌آمیز بودن این فرایند، پیامک اعلان برای برنامه اندرویدی ارسال خواهد شد.

۷-۲-۱-حالت حذف کاربر

در این حالت نیز، با دریافت رمز عبور و شناسه کاربر مورد نظر، مشخصات اثرانگشت وی از سیستم حذف خواهد شد. در این حاالت نیز پس از موفق بودن عمل، پیامک اعلان برای برنامه اندرویدی ارسال خواهد شد.

۸-۲-۱-حالت تنظیم مجدد

در این حالت همه تنظیمات و کاربران پاک شده و سیستم اصطلاحا به تنظیمات کارخانه برمیگردد.

۲-۲- چالش های پروژه

در طراحی و پیاده‌سازی چنین سیستمی، چالش‌های بسیاری وجود دارد که در ادامه مهمترین آنها ذکر شده است.

۱-۲-۲-هم‌زمانی

روش برنامه‌نویسی معمول که تمامی دستورات در یک حلقه بی‌نهایت قرار می‌گیرند، اصطلاحا Super Loop می‌نامند. این روش برای بسیاری از کدهای ساده پاسخگو است؛ اما در کدهای پیچیده، مواقعی پیش می‌آید که نیاز است پردازنده بصورت همزمان چند کار را انجام دهد؛ و بخش‌هایی مانند تایمر و وقفه[1] هم کمکی نکند. در این مواقع روش معرفی شده دیگر پاسخگو نخواهد بود.

[1] Intterrupt

راه‌حل رفع این محدودیت استفاده از سیستم عامل بلادرنگ[1] است. سیستم عامل بلادرنگ، یک سیستم‌عامل برای برنامه‌های محاسباتی بلادرنگ است که داده‌ها و رویدادهایی را پردازش می‌کند که محدودیت‌های زمانی مشخصی دارند. در واقع RTOS یک مولفه نرم افزاری است که باعث می‌شود پردازنده با سرعت بسیار بین وظایف جابجا شود. هر وظیفه[2] در داخل خود یک حلقه بی‌نهایت دارد و وظایف تقریبا بصورت همزمان با یکدیگر اجرا می‌شوند. همچنین میتوان با استفاده از این سیستم‌عامل برای هر وظیفه اولویت انتخاب کرد؛ به این صورت که اگر وظیفه دارای اولویت بالاتر سر برسد. سیستم عامل ابتدا وظیفه دارای اولویت را انجام می‌دهد و سپس به ادامه وظایف می‌پردازد.

۲-۲-۲-مقدار رم

پردازنده‌ها معمولا دارای حافظه داخلی کمی هستند که برای ذخیره داده‌ها و کدهای برنامه استفاده می‌شود. چالش مقدار رم[3] در برنامه‌نویسی پردازنده این است که باید با حافظه محدودی که در دسترس است، برنامه‌های کارآمد و بهینه‌ای طراحی کرد. این چالش ممکن است باعث شود که برنامه‌نویس باید از الگوریتم‌های ساده‌تر و کم حجم‌تر استفاده کند، یا برخی از قابلیت‌ها و ویژگی‌های برنامه را حذف یا کاهش دهد.

برای این چالش راه‌حل کاملی وجود نداشته و فقط میبایست سعی شود تا جایی که امکان دارد برنامه‌ها را جوری نوشت که کمترین میزان رم اشغال شود؛ مانند مثال میتوان سعی کرد از روش‌های زیر استفاده کرد:

- استفاده از داده‌ساختارهای مناسب و کارآمد
- استفاده از حافظه مشترک بین وظایف
- استفاده کمتر از توابع تودرتو یا بازگشتی
- و ...

۳-۲-۲-امنیت

امنیت در ارسال و دریافت پیامک بسیار حائز اهمیت است. شماره تلفن برنامه اندرویدی و ماژول سیمکارت و رمز عبور نیز می‌بایست از دیگران مخفی و غیرقابل دسترس باشد. زیرا امکان تغییر تنظیمات، تغییر رمز عبور و ... با دسترسی به این اطلاعات وجود دارد.

[1] Real-time operating system (RTOS)
[2] Task
[3] Random-access memory

راه‌حل حل این مشکل، رمزنگاری کردن متن پیامک‌ها و ذخیره کردن شماره تلفن‌ها و رمز عبور بصورت هش[1] شده است.

۲-۲-۴-پردازش متن پیامک

نیاز است که متن پیامک‌ها توسط خود سیستم پردازش شود. در عین حال باید جوری باشد که هم در پردازنده و هم در برنامه موبایلی مشکلی بابت پردازش آن وجود نداشته باشد.

راه‌حل این مسئله این است که متن پیامک‌ها بصورت یک فرمت استاندارد نوشته شود. فرمت استاندارد جیسون[2] می‌تواند برای این مورد بهترین انتخاب باشد.

۲-۳- مفاهیم استفاده شده در پروژه

چندین مفاهیم مختلف در این پروژه استفاده شده که در ادامه به تفصیل توضیح داده خواهند شد.

۲-۳-۱-سیستم‌عامل اندروید

اندروید[3]، یک سیستم‌عامل موبایل بر پایه نسخه اصلاح‌شده‌ای از هسته لینوکس[4] و دیگر نرم‌افزارهای متن‌باز[5] می‌باشد که در دستگاه‌های لمسی مانند تلفن هوشمند و تبلت استفاده می‌شود. اندروید از سال ۲۰۱۱ پرکاربردترین سیستم‌عامل موبایل و از سال ۲۰۱۳ پرکاربردترین سیستم‌عامل برای تبلت‌ها بوده است

۲-۳-۲-سیستم‌عامل FreeRTOS

سیستم‌عامل FreeRTOS، یک هسته سیستم‌عامل بی‌درنگ برای سامانه‌های نهفته است و از ۳۵ نوع معماری پلت‌فرم میکروکنترلر پشتیبانی می‌کند. این سیستم‌عامل، طراحی شده‌است تا ساده و کوچک باشد. هسته اصلی تنها از ۳ فایل با فرمت C تشکیل شده‌است. به منظور سادگی و سهولت در تغییر عمدتاً به زبان C نوشته شده‌است، اما تعداد کمی توابع به زبان ماشین (اسمبلی) نیز در نقاطی از برنامه که مورد نیاز بوده استفاده شده‌است.

[1] Hash
[2] JSON
[3] Android
[4] Linux
[5] Open-Source

سیستم فری‌آرتی‌اواس روش‌هایی را برای چندریسه‌ای (رشته‌های اجرایی همزمان)، چندوظیفگی، نشان‌برها (سمافورها) و شمارنده‌های نرم‌افزاری فراهم می‌آورد. یک حالت بدون تیک (بدون وقفه شمارنده) نیز برای کاربردهایی با مصرف انرژی کم تهیه شده‌است. اولویت‌بندی نخ‌ها (رشته‌های پردازشی) نیز پشتیبانی می‌شود. [۱]

۳-۳-۲-پایگاه داده

در علم کامپیوتر، پایگاه داده[1] مجموعه‌ای سازمان‌یافته از داده‌ها یا نوعی ذخیره‌سازی داده است که مبتنی بر استفاده از سیستم مدیریت پایگاه داده (DBMS) که نرم‌افزاری است که با کاربران نهایی، برنامه‌های کاربردی و خود پایگاه داده برای جمع‌آوری و تجزیه و تحلیل داده‌ها در تعامل است. طراحی پایگاه‌های داده شامل تکنیک‌های رسمی و ملاحظات عملی، از جمله مدل‌سازی داده، نمایش و ذخیره داده‌های کارآمد، زبان‌های پرس و جو، امنیت و حریم خصوصی داده‌های حساس، و مسائل محاسباتی توزیع‌شده، از جمله پشتیبانی از دسترسی همزمان و تحمل خطا است.

دانشمندان کامپیوتر ممکن است سیستم های مدیریت پایگاه داده را بر اساس مدل های پایگاه داده‌ای که پشتیبانی می‌کنند، طبقه بندی کنند. پایگاه داده‌های رابطه‌ای در دهه ۱۹۸۰ غالب شدند که داده‌ها را به صورت ردیف و ستون در یک سری جداول مدل می‌کنند و اکثریت قریب به اتفاق از SQL برای نوشتن و جستجوی داده ها استفاده می‌کنند. در دهه ۲۰۰۰، پایگاه‌های اطلاعاتی غیر رابطه‌ای محبوب شدند که در مجموع به آنها NoSQL می‌گویند، زیرا از زبان‌های پرس و جوی مختلفی استفاده می‌کنند.

۴-۳-۲-داده جیسون

نام جیسون معادل اختصاری «نمادگذاری اشیا در جاوا اسکریپت[2]» است. جیسون، یک استاندارد باز متنیِ سبک برای انتقال داده‌ها بصورت کلید-مقدار[3] است به گونه‌ای که برای انسان نیز خوانا باشد. این یک فرمت داده رایج با کاربردهای متنوع در مبادله الکترونیکی داده‌ها، از جمله برنامه‌های کاربردی وب با سرورها است. جیسون یک فرمت داده مستقل از زبان است. جیسون از جاوا اسکریپت مشتق شده است، اما بسیاری از زبان های برنامه‌نویسی مدرن شامل کدهایی برای تولید و تجزیه داده‌های با فرمت جیسون هستند.[۲]

[1] DataBase
[2] JavaScript Object Notation
[3] Key-Value

۵-۳-۲-رمزنگاری داده

رمزنگاری[1] استفاده از روش‌های ریاضی، برای برقراری امنیت اطلاعات است. در اصل، رمزنگاری دانش تغییر دادن متن پیام یا اطلاعات به کمک کلید رمز و با استفاده از یک الگوریتم رمز است. به صورتی که فقط شخصی که از کلید و الگوریتم آگاه است، می‌تواند اطلاعات اصلی را از اطلاعات رمزگذاری استخراج کند و شخصی که از یکی یا هر دوی آن‌ها آگاهی ندارد نمی‌تواند به اطلاعات دسترسی پیدا کند.[۳]

استاندارد رمزنگاری پیشرفته[2] یا به اختصار AES مشخصه‌ای برای رمزنگاری داده‌های دیجیتال است که در سال ۲۰۰۱ توسط مؤسسه ملی فناوری و استانداردهای ایالات متحده ایجاد شد. این استاندارد توسط دولت ایالات متحده پذیرفته شده و اکنون در سراسر جهان استفاده می‌شود. الگوریتم AES یک الگوریتم کلید متقارن است؛ بدین معنی که از یک کلید یکسان برای رمزنگاری و رمزگشایی استفاده می‌شود.[۴]

تابع هش رمزنگاری، یک الگوریتم ریاضی است که داده‌هایی با اندازه اختیاری (به آن «پیام» گفته می‌شود) را به یک آرایه بیتی با اندازه ثابت (به آن «درهَمَک»، «هش» یا «چکیده پیام» گفته می‌شود) نگاشت می‌دهد. ویژگی مهم این تابع «یک‌طرفه بودن» آن است و این یعنی تابعی است که وارون‌سازی آن از نظر عملی اجراپذیر نیست. توابع درهمک‌ساز رمزنگاری، کاربردهای زیادی در امنیت اطلاعات دارند، به ویژه آن‌ها در امضای دیجیتال، کدهای احرازهویت پیام، و دیگر حالت‌های احرازهویت کاربرد دارند. [۵]

اس‌اچ‌ای-۳[3] (الگوریتم هش ایمن ۳[4]) توسط مؤسسه ملی فناوری و استانداردها[5] در ۵ آگوست ۲۰۱۵ منتشر شد. اس‌اچ‌ای-۳ خروجی‌های ۲۲۴، ۲۵۶، ۳۸۴ و ۵۱۲ بیت را ارائه می‌دهد.[۶]

۶-۳-۲-ارتباط سریال

ارتباط سریال در علم مخابرات و کامپیوتر به فرایند ارسال داده‌ها به وسیله یک بیت در واحد زمان و به ترتیب در چند کانال ارتباطی گفته می‌شود. در برابر این نوع از ارسال، ارسال موازی قرار دارد که چندین بیت را بطور همزمان در یک لینک کانال موازی می‌فرستد. در این پروژه از ارتباط سریال آسنکرون[6] استفاده شده که این در ارتباط هنگام ارسال بیت‌ها، علاوه بر دیتای اصلی، تعدادی بیت کنترلی نیز با آن ارسال می‌شود؛[۷] این نوع از ارتباط نیازی به اتصال پایه کلاک نداشته اما دو طرف ارتباط باید از عرض هر بیت اطلاع داشته باشند.

[1] Cryptography
[2] Advanced Encryption Standard
[3] SHA-3
[4] Secure Hash Algorithm 3
[5] National Institute of Standards and Technology (NIST)
[6] Universal Asynchronous Receiver And Transmitter

۷-۳-۲-ارتباط I2C

پروتکل ارتباط آی‌سی، یک گذرگاه ارتباط سریال همگام، چند پیرو[1]، چند رهبر[2]، راه‌گزینی بسته کوچک[3]، تک-انتهایی[4] است. پروتکل آی۲سی، در واقع ادغام شده از بهترین ویژگی‌های SPI و UART می‌باشد. توسط آی۲سی امکان اتصال چند پیرو به یک رهبر (مانند SPI) یا استفاده از چند رهبر برای کنترل یک یا چند پیرو وجود دارد. این ویژگی زمانی که شما می‌خواهید از چند ریزکنترلگر برای ارسال داده به یک کارت حافظه یا نمایش بر روی LCD استفاده کنید، بسیار مناسب می‌باشد. مانند روش یوای‌آرتی در آی۲سی نیز از دوسیم برای انتقال اطلاعات استفاده می‌شود.

۸-۳-۲-ساعت پازیکس

ساعت پازیکس[5]، یکی از مقیاس اندازه‌گیری زمان آنی است. این عدد تعداد ثانیه‌ها از ساعت ۰۰:۰۰:۰۰ ساعت هماهنگ جهانی اول ژانویه ۱۹۷۰ است. (شامل ثانیه‌های کبیسه نمی‌شود) همچنین برای زمان‌های قبل از اول ژانویه از اعداد منفی استفاده می‌شود.

این استاندارد امکان همسان‌سازی و ارتباط آسان‌تر بین محیط‌های نرم‌افزاری مختلف را بوجود می‌آورد.

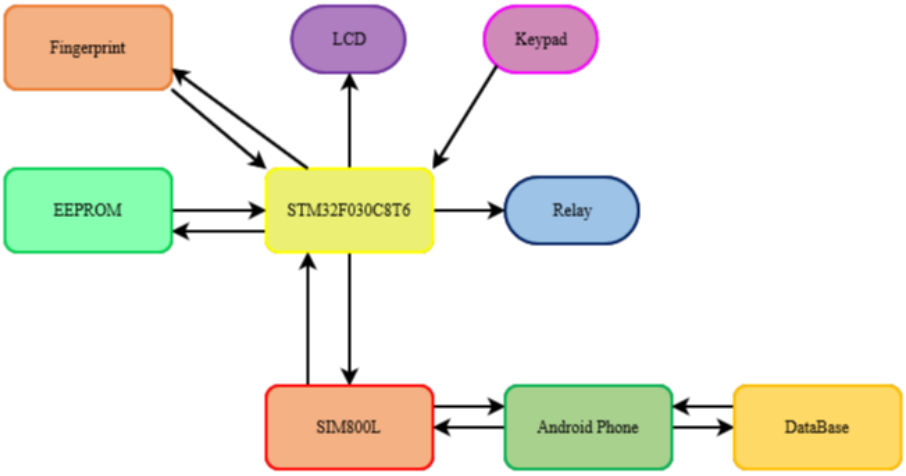
[1] Slave
[2] Master
[3] Packet switching
[4] Signal-ended
[5] Posix Time

فصل سوم

عملکرد اجزای سیستم

۳-۱ مقدمه

تصویر زیر دیاگرام کلی سیستم را نشان می‌دهد. در این فصل، اجزای سیستم مورد بررسی قرار می‌گیرند.



شکل ۱-۰: دیاگرام کلی سیستم

۳-۲- اجزای سیستم

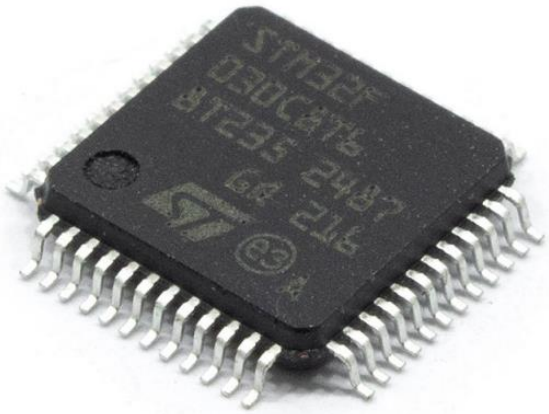
در ادامه اجزای سیستم و عملکرد آنها توضیح داده می‌شوند.

۳-۲-۱-پردازنده STM32f030C8T6

این پردازنده دارای هسته 32 RISC بیتی Arm® Cortex®-M0 با کارایی بالا است که با فرکانس ۴۸ مگاهرتز، حافظه‌های تعبیه‌شده با سرعت بالا (۴۸ کیلوبایت حافظه فلش و ۸ کیلوبایت حافظه SRAM) و طیف وسیعی از تجهیزات جانبی و ورودی/خروجی های پیشرفته کار می‌کند. این دستگاه رابط‌های ارتباطی استاندارد (دو I2C، دو SPI و دو USART)، یک ADC 12 بیتی، هفت تایمر ۱۶ بیتی همه منظوره و یک تایمر PWM با کنترل پیشرفته را ارائه می‌دهد.

این پردازنده، در محدوده دمایی -۴۰ تا +۸۵ درجه سانتیگراد از منبع تغذیه ۲.۴ تا ۳.۶ ولت کار می‌کند. مجموعه‌ای جامع از حالت‌های صرفه‌جویی در مصرف انرژی، امکان طراحی برنامه‌های کم مصرف را فراهم می‌کند. [۸]

این پردازنده وظیفه کنترل و مدیریت همزمان بقیه اجزای سیستم را دارد. اطلاعات را از ماژول‌های سیمکارت، اثرانگشت، صفحه کلید، حافظه EEPROM گرفته و پس از پردازش و اتخاذ تصمیات لازم، خروجی متناسب را در صورت لزوم برای ماژول‌های صفحه نمایش، رله، ماژول سیمکارت و حافظه EEPROM ارسال می‌کند.

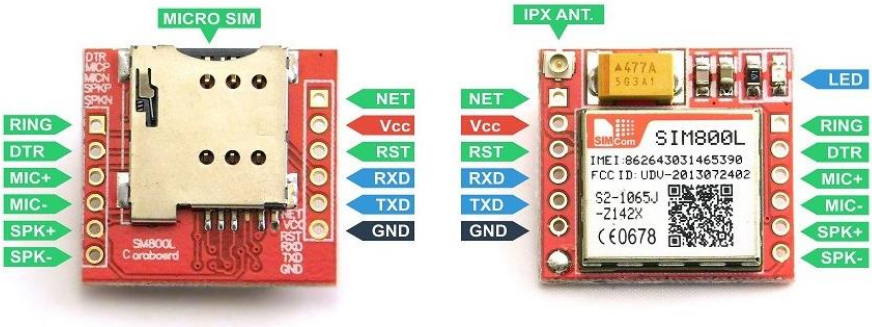


شکل ۲-۰: پردازنده STM32F030C8T6

۲-۲-۳-ماژول سیمکارت SIM800L

ماژول SIM800L، در واقع مدل مینیاتوری یک مودم است. با استفاده از این ماژول می‌توان به تمامی امکانات یک تلفن همراه معمولی نظیر ارسال پیامک، برقراری یا دریافت تماس تلفنی و اتصال به اینترنت دست پیدا کرد. یک اسلات به منظور قرارگیری سیمکارت در پشت ماژول وجود دارد.

این ماژول وظیفه ارسال و دریافت پیامک‌ها را دارد. با استفاده از ارتباط سریال، اطلاعات را از پردازنده گرفته و برای برنامه اندرویدی ارسال کرده یا پیامک دریافتی از برنامه اندرویدی را به اطلاع پردازنده می‌رساند.



شکل ۳-۰: ماژول سیمکارت SIM800L

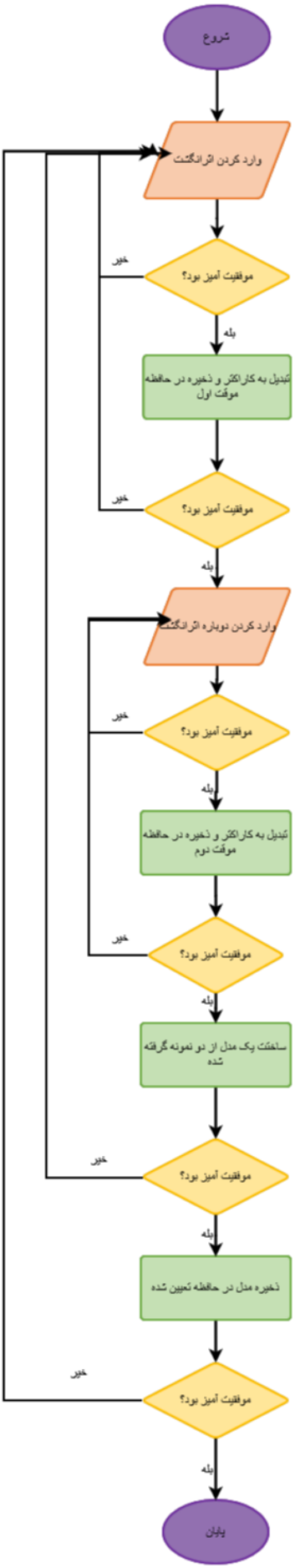
۳-۲-۳-ماژول اثرانگشت R308

ماژول سنسور اثرانگشت R308 ماژولی با یک سنسور نوری اثرانگشت، پردازنده DSP سرعت بالا، الگوریتم تطبیق با کارایی بالا و تراشه فلش با ظرفیت بالاست که بر مبنای پردازش تصویر اثرانگشت و تطبیق و جستجو در حافظه و انجام عملکردهای مورد نظر، کار می‌کند.

در این پروژه وظیفه خواندن اثرانگشت، ذخیره و بازیابی آنها را دارد. این ماژول با رابط سریال به پردازنده وصل شده و نتیجه عملیات را به اطلاع پردازنده می‌رساند.

شکل ۴-۰: ماژول اثرانگشت R308

مراحل ثبت اثرانگشت در این ماژول را در فلوچارت زیر میبینید:


شروع
وارد کردن انرادگشت
موفقیت آمیز بود؟
خیر
بله
تبدیل به کاراکتر و ذخیره در حافظه موقت اول
موفقیت آمیز بود؟
خیر
بله
وارد کردن دوباره انرادگشت
موفقیت آمیز بود؟
خیر
بله
تبدیل به کاراکتر و ذخیره در حافظه موقت دوم
موفقیت آمیز بود؟
خیر
بله
ساختن یک مدل از دو نمونه گرفته شده
موفقیت آمیز بود؟
خیر
بله
ذخیره مدل در حافظه تعیین شده
موفقیت آمیز بود؟
خیر
بله
پایان

۳-۲-۴-صفحه نمایش کاراکتری

صفحه نمایش LCD، ابزاری برای نمایش اطلاعاتی است که شامل حروف، اعداد و همچنین برخی کاراکترهای گرافیکی می‌شود. همراه با آی‌سی کنترلر و مدارهای جانبی‌اش و عموما با لامپ پشت صفحه[1] در یک بسته از پیش ساخته و در دو نوع کاراکتری و گرافیکی عرضه می‌شود. LCD داری یک کنترلر است که با فرستادن اطلاعات به آن، این اطلاعات را در صفحه‌ای که عموما به چند سطر و ستون تقسیم شده نمایش می‌دهد.

این ماژول در این پروژه پیغام‌هایی که نیاز به نمایش به کاربر دارند را از پردازنده گرفته و نمایش می‌دهد.

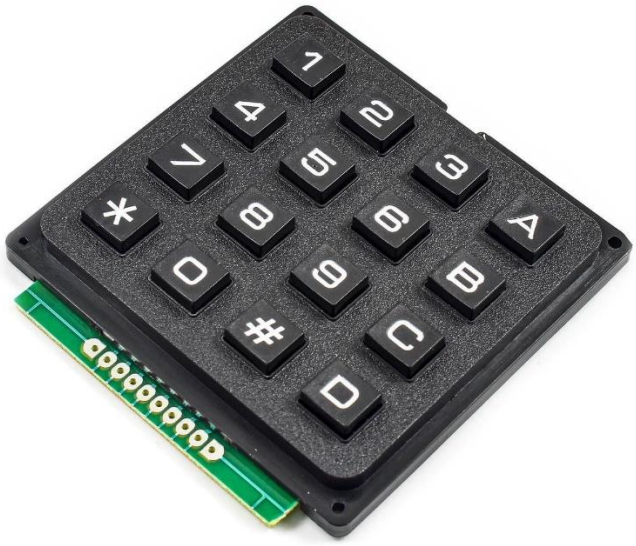
شکل ۵-۰: صفحه نمایش کاراکتری

۳-۲-۵-صفحه کلید ماتریسی

صفحه کلید ماتریسی[2] متشکل از تعداد زیادی کلید است که بصورت ماتریسی روی هم قرار گرفته‌اند که با فشار دادن بر روی هر کلید، فرمانی ارسال می‌شود.

در این پروژه، پردازنده با اسکن متوالی و سریع کلیدها، فرمان کاربر را دریافت کرده و مطابق با آن عکس‌العملی را انجام می‌دهد.

[1] Back Light
[2] Matrix Keypad

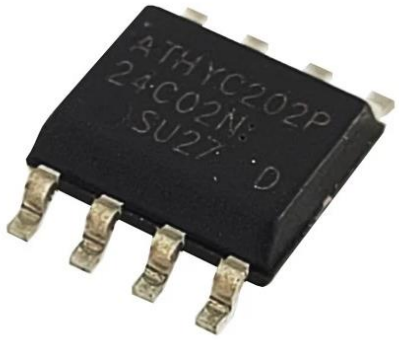


شکل ۶-۰: صفحه کلید ماتریسی

۶-۲-۳-حافظه EEPROM

EEPROM مخفف حافظه فقط-خواندنی قابل‌برنامه‌ریزی قابل‌پاک‌کردن به‌طور الکتریکی[1]، گونه‌ای از حافظه‌های غیرفرّار است که در دستگاه‌ها و رایانه‌ها برای ذخیره حجم پایینی از اطلاعات استفاده می‌شود که نبایستی با قطع برق از دست بروند. اطلاعاتی همچون تنظیمات دستگاه.[۹]

وظیفه این حافظه در این پروژه این است که لیست اسامی کاربران و تنظیمات سیستم را در خود ذخیره و بازیابی کند. این ماژول با پروتکل I2C با پردازنده تبادل اطلاعات می‌کند.



شکل ۷-۰: حافظه EEPROM

[1] Electrically Erasable Programmable Read-Only Memory

۷-۲-۳-ماژول ساعت DS1307

ماژول ساعت DS1307 برای محاسبه و نگهداری زمان واقعی[1] استفاده می‌شود. این ماژول از پروتکل I2C پشتیبانی می‌کند و از یک باتری سکه‌ای لیتیومی CR1225 یا CR2032 استفاده می‌کند. این ماژول اطلاعات ثانیه، دقیقه، ساعت، روز، تاریخ، ماه و سال را در حافظه ۳۲ کیلوبایتی خود نگه می‌دارد. باتری سکه‌ای این ماژول برای حفظ اطلاعات ساعت و تاریخ در مواقع قطع تغذیه استفاده می‌شود.

وظیفه این ماژول در سیستم این است که زمان حال را نگهداری کرده و در مواقع مورد نیاز به سیستم اطلاع دهد.

شکل ۸-۰: ماژول ساعت DS1307

۸-۲-۳-رله

رله[2] نوعی کلید الکتریکی سریع یا بی‌درنگ است که با هدایت یک مدار الکتریکی دیگر باز و بسته می‌شود. رله دو پایه نرمال بسته و نرمال باز و دو پایه سیم پیچ و یک پایه com دارد. به این صورت که نرمال بسته همیشه به کام وصل است اما اگر ولتاژ سیم پیچ تغییر کند نرمال باز به کام وصل و اگر دوباره ولتاژ به حالت اول برگردد نرمال بسته به کام وصل می‌شود.[۱۰]

وظیفه رله در این پروژه، جداسازی مدار کنترلی از مدار اصلی عملگر برای باز کردن قفل است که باعث می‌شود جریان معقولی از این سیستم الکترونیکی کشیده شود و به قطعات آسیبی نرسد. رله فرمان باز کردن را از پردازنده می‌گیرد.

[1] Real Time Clock
[2] Relay

شکل ۹-۰: رله

۹-۲-۳-برنامه اندرویدی

برنامه‌های اندرویدی که عملکرد دستگاه‌ها را گسترش می‌دهند، با استفاده از کیت توسعه نرم‌افزار اندروید[1] و اغلب زبان برنامه‌نویسی کاتلین[2] که جایگزین جاوا می‌شود، نوشته می‌شوند. همچنان می‌توان از زبان‌های جاوا، سی‌شارپ، پایتون، جاوااسکریپت و ... برای توسعه برنامه اندرویدی استفاده کرد.

این پروژه، برنامه اندرویدی برای ارتباط از راه دور و مدیریت سیستم دارد که قابلیت های زیر را داراست:

- قابلیت نمایش ورود کاربران
- قابلیت نمایش تغییرات تنظیمات از صفحه کلید
- قابلیت نمایش اضافه شدن، بروز رسانی شدن یا حذف شدن کاربران
- قابلیت ایجاد تغییرات در تنظیمات سیستم
- قابلیت تغییر حالت سیستم

این برنامه با استفاده از پیامک، اطلاعات فوق را از ماژول سیمکارت دریافت کرده و فرامین جدید را به این ماژول می‌فرستد تا بر روی سیستم اعمال شود. اطلاعات فوق، در پایگاه داده در خود برنامه ذخیره می‌شوند.

۱۰-۲-۳- پایگاه داده

نیاز است در این پروژه اطلاعات کاربران، سابقه ورود و خروج کاربران و تنظیمات موجود سیستم ذخیره شده و به بهترین نحو بازیابی شوند. سعی شده برای اطلاعات کاربران و ورود و خروج کاربران از پایگاه داده SQL مانند Sqlite و برای ذخیره تنظیمات سیستم از پایگاه داده NoSQL مانند Shared Preference استفاده شود.

برای دیتابیس‌های SQL میبایست با زبان SQL آشنا بوده و برای هر مقدار ابتدا مدلسازی کرده و سپس برای هر مدل یک جدول رابطه‌ای ساخت.

پایگاه داده NoSQL نیازی به مدلسازی و ایجاد جدول ندارد.

[1] Android SDK
[2] Kotlin

فصل چهارم

پیاده‌سازی سیستم

۱-۴- کدنویسی سخت‌افزار

این بخش مربوط به نحوه کدنویسی پردازنده با زبان سی است که به تفکیک اجزای مختلف ارائه می‌شود.

۱-۱-۴-صفحه کلید

کدهای نوشته شده برای مدیریت فرامین دریافتی از صفحه کلید از کتابخانه KeyPad استفاده شده است. در این کتابخانه برای صفحه کلید یک ساختمان[1] ساخته شده که تعداد سطر، ستون و آخرین کلید دریافتی را در خود ذخیره می‌کند.

ابتدا میبایست پین و پورت‌های سطر و ستون‌ها را در فایل KeyPadConfig.h وارد شود.

برای شروع استفاده از صفحه کلید نیاز است ابتدا تابع `void KeyPad_Init(void)` فراخوانی شود؛ این تابع باعث می‌شود پین‌های ستون‌ها در حالت خروجی و دارای مقدار یک منطقی شود. همچنین پین‌های سطرها را بصورت ورودی و پول‌آپ[2] شده، تنظیم می‌شود.

تابع `uint16_t KeyPad_WaitForKey(uint32_t Timeout_ms)` یک عدد بعنوان زمان می‌گیرد که طی این زمان صفحه کلید را اسکن کرده و اگر کلیدی فشار داده شده باشد یا زمان اتمام یابد، از تابع خروج می‌کند. اگر زمان وارد شده صفر باشد، درون حلقه بی‌نهایت منتظر کلید فشار داده شده می‌ماند.

با استفاده از تابع `char KeyPad_WaitForKeyGetChar(uint32_t Timeout_ms)` می‌توان خروجی تابع قبلی که به عدد است را به کاراکتر تبدیل کرد؛ این کار توسط دستور `switch` که یک نمونه شروط پی در پی است، انجام می‌شود.

```
typedef struct {
  uint8_t  ColumnSize;
  uint8_t  RowSize;
  uint16_t LastKey;

} KeyPad_t;

void     KeyPad_Init(void);
uint16_t KeyPad_WaitForKey(uint32_t Timeout_ms);
char     KeyPad_WaitForKeyGetChar(uint32_t Timeout_ms);
```

[1] Struct
[2] Pull-Up

۲-۱-۴-صفحه نمایش

برای راه‌اندازی صفحه نمایش و ارسال دستورات لازم برای نمایش پیغام‌ها، از کتابخانه LCD استفاده شده است. این کتابخانه نیز برای مدیریت صفحه نمایش‌های موجود برای هر کدام یک متغیر از نوع ساختمان ساخته که پین و پورت‌های پایه‌های مربوطه و حالت صفحه نمایش در آن ذخیره می‌شود.

برای شروع این تابع

```
Lcd_HandleTypeDef Lcd_create(
    Lcd_PortType port[], Lcd_PinType pin[],
    Lcd_PortType rs_port, Lcd_PinType rs_pin,
    Lcd_PortType en_port, Lcd_PinType en_pin,
    Lcd_ModeTypeDef mode
)
```

یک متغیر LCD ساخته و آن را با توجه به مقادیر ورودی مقداردهی می‌کند. سپس این متغیر را به تابع `void Lcd_init(Lcd_HandleTypeDef *lcd)` ارسال می‌کند تا دستورات لازم جهت راه‌اندازی برای ماژول صفحه نمایش فرستاده شود.

سپس می‌توان با استفاده از تابع

```
void Lcd_string(Lcd_HandleTypeDef *lcd, char *string)
```

متن مورد نظر را جهت نمایش ارسال کرد. جهت تغییر مکان مکان‌نما میتوان از تابع

```
void Lcd_cursor(Lcd_HandleTypeDef *lcd, uint8_t row, uint8_t col)
```

استفاده کرد.

همچنین تابع `void Lcd_clear(Lcd_HandleTypeDef *lcd)` صفحه نمایش را پاک کرده و مکان‌نما را به ابتدای خط اول می‌فرستد.

```
typedef struct {
  Lcd_PortType *data_port;
  Lcd_PinType  *data_pin;

  Lcd_PortType rs_port;
  Lcd_PinType  rs_pin;

  Lcd_PortType en_port;
  Lcd_PinType  en_pin;

  Lcd_ModeTypeDef mode;

} Lcd_HandleTypeDef;

void              Lcd_init(Lcd_HandleTypeDef *lcd);
void              Lcd_string(Lcd_HandleTypeDef *lcd, char *string);
void              Lcd_cursor(Lcd_HandleTypeDef *lcd, uint8_t row, uint8_t
col);
Lcd_HandleTypeDef Lcd_create(
```

```
    Lcd_PortType port[], Lcd_PinType pin[], Lcd_PortType rs_port,
Lcd_PinType rs_pin,
    Lcd_PortType en_port, Lcd_PinType en_pin, Lcd_ModeTypeDef mode
);
void Lcd_clear(Lcd_HandleTypeDef *lcd);
```

۳-۱-۴-ماژول اثرانگشت

ماژول اثرانگشت با ارتباط سریال کار می‌کند. کافیست ارتباط سریال را با نرخ ۵۷۶۰۰ راه‌اندازی و از توابع کمکی نوشته شده استفاده کرد. این توابع داده‌هایی متناسب با پروتکل این ماژول ارسال و جواب را پردازش می‌کنند

- برای فعالسازی ماژول ابتدا باید تابع `char r308_verifypassword(void)` جهت تایید رمز عبور فراخوانی شود. اگر خروجی برابر `FINGERPRINT_OK` باشد، ماژول فعال می‌شود.
- تابع `char r308_getimage(void)` اثرانگشت را می‌گیرد.
- تابع `char r308_genchar(char id)` اثرانگشت را به کاراکتر تبدیل و طبق شناسه در حافظه موقت ذخیره می‌کند.
- تابع `char r308_regmodel(void)` یک مدل از روی کاراکترهای اثرانگشت ذخیره شده در حافظه موقت می‌سازد.
- تابع `char r308_store(char id)` مدل ساخته شده را طبق شناسه داده شده در حافظه اصلی ذخیره می‌کند.
- تابع `uint16_t r308_search(void)` اثرانگشت موجود در حافظه موقت را در حافظه اصلی جستجو می‌کند و اگر خروجی برابر `FINGERPRINT_OK` شود، شناسه کاربر موردنظر برگشت داده می‌شود.
- تابع `char r308_deletechar(int id)` اثرانگشت موجود در شناسه داده شده را پاک می‌کند.
- تابع `char r308_empty(void)` همه اثرانگشت‌های موجود در حافظه اصلی را پاک می‌کند.

```
#define FINGERPRINT_OK              0x00 // Command execution is complete
#define FINGERPRINT_PACKETRECIEVEERR 0x01 // Error when receiving data
package
#define FINGERPRINT_NOFINGER        0x02 // No finger on the sensor (can't
detect finger)
#define FINGERPRINT_IMAGEFAIL       0x03 // Failed to collect finger
#define FINGERPRINT_IMAGEMESS \
```

```
  0x06 // Failed to generate character file due to the over-disorderly
fingerprint image
#define FINGERPRINT_FEATUREFAIL \
  0x07 // Failed to generate character file due to lackness of character
point or over-smallness
        // of fingerprint image
#define FINGERPRINT_NOTFOUND       0x09 // Failed to find matching finger
#define FINGERPRINT_DELETEFAIL     0x10 // Failed to delete the template
#define FINGERPRINT_DBCLEARFAIL    0x11 // Failed to clear finger library
#define FINGERPRINT_ENROLLMISMATCH 0x0A // Failed to combine the character
files.
#define FINGERPRINT_BADLOCATION    0x0B // Addressed PageID is beyond the
finger library
#define FINGERPRINT_WRONGPASSWORD  0x13 // Wrong password
#define FINGERPRINT_FLASHERR       0x18 // Error when writing flash

char     r308_verifypassword(void);
char     r308_getimage(void);
char     r308_genchar(char id);
char     r308_regmodel(void);
char     r308_store(char id);
uint16_t r308_search(void);
char     r308_deletechar(int id);
char     r308_empty(void);
```

۴-۱-۴-حافظه EEPROM

برای استفاده از حافظه EEPROM از کتابخانه ee24 استفاده می‌شود. این حافظه با استفاده از اتصال i2c با پردازنده صحبت می‌کند.

در ابتدا نیاز است نوع ماژول مورد استفاده را در فایل ee24Config.h تعیین شود. سپس برای ارسال دستورات نیاز است متغیر `EE24_HandleTypeDef ee24` ساخته و با استفاده از تابع

```
bool EE24_Init(EE24_HandleTypeDef *Handle, I2C_HandleTypeDef *HI2c, uint8_t
I2CAddress)
```

مقداردهی شود.

حال میتوان با استفاده از تابع

```
bool EE24_Write(
    EE24_HandleTypeDef *Handle, uint32_t Address, uint8_t *Data, size_t Len,
uint32_t Timeout
)
```

مقادیر مورد نیاز را در حافظه نوشته و با استفاده از تابع

```
bool EE24_Read(
    EE24_HandleTypeDef *Handle, uint32_t Address, uint8_t *Data, size_t Len,
uint32_t Timeout
```

```
)
```

مقادیر را از آن می‌خوانیم.

۵-۱-۴-ماژول ساعت

ماژول ساعت با ارتباط I2C کار می‌کند. برای دسترسی راحت به امکانات این ماژول، از کتابخانه ds1307_for_stm32_hal استفاده شده است. برای تنظیم کردن ساعت این ماژول از توابع زیر استفاده می‌شود:

```
void DS1307_SetDayOfWeek(uint8_t dow);
void DS1307_SetDate(uint8_t date);
void DS1307_SetMonth(uint8_t month);
void DS1307_SetYear(uint16_t year);

void DS1307_SetHour(uint8_t hour_24mode);
void DS1307_SetMinute(uint8_t minute);
void DS1307_SetSecond(uint8_t second);
void DS1307_SetTimeZone(int8_t hr, uint8_t min);
```

جهت گرفتن اطلاعات زمان واقعی حال حاضر، توابع زیر مورد استفاده قرار می‌گیرند:

```
uint8_t  DS1307_GetDayOfWeek(void);
uint8_t  DS1307_GetDate(void);
uint8_t  DS1307_GetMonth(void);
uint16_t DS1307_GetYear(void);

uint8_t DS1307_GetHour(void);
uint8_t DS1307_GetMinute(void);
uint8_t DS1307_GetSecond(void);
int8_t  DS1307_GetTimeZoneHour(void);
uint8_t DS1307_GetTimeZoneMin(void);
```

نیاز است زمان حال به صورت ساعت پازیکس[1] تبدیل شده تا راحت‌تر بتوان آن را میان نرم‌افزارهای مختلف با زبان‌های مختلف مبادله کرد. در زبان سی، می‌توان از ساختمان `struct tm` و تابع

```
time_t mktime(struct tm* const _Tm)
```

موجود در کتابخانه time، به مقدار ساعت پازیکس رسید. این مقدار بصورت `long` بوده و برای تبدیل آن به متن میبایست از `"%ld"` استفاده کرد.

[1] Posix Time

۴-۱-۶-ماژول سیمکارت

ماژول سیمکارت دارای اتصال سریال می‌باشد و از این طریق با پردازنده ارتباط برقرار می‌کند. این ماژول برای هر کدام از ویژگی‌هایش، کد از پیش تعیین شده دارد که بعد از ارسال این دستورات، عمل موردنظر را انجام می‌دهد.

اعمال مورد استفاده در این پروژه، عبارتند از فعالسازی حالت متنی، ارسال و دریافت پیامک و حذف پیامک‌های موجود در حافظه سیمکارت است.

```
#define ENABLE_TEXT_MODE     "AT+CMGF=1"
#define SEND_SMS             "AT+CMGS="
#define READ_SMS             "AT+CMGR=1"
#define DELETE_ALL_SMS       "AT+CMGDA="
#define DELETE_ALL_SMS_MODE "DEL ALL"
#define SMS_REC_NOTIF        "+CMTI"
```

نیاز است برای ارسال برخی کاراکترها، مقادیر معادل آنها به فرمت ASCII[1] فرستاده شود.

```
#define Enter            10
#define CR               13
#define Double_Quotation 34
#define SUB              26
```

برای راحتی استفاده از این ماژول دو تابع

```
void Gsm_SendSms(char *phonenumber, char *message)
```

و `void Gsm_ReadSms(char *phonenumber, char *message)` نوشته شده است.

با استفاده از تابع `bool Gsm_WaitForMessage()` میتوان بصورت پیوسته بافر[2] سریال را بررسی کرده و در صورت وجود پیامک جدید، آن را اطلاع دهد.

۴-۱-۷-سیستم‌عامل پردازنده

برای همزمان اجرا شدن وظایف محول شده به پردازنده میبایست از سیستم‌عامل RTOS استفاده کرده و وظایف را دسته‌بندی کنیم.

۴-۱-۷-۱ وظیفه مدیریت صفحه نمایش

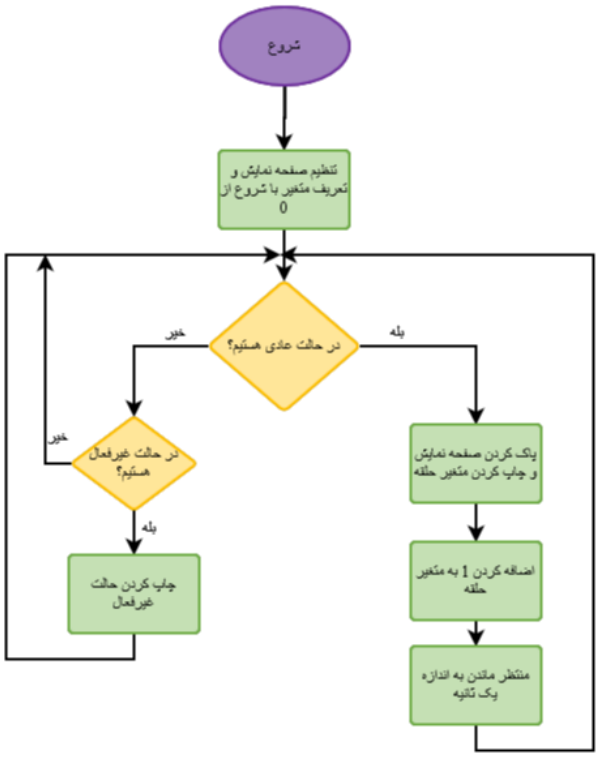
این وظیفه برای نمایش حالت سیستم در صفحه نمایش است. در حین اجرای این وظیفه، یک شمارنده نیز در پس‌زمینه اجرا شده و ثانیه‌های گذرانده از زمان روشن شدن سیستم را در خود ذخیره می‌کند. این شمارنده در حالت عادی سیستم در صفحه نمایش نشان داده می‌شود. در حالت غیرفعال وظیفه دارد صفحه نمایش را کامل پاک کرده و فقط پیغام مرتبط را در صفحه نمایش نشان دهد و همچنان شمارنده را در

[1] American Standard Code for Information Interchange
[2] Buffer

حالیکه همچنان در پس‌زمینه در حال اجراست، از صفحه نمایش حذف کند. این وظیفه نیاز به رم کمی دارد زیرا تخصیص حافظه زیادی در آن انجام نشده است؛ همچنین اولویت اجرای آن پایین است.

```
/* Definitions for counterTask */
osThreadId_t          counterTaskHandle;
const osThreadAttr_t counterTask_attributes = {
    .name       = "counterTask",
    .stack_size = 128 * 1,
    .priority   = (osPriority_t)osPriorityNormal,
};
```

فلوچارت این وظیفه را می‌توان در تصویر زیر دید:



شکل ۱-۰: فلوچارت وظیفه صفحه نمایش

۲-۷-۱-۴- وظیفه بررسی مداوم صفحه کلید

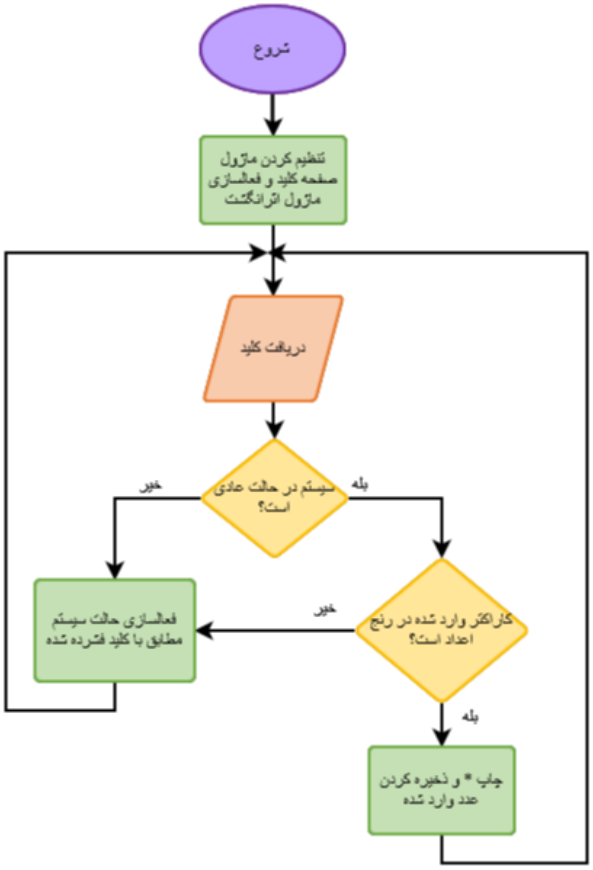
نیاز است صفحه کلید مداوم بررسی شود تا بلافاصله پس از دریافت فرمان جدید، دستورالعمل‌های لازم اجرا شوند. به دلیل اینکه بررسی این موضوع باعث می‌شود بقیه اجزای سیستم متوقف شوند، میبایست در وظیفه جداگانه‌ای انجام شود. این وظیفه نیاز به رم متوسطی داشته و اولویت متوسطی نسبت به بقیه وظایف حیاتی دارد.

```
/* Definitions for keypadTask */
osThreadId_t          keypadTaskHandle;
const osThreadAttr_t keypadTask_attributes = {
    .name       = "keypadTask",
```

```
    .stack_size = 128 * 4,
    .priority   = (osPriority_t)osPriorityLow,
};
```

تنظیمات اولیه این وظیفه شامل فعالسازی صفحه کلید، منتظر ماندن برای وظیفه قبلی که صفحه نمایش را مقداردهی کند و تایید کردن رمز عبور ماژول اثرانگشت که باعث فعال شدن این ماژول می‌شود.

پس از اجرای تنظیمات اولیه، روند برنامه منتظر فشردن کلیدی می‌ماند. سپس با توجه به کلید فشرده شده، حالت سیستم را به حالت عادی، حالت غیرفعال، حالت تنظیمات یا حالت‌های عملیات کاربر تغییر داده و روند برنامه به توابع متناسب حالت جدید سیستم سپرده می‌شود.



شکل ۲-۰: دیاگرام وظیفه صفحه کلید و اثرانگشت

۳-۷-۱-۴- وظیفه پردازش پیامک‌های دریافتی

در این وظیفه، ابتدا تنظیمات اولیه ماژول سیمکارت از جمله فعالسازی حالت متنی و حذف پیامک‌های قبلی انجام شده و سپس روند اجرای برنامه، جهت بررسی پیامک‌ها، بلافاصله پس از دریافت، به حلقه بی‌نهایت رفته و منتظر پیامک جدید می‌ماند.

این وظیفه نیاز به رم متوسطی جهت ذخیره شماره تلفن، متن و ایجاد داده جیسون از متن دارد و اولویت آن نسبت به بقیه سیستم کمی بیشتر است.

```
/* Definitions for gsmTask */
osThreadId_t          gsmTaskHandle;
const osThreadAttr_t gsmTask_attributes = {
    .name       = "gsmTask",
    .stack_size = 128 * 4,
    .priority   = (osPriority_t)osPriorityLow,
};
```

شکل ۳-۰: دیاگرام وظیفه پردازش پیامک

۴-۷-۱-۴- وظیفه تنظیم ساعت و حافظه EEPROM

ابتدا ماژول ساعت راه‌اندازی شده و ساعت تنظیم شده را به سیستم اعلان می‌کند. سپس متغیر مربوط به ماژول حافظه را مقداردهی کرده که بتوان تنظیمات ذخیره شده را بازیابی کرد. اگر عمل خواندن به درستی انجام شود، داده ذخیره شده به فرم جیسونی تبدیل و متغیرهای سیستم را مقداردهی می‌کند. در ادامه، وارد حلقه بی‌نهایت شده و منتظر دریافت سیگنال ذخیره تنظیمات می‌ماند. با دریافت این سیگنال، بلافاصله متغیرهای برنامه را به فرم جیسونی تبدیل و آنها را در حافظه ذخیره می‌کند. این وظیفه نیاز به رم بیشتر و اولویت بیشتری نسبت به سایر وظایف سیستم دارد.

```
/* Definitions for settingsTask */
osThreadId_t          settingsTaskHandle;
const osThreadAttr_t settingsTask_attributes = {
    .name       = "settingsTask",
    .stack_size = 128 * 8,
    .priority   = (osPriority_t)osPriorityLow,
};
```

شکل ۴-۰: دیاگرام وظیفه مدیریت ماژول حافظه و ساعت

۴-۲- کدنویسی برنامه اندرویدی

برنامه اندرویدی با زبان دارت نوشته شده و از کتابخانه Material [1] برای ظاهرسازی قشنگ استفاده شده است. این بخش مربوط به نحوه کدنویسی این برنامه است که به تفکیک اجزای مختلف ارائه می‌شود.

۴-۲-۱-ارسال و دریافت پیامک

کتابخانه telephony وظیفه ارسال و دریافت پیامک را دارد. کافیست تابع

```
Future<void> sendSms(String to, String message)
```

برای ارسال یا تابع

```
reciveSms(dynamic Function(SmsMessage) onNewMessage)
```

برای دریافت پیامک فراخوانی شوند.

[1] Material Design 3, Google

تابع ارسال پیامک، در صفحه تنظیمات و کاربران، بعد از فشردن دکمه‌های «ارسال تنظیمات»، «اضافه کردن کاربر» یا بعد از درخواست بروزرسانی لیست کاربران فراخوانی می‌شود و تنظیمات جدید یا کاربر جدید را به قفل الکترونیکی ارائه می‌دهد.

تابع دریافت پیامک، بعد از باز شدن صفحه اصلی یا درخواست بروزرسانی لیست کاربران فراخوانی می‌شود تا نرم‌افزار اطلاعات ارسالی را در صفحه اصلی یا کاربران نمایش داده نشده را در صفحه کاربران نمایش دهد.

۲-۲-۴-پایگاه داده

در نرم‌افزار اندروید، دو نوع پایگاه داده متفاوت استفاده شده است. پایگاه داده‌ای از نوع SQL با استفاده از سیستم Sqlite و پایگاه داده‌ای از نوع NoSQL با استفاده از Shared Preference وجود دارد.

۱-۲-۲-۴- پایگاه داده Sqlite

برای استفاده از پایگاه داده Sqlite نیاز است مقادیر قابل ذخیره، مدلسازی شوند. یک مدل برای پیغام‌ها که اطلاعاتی همچو شناسه، شماره تلفن، متن پیغام و زمان دریافت شده و یک مدل برای کاربران که اطلاعاتی همچو شناسه، نام کاربر و زمان اضافه شده وجود دارد.

```
class Message {
  int? id;
  String? address;
  String? text;
  int? date;
}

class User {
  int? id;
  String name;
  int? date;
}
```

این مدل‌ها نیاز به پیاده‌سازی دو تابع `Map<String, dynamic> toMap()` و

```
  factory Message.fromMap(Map<String, dynamic> json)
```

برای تبدیل به / از فرمت جیسون دارند.

سپس باید کلاس پایگاه داده را برای هر کدام پیاده‌سازی کرد. برای پیاده‌سازی میبایست برای هر کدام یک جدول در پایگاه داده ساخت. جداول موردنیاز به شرح ذیل می‌باشند:

```
CREATE TABLE IF NOT EXISTS messages (
  id INTEGER PRIMARY KEY,
  address TEXT,
```

```
  text TEXT,
  date INTEGER
)
```

و برای کاربران داریم:

```
CREATE TABLE IF NOT EXISTS users (
  id INTEGER PRIMARY KEY,
  name TEXT,
  date INTEGER
)
```

و در آخر از کتابخانه sqflite استفاده کرده تا فایل پایگاه داده را در محل مطمئنی بسازد. ساخت فایل پایگاه داده، جدول‌ها و راه دسترسی به آنها از جمله اضافه، بروزرسانی، حذف کردن و گرفتن لیست مقادیر موجود در کلاس‌های مربوطه پیاده‌سازی شده‌اند.

```
class MessagesDB {
  MessagesDB._privateConstructor();
  static final MessagesDB instance = MessagesDB._privateConstructor();

  static Database? _database;
  Future<Database> get database async => _database ??= await
_initDatabase();

  Future<Database> _initDatabase() …

  Future _onCreate(Database db, int version) …

  Future<List<Message>> getMessages() …

  Future<int> add(Message message) …

  Future<int> remove(int id) …

  Future<int> update(Message message) …

class UsersDB {
  UsersDB._privateConstructor();
  static final UsersDB instance = UsersDB._privateConstructor();

  static Database? _database;
  Future<Database> get database async => _database ??= await
_initDatabase();

  Future<Database> _initDatabase() …

  Future _onCreate(Database db, int version) …

  Future<List<User>> getUsers() …

  Future<int> add(User user) …
```

```
  Future<int> remove(int id) …

  Future<int> update(User user) …
}
```

۲-۲-۴-۲- پایگاه داده Shared Preference

کتابخانه shared_preferences دسترسی موردنیاز به این پایگاه داده را ایجاد می‌کند. کلاس این پایگاه داده از ChangeNotifier مشتق[1] شده تا پس از تغییر یک متغیر از تنظیمات بتوان برنامه را متناسب با تغییرات بروزرسانی کرد.

در این کلاس میبایست شی SharedPreferences ساخته شده و برای هر متغیر در تنظیمات، متدهای تنظیم و خواندن نوشته شوند.

```
class DataStorage extends ChangeNotifier {
  SharedPreferences? _prefs;

  final String phoneNumberKey = 'phoneNumber';
  late String _phoneNumber;
  String get phoneNumber => _phoneNumber;

  DataStorage() {
    _loadprefs();
  }

  setPhoneNumber(String phoneNumber) …

  _initprefs() …

  _loadprefs() …

  _saveprefs() …
}
```

در کد بالا برای مثال متغیر شماره تلفن نشان داده شده است.

۳-۲-۴-صفحات نرم‌افزار

در نرم‌افزار، قابلیت‌ها به چندین صفحه تقسیم شده که به راحتی بتوان آنها را پیدا یا کنترل کرد.

[1] Extend

۱-۳-۲-۴- صفحه ورود

در این صفحه میبایست کاربر اطلاعات هویتی خود را وارد کند که بتواند به قابلیت‌های برنامه دسترسی پیدا کند.

این صفحه از ویجت‌های[1] ارائه شده در فریمورک فلاتر[2] استفاده کرده و از کاربر می‌خواهد نام کاربری، رمز عبور یا اثرانگشت خود را وارد کند.

ظاهر صفحه در تابع `Widget build(BuildContext context)` پیاده‌سازی شده است.

```
class LoginPage extends StatelessWidget {
  const LoginPage({super.key});

  @override
  Widget build(BuildContext context) …
}
```

۲-۳-۲-۴- صفحه اصلی

در صفحه اصلی، پیغام‌های ارسالی از قفل الکترونیکی نشان داده می‌شوند. هنگام اجرای این صفحه، متد `reciveSms(dynamic Function(SmsMessage) onNewMessage)`

`ListView` تنظیم شده که پس از دریافت پیغامی، بلافاصله آن را به لیست اضافه کند. هر پیغام درون یک نمایش داده می‌شوند. برای هر پیغام، یک `Card` ساخته شده (که در آن نمایش داده شود) و متدهای ارسال، انتخاب و حذف پیاده‌سازی شده است. اطلاعاتی همچون زمان دریافت، زمان ارسال، متن پردازش شده و کد عملیات موردنظر نیز قابل دیدن می‌باشند.

```
class SmsCard extends StatefulWidget {
  const SmsCard({
    super.key,
    required this.message,
    this.onSend,
    this.onSelect,
    this.onDelete,
  });

  final Message message;
  final onSendFunction? onSend;
  final onSelectFunction? onSelect;
  final onDeleteFunction? onDelete;
```

[1] Widget
[2] Flutter Framework

```
  @override
  State<SmsCard> createState() => _SmsCardState();
}

class _SmsCardState extends State<SmsCard> {
  @override
  Widget build(BuildContext context) …
}
```

۳-۳-۲-۴- صفحه کاربران

در این صفحه، لیست کاربران را مشاهده می‌کنید. پس از اضافه شدن کاربری به سیستم، شناسه نمایش داده شده و نام کاربر موردنظر میبایست در این صفحه اضافه شود؛ پس از فشردن دکمه اضافه کردن، شناسه و نام کاربر جدید به پایگاه داده اضافه شده و پیامک حاوی نام کاربر برای پردازنده ارسال می‌شود که در آنجا نیز ذخیره شود. لیست کاربران و نحوه نمایش آنها مانند پیغام‌ها می‌باشد.

```
class UserCard extends StatefulWidget {
  const UserCard({
    super.key,
    required this.user,
    this.onSend,
    this.onSelect,
    this.onDelete,
  });

  final User user;
  final onSendFunction? onSend;
  final onSelectFunction? onSelect;
  final onDeleteFunction? onDelete;

  @override
  State<UserCard> createState() => _UserCardState();
}

class _UserCardState extends State<UserCard> {
  @override
  Widget build(BuildContext context) …

}
```

4-2-3-4- صفحه تنظیمات

در این صفحه، لیست متغیرهای سیستم قفل الکترونیکی را مشاهده می‌کنید. برای تغییر هر کدام میبایست ابتدا آن بخش را فعال کرده و سپس تغییرات متناسب را اعمال کنید. در نهایت با فشردن دکمه «ارسال تنظیمات جدید»، پیامک متناسب با این عمل ساخته شده و برای قفل الکترونیکی فرستاده می‌شود. دقت داشته باشید جهت کاهش حجم متن پیامک ارسالی، صرفا فقط تنظیمات بخش‌های فعال ارسال و از سایر بخش‌ها صرفه‌نظر می‌شود.

در هر بخش یک چک‌باکس برای فعال کردن و متناسب با هر متغیر، ویجت‌های مرتبط وجود دارد.

```
class CheckboxCard extends StatefulWidget {
  const CheckboxCard({
    super.key,
    required this.text,
    required this.value,
    required this.onChanged,
  });

  final String text;
  final bool value;
  final onChangedFunction onChanged;

  @override
  State<CheckboxCard> createState() => _CheckboxCardState();
}

class _CheckboxCardState extends State<CheckboxCard> {
  @override
  Widget build(BuildContext context) …
}
```

۴-۲-۳-۵- صفحه تنظیمات برنامه اندرویدی

این صفحه تنظیمات، به تنظیمات خود برنامه اندرویدی اختصاص داده شده است. تنظیماتی مانند فعالسازی حالت تاریک و ...

همچنان برای معرفی قفل الکترونیکی به برنامه، در این صفحه می‌توان شماره تلفن ماژول سیمکارت را وارد کرده تا نرم‌افزار بتواند با آن ماژول صحبت کند.

در این صفحه، از `SwitchCard` استفاده شده تا تغییرات تنظیمات سریعا اعمال شده و نیازی به فشردن دکمه اضافی نباشد.

```
class SwitchCard extends StatefulWidget {
  const SwitchCard({
    super.key,
```

```
    required this.text,
    required this.value,
    required this.onChanged,
  });

  final String text;
  final bool value;
  final onChangedFunction onChanged;

  @override
  State<SwitchCard> createState() => _SwitchCardState();
}

class _SwitchCardState extends State<SwitchCard> {
  @override
  Widget build(BuildContext context) …
}
```

فصل پنجم

تست عملی

۵-۱- مقدمه

در این فصل تصاویری از تست عملی این پروژه ارائه خواهد شد.

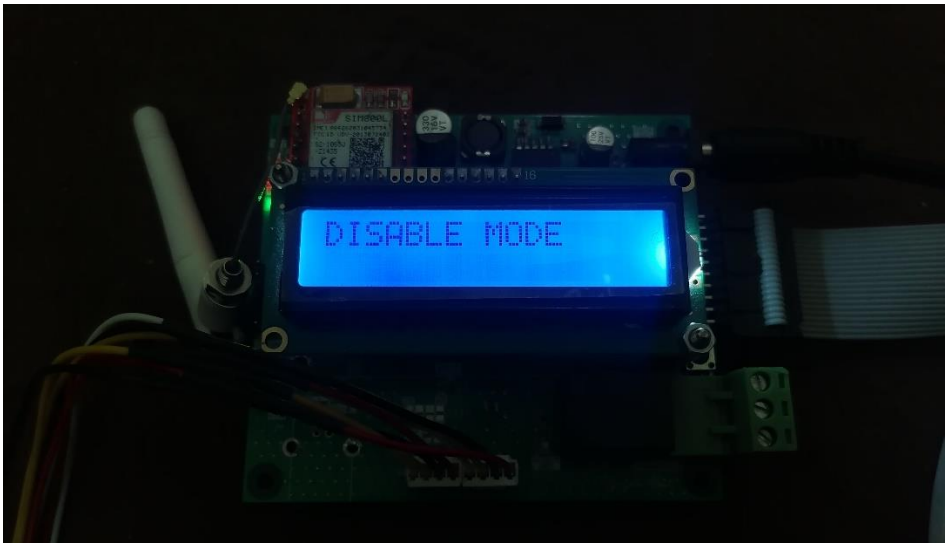
۵-۲- تصاویر پروژه

۵-۲-۱-سخت‌افزار

این تصویر نمایانگر حالت عادی سیستم است. کاربر در حال وارد کردن رمز عبور می‌باشد.

شکل ۱-۰: تصویری از نمایشگر در حالت عادی سیستم حین ورود رمز

تصویر بعدی حالت غیرفعال سیستم را نشان می‌دهد.



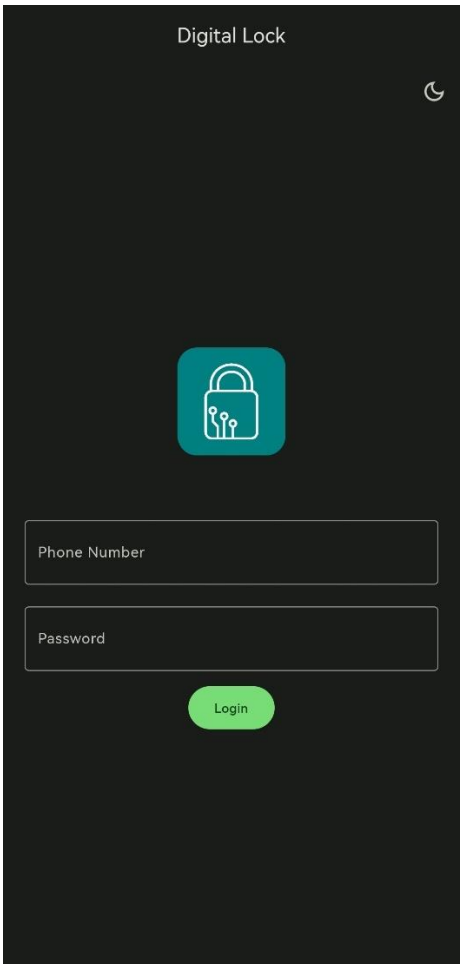
شکل ۲-۰: تصویری از نمایشگر در حالت غیرفعال سیستم

در ادامه، تصویر شروع مراحل ثبت اثرانگشت را خواهید دید.

شکل ۳-۰: تصویری از نمایشگر در ابتدای مراحل ثبت اثرانگشت

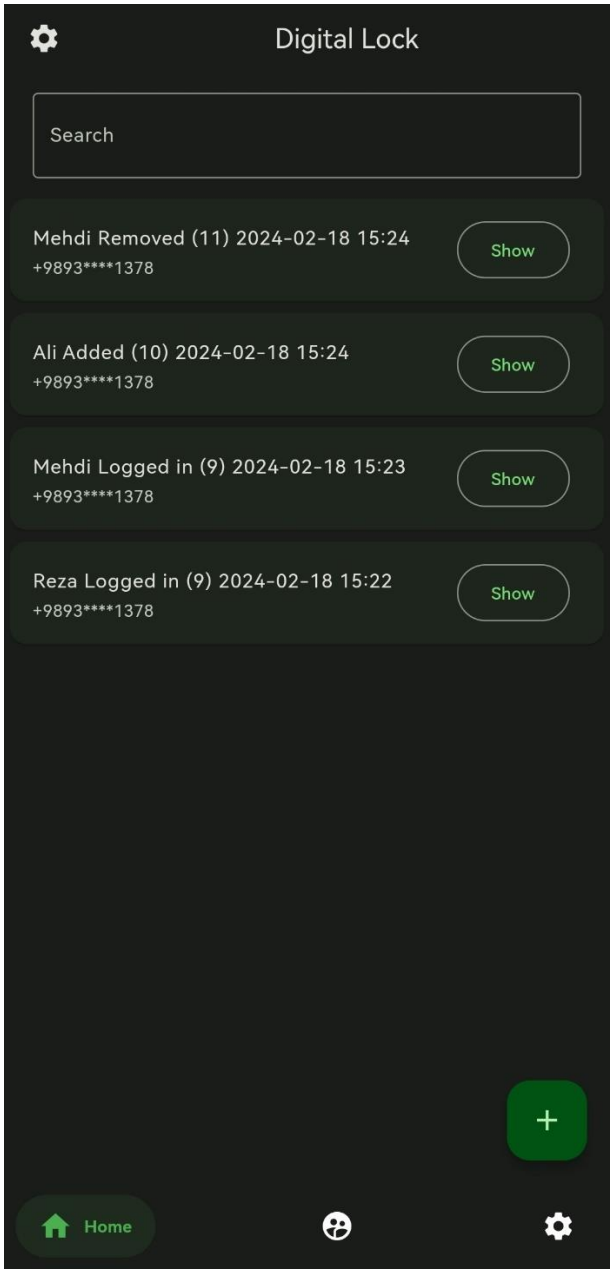
۲-۲-۵-برنامه اندرویدی

تصویر زیر، صفحه ورود به برنامه اندرویدی را نشان می‌دهد.



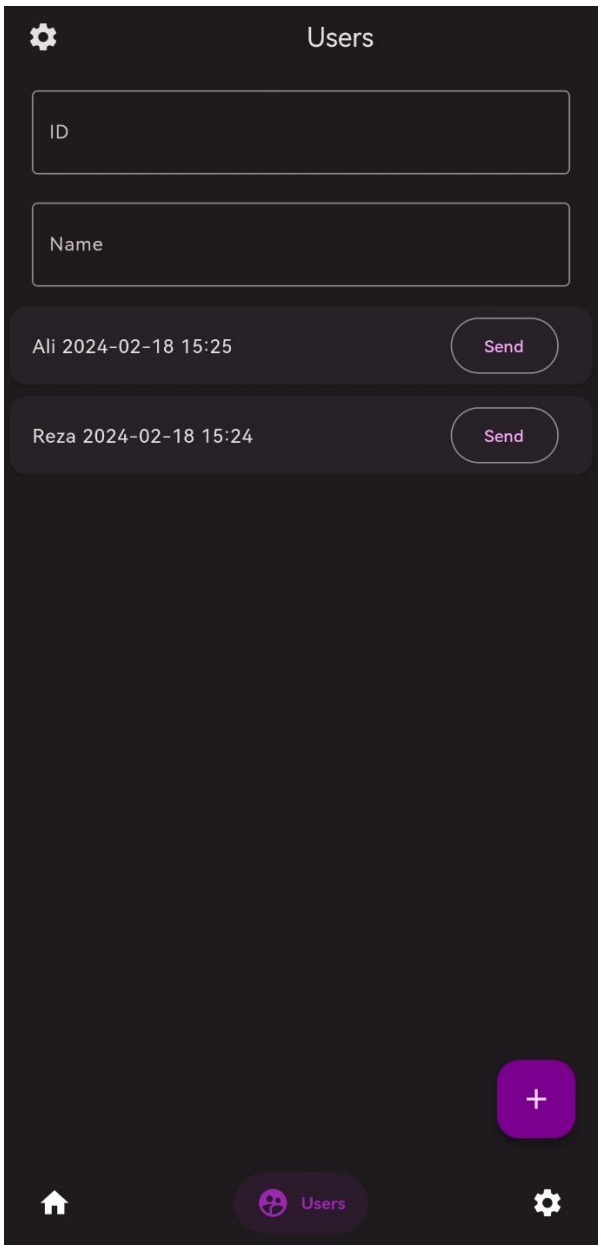
شکل ۴-۰: صفحه ورود به برنامه اندرویدی

تصویر زیر، صفحه اصلی را نشان می‌دهد که محل نمایش پیغام‌های سیستم است.



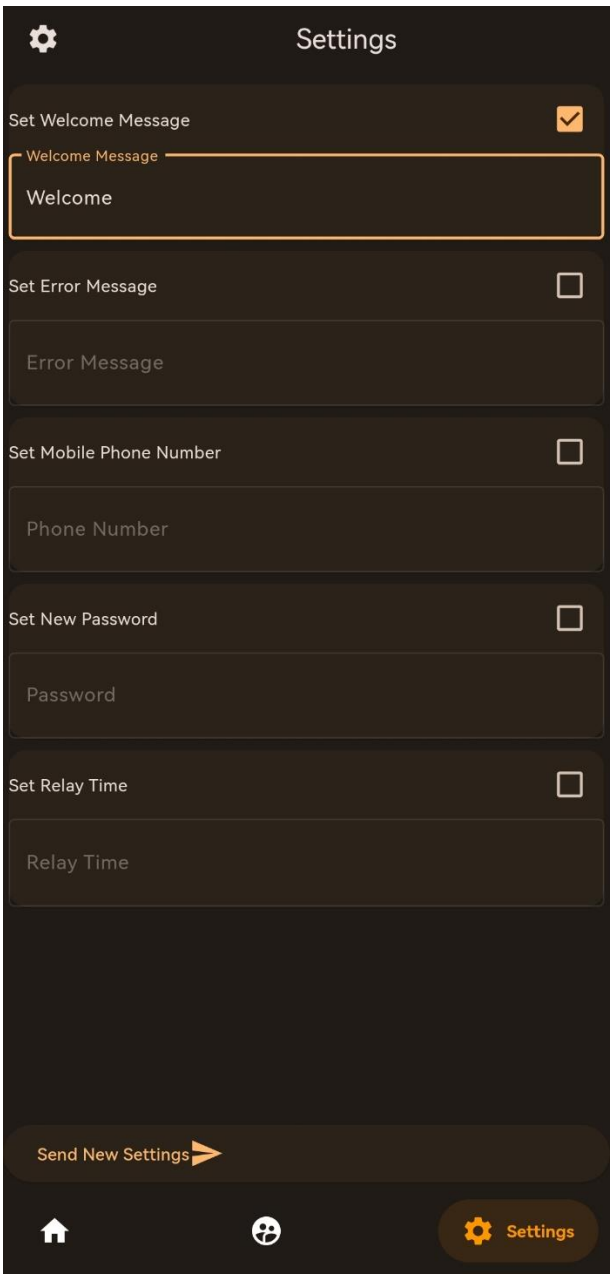
شکل ۵-۰: صفحه اصلی برنامه دارای چند پیغام

صفحه کاربران که لیستی از کاربران ثبت شده در سیستم را در خود نگه می‌دارد، در تصویر زیر مشاهده می‌شود.



شکل ۶-۰: صفحه کاربران دارای چند کاربر تعریف شده

صفحه تنظیمات سیستم، که می‌توان از طریق آن تنظیمات جدید به سیستم اعمال کرد را در تصویر زیر مشاهده می‌کنید.



شکل ۷-۰: صفحه تنظیمات قابل کنترل از طریق برنامه اندرویدی

فهرست مراجع

[1] FreeRTOS Github Release, Github (https://github.com/FreeRTOS/FreeRTOS-Kernel/releases/tag/V10.6.1)

[2] Introducing JSON, JSON (https://www.json.org/json-en.html)

[3] Rivest, Ronald L. (1990). "Cryptography". In J. Van Leeuwen (ed.). Handbook of Theoretical Computer Science. Vol. 1. Elsevier

[4] Daemen, Joan; Rijmen, Vincent (March 9, 2003). "AES Proposal: Rijndael". National Institute of Standards and Technology. p. 1. Archived from the original on 5 March 2013. Retrieved 21 February 2013.

[5] Schneier, Bruce. "Cryptanalysis of MD5 and SHA: Time for a New Standard". Computerworld. Archived from the original on 2016-03-16. Retrieved 2016-04-20. "Much more than encryption algorithms, one-way hash functions are the workhorses of modern cryptography."

[6] Information Technology Laboratory (August 2015). SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions. National Institute of Standards and Technology. Federal Information Processing Standard Publication 202. Retrieved February 29, 2020

[7] Mackenzie, Charles E. (1980). Coded Character Sets, History and Development. Addison-Wesley Publishing Company, Inc. pp. 247–253. ISBN 978-0-201-14460-4. LCCN 77-90165. Archived from the original on May 26, 2016. Retrieved December 29, 2022.

[8] STM32F030C8, STMicroelectronics (https://www.st.com/en/microcontrollers-microprocessors/stm32f030c8.html)

[9] Tarui, Yasuo; Hayashi, Yutaka; Nagai, Kiyoko (1971-09-01). "Proposal of electrically reprogrammable non-volatile semiconductor memory". Proceedings of the 3rd Conference on Solid State Devices, Tokyo. The Japan Society of Applied Physics: 155–162.

[10] Understanding Relays & Wiring Diagrams". Swe-Check. Retrieved 16 December 2020

Abstract

In order to record the entry and exit information of people, it is necessary to install a lock on the door that has the ability to communicate. There should also be a software that can process and display this information in the best way. Also, this lock and software should save the information of trusted people and prevent the entry of unauthorized people. At the same time, it should be possible to update and change the system settings with the available software.

Key Words: Digital Lock, Android Application, Remote Control, SMS, Fingerprint.