بسمه تعالی

گزارش کار پروژه پایانی درس مبانی اینترنت اشیاء

سالن مطالعه هوشمند

اعضای تیم:

مهسا توسلی

زهرا توکلی

دانشگاه صنعتی اصفهان

بهار 1404

چکیده طرح

در این پروژه با بهره گیری از اینترنت اشیاء و بدون نیاز به ناظر انسانی و به صورت خودکار، مواردی مانند سنجش حضور واقعی افراد و عدم اشغال میز مطالعه بدون استفاده از آن، سنجش میزان سکوت سالن مطالعه و هشدار به افراد در صورت رعایت نکردن سکوت، پایش شرایط محیطی و انجام اقدامات لازم انجام می شود و دانشجویان با روش های تعاملی به رعایت نظم و استفاده موثر از فضای مطالعه ترغیب می شوند.

وسایل الکترونیکی مورد استفاده

- سه عدد میکروکنترلر ESP8266
- دو عدد سنسور حرکت HC-SR501
- یک عدد سنسور کنترل کیفیت هوا MQ-135
- یک عدد سنسور دما و رطوبت DHT22
- یک عدد سنسور صدا FC-04
- دو عدد بازر
- تعدادی لامپ LED
- تعدادی مقاومت
- سه عدد برد بورد و سیم جامپر

نحوه بستن مدارها

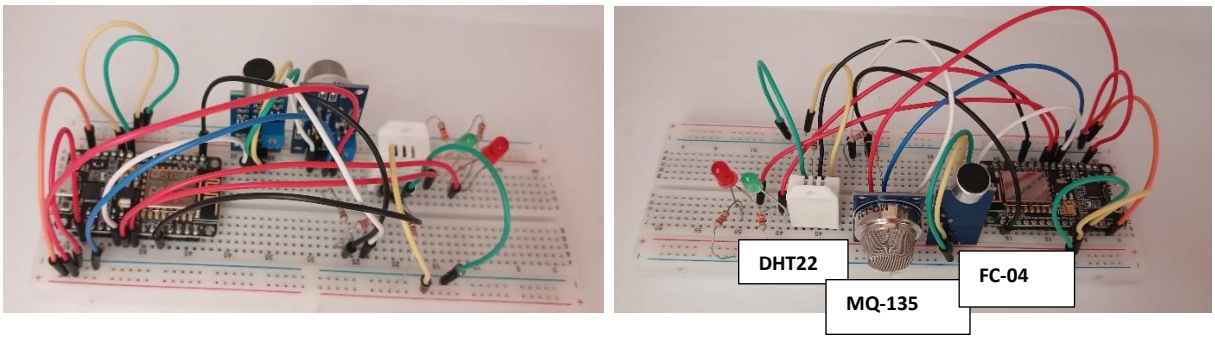
در این پروژه ما برای هر میز مطالعه از یک سنسور حرکت، یک بازر و LED برای اخطارها و یک میکروکنترلر برای ارسال و دریافت داده ها به سرور استفاده کرده ایم.

هم چنین برای هر چند میز که نزدیک به هم قرار دارند یک سنسور صدا برای سنجش میزان صدا در آن ناحیه، یک سنسور دما و یک سنسور کنترل کیفت هوا و یک میکروکنترلر برای ارتباط با سرور داریم.

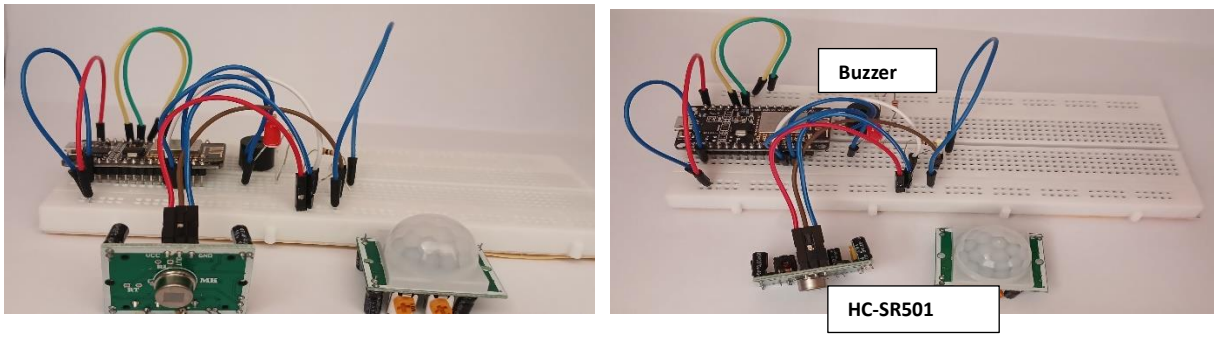
سنسورها با استفاده از سیم جامپر به میکروکنترلرها متصل شده اند ولی در پیاده سازی در محیط واقعی بهتر است در صورتی که به صرفه باشد از سنسورهایی با قابلیت برقراری ارتباط بدون سیم استفاده کنیم.

میکروکنترلرها از طریق ماژول Wi-Fi با سرور ارتباط برقرار کرده و داده رد و بدل می کنند.

برای کالیبره کردن سنسورهای حرکت و صدا از پتانسیومتر تعبیه شده روی سنسور استفاده شده است. به این صورت که برای کالیبره کردن سنسور صدا پتانسیومتر به گونه ای تنظیم می شود که در سکوت LED روی سنسور خاموش و در صورت ایجاد صدا روشن شود. برای کالیبره کردن سنسور حرکت، پتانسیومترها به گونه ای تنظیم می شوند که دامنه تشخیص حرکت (برای مثال تشخیص حرکت به شعاع 3 متر از سنسور)، و مدت زمان high بود پین مربوطه در صورت تشخیص حرکت تنظیم شود. در این پروژه شعاع تشخیص و مدت زمان high بودن پایه سنسور، هر کدام روی کم ترین حد آنها یعنی 3 متر و 2.5 ثانیه تنظیم شده اند. هم چنین برای کالیبره کردن سنسور کنترل کیفیت هوا با توجه به اینکه از پین خروجی آنالوگ سنسور استفاده کرده ایم، از کد مناسب استفاده شده است. (کالیبره کردن با استفاده از پتانسیومتر روی خروجی دیجیتال سنسور اثر دارد و نه خروجی آنالوگ آن.)



تصویر مدار بسته شده برای هر ناحیه از میزها جهت پایش متغیرهای محیطی



تصویر مدار بسته شده برای هر میز مطالعه

توجه: پوشش روی سنسور حرکت در مدار بسته شده برای میزها به برای تشخیص بهتر حرکت برداشته شده است.

در مدارهای بالا برای تغذیه LED ها و بازرها از پین 3.3 ولت میکروکنترلر و برای تغذیه سنسورها از پین V_{in} استفاده شده است تا ولتاژ مناسب جهت راه اندازی سنسورها تأمین شود.

توضیح کدهای میکروکنترلرهای مربوط به میزهای مطالعه

کدهای میکروکنترلرها در این پروژه به دو شیوه‌ی استفاده از پروتکل HTTP و استفاده از پروتکل MQTT برای برقراری ارتباط با سرور نوشته شده اند. متاسفانه به دلیل اینکه در برقراری ارتباط با MQTT Server یا MQTT Broker پلتفرم ThingsBoard با مشکل مواجه شدیم، ادامه کار را با روش HTTP انجام دادیم ولی در اینجا کدهای MQTT نیز توضیح داده می شوند. همچنین کدها در محیط آردوینو نوشته شده اند.

منطق برنامه این میکروکنترلر به این صورت است که وقتی میزی توسط کاربر رزرو می شود، میکروکنترلر وظیفه دارد در صورتی که هیچ گونه حرکتی به مدت 20 دقیقه از کاربر تشخیص داده نشد، بازر را برای هشدار به صدا در آورد و LED میز را روشن کند و این وضعیت را به سرور گزارش کند تا پیام مناسبی جهت تحویل میز و برداشتن وسایل برای کاربر در سایت نمایش دهد. سطح صدای بازر برای رعایت سکوت سالن کم بوده و در مدت زمان کوتاهی به صدا در می آید ولی با این حال بهتر است از روشهای مناسب تری برای اخطار استفاده شود. هم چنین در صورتی که در ناحیه ای سر و صدا وجود داشته باشد، میکروکنترلر دستوری مبنی بر به صدا درآوردن بازر و روشن کردن LED برای اخطار به کاربر از سرور دریافت می کند.

روش HTTP: ابتدا برای جلوگیری از stackoverflow سعی شده است متغیرها به صورت گلوبال تعریف شوند. متغیر deskOccupied مشخص می کند که میز توسط فردی در سایت رزرو شده است یا خیر. متغیر lastMotionTime زمان آخرین حرکت کاربر را ذخیره میکند تا در صورت گذشت 20 دقیقه از آن وضعیت به سرور گزارش شود. متغیر

TIMEOUT همان 20 دقیقه را برحسب میلی ثانیه ذخیره می کند. که البته در مرحله تست برنامه این زمان 1 دقیقه در نظر گرفته شده است. برای تایمر، از تایمر نرم افزاری کتابخانه Ticker.h استفاده شده است که هر یک دقیقه مدت زمانی که از آخرین حرکت گذشته است را بررسی می کند و در صورت عدم حرکت متغیر منطقی ای را برابر true می کند تا در تابع loop اقدامات لازم انجام شود. هم چنین سنسور حرکت در صورت تشخیص حرکت وقفه ای ایجاد میکند و در تابع هندلر وقفه متغیر منطقی ای true می شود تا در تابع loop اقدامات لازم مانند تغییر متغیر lastMotionTime انجام شود. با توجه به سناریو ذکر شده در بالا میکروکنترلر موقع ارسال وضعیت عدم حرکت به سرور در نقش کلاینت و موقع دریافت دستورات سرور مبنی بر به صدا در آوردن بازر و اخطار در نقش سروری عمل می کند که سرور اصلی به آن متصل شده و دستورات را می فرستد. برای ارسال وضعیت عدم حرکت میکروکنترلر یک درخواست HTTP POST به URL مورد نظر یعنی http://192.168.75.224:5000/sensor-data می زند که آدرس IP با IP سرور یا نام دامنه آن جایگزین می شود. هم چنین سرور در صورتی که فردی در سایت میزی را رزرو یا آزاد کند برنامه سرور به آدرس http://microcontroller IP address/reserve و http://microcontroller IP address/release درخواست HTTP POST می زند تا میکروکنترلر مقدار متغیر deskOccupied را تغییر دهد. برای ارسال دستور اخطار به میکرو کنترلر، سرور به آدرس http://microcontroller IP address/sound_alert درخواست می دهد.

در تابع setup میکروکنترلر هم کارهایی مانند راه اندازی ارتباط سریال با کامپیوتر، اتصال به Wi-Fi، راه اندازی سرور، پیکربندی پین ها و وقفه ها انجام شده است.

روش MQTT: در این روش ابتدا در سایت ThingsBoard بخش Cloud نظیر هر میکروکنترلر و سرور یک Device ساخته می شود و از token آن در کد برای برقراری ارتباط با MQTT Server استفاده می شود. در کدهای این بخش هم تمام اجزا مانند تابع setup و loop، تقریبا مانند بخش قبل هستند با این تفاوت که در تابع setup میکروکنترلر سعی می کند به MQTT Server با استفاده از تابع client.connect و token ای که از سایت ThingsBoard گرفته است متصل شود و در تاپیک هایی به صورت v1/devices/me/rpc/request/+ با استفاده از تابع client.subscribe مشترک شود و subscribe کند. در تاپیک ذکر شده و تاپیک هایی که در ادامه ذکر می شوند، بخش me با token دستگاه جایگزین می شود. مشترک شدن میکروکنترلر در این تاپیک برای دریافت دستورات سرور است و در واقع سرور در این تاپیک ها دستورات را publish می کند. هر زمان سرور چیزی در این تاپیک ها منتشر کند تابع callback میکروکنترلر اجرا شده و اقدامات مناسب انجام می شود. هم چنین میکروکنترلر عدم حرکت را با استفاده از تابع client.publish در تاپیک v1/devices/me/telemetry به صورت یک document جیسون به صورت {"\timer\":\"timeout"\} منتشر میکند و سرور در این تاپیک ها قبلا subscribe کرده است و این انتقال داده ها بین میکروکنترلرها و سرور توسط MQTT Server انجام می شود.

توضیح کدهای میکروکنترلرهای مربوط به پایش متغیرهای محیطی در یک ناحیه

منطق برنامه این میکروکنترلر به این صورت است که متغیرهای محیطی مانند دما و رطوبت و یا کیفیت هوا را با استفاده از سنسورها پایش می کند و اقدامات لازم مانند روشن کردن cooler در صورت گرم بودن محیط سالن مطالعه و یا روشن کردن heater در صورت سرد بودن محیط را انجام داده و در صورت نیاز وضعیت را به سرور گزارش می کند. در این پروژه ما از یک LED سبز رنگ به عنوان cooler و یک LED قرمز رنگ به عنوان heater استفاده کرده ایم که به ترتیب در صورتی که دما بیش از 25 درجه و کمتر از 15 درجه سانتی گراد باشد روشن می شوند. یک وظیفه دیگر میکروکنترلر این است که در صورت تشخیص سر و صدا توسط سنسور صدا آن را به سرور گزارش کند تا سرور به کاربران اخطار دهد.

روش HTTP: در ابتدا سعی شده است اکثر متغیرهای برنامه برای جلوگیری از stackoverflow به صورت گلوبال تعریف شوند. برای استفاده از سنسورهای دما و رطوبت و کیفیت کنترل هوا به ترتیب از کتابخانه های DHT.h و MQ135.h و اشیای کلاسهای DHT و MQ135 استفاده شده است. کتابخانه MQ135 شامل کدهای کالیبره کردن سنسور کنترل کیفیت هوا می باشد. سپس در تابع setup کارهایی مانند راه اندازی ارتباط سریال با کامپیوتر، اتصال به Wi-Fi، پیکربندی پین ها و سنسورها و وقفه ها انجام شده است. در تابع loop مقادیر سنسورهای دما و کنترل کیفیت هوا بررسی شده و اقدامات لازم مانند روشن کردن cooler و heater انجام می شود. علاوه بر آن در صورتی که سنسور صدا، صدایی را تشخیص دهد یک پالس با لبه پایین رونده تولید کرده و وقفه ای ایجاد میکند. در تابع هندلر وقفه متغیر soundDetected برابر true شده و در صورت true بودن این متغیر در تابع loop گزارش وضعیت به سرور با ارسال درخواست HTTP POST به http://192.168.75.224:5000/sensor-data انجام می شود.

روش MQTT: این روش هم از نظر انتشار یا اشتراک در تاپیک ها دقیقا مشابه روش MQTT در میکروکنترلرهای میزها است. سایر موارد مانند کدهای HTTP است که در بالا توضیح داده شده است.

توضیح نحوه کالیبره کردن سنسور MQ-135

برای کالیبره کردن سنسور MQ135 ابتدا سنسور را حدود ۲۴ ساعت در هوای پاکیزه و در دمای حدود ۲۰ درجه سانتی گراد، روشن می‌کنیم و مقاومت سنسور MQ135 که همان Rzero می‌باشد را از طریق تابع getRZero() به دست می‌آوریم. سپس در فایل MQ135.h مقدارRZERO را بر اساس داده به دست آمده تغییر می‌دهیم.

توضیح کدهای مربوط به پایگاه داده

در فایل models.py مدل‌های پایگاه داده با استفاده از SQLAlchemy تعریف شده‌اند. این مدل‌ها هر کدام نمایانگر جدولی در پایگاه داده هستند. مدل User اطلاعات کاربران را شامل شماره دانشجویی، رمز عبور هش‌شده، نام کامل، امتیاز و زمان ثبت‌نام ذخیره می‌کند. مدل Zone نشان‌دهنده ناحیه‌های مختلف سالن مطالعه است که می‌توانند به سنسورهای دما، صدا و آلودگی متصل باشند. مدل Desk میزهای مطالعه را به نواحی مرتبط می‌کند و اطلاعاتی مانند وضعیت میز، سنسور متصل، زمان آخرین حضور و توضیحات مربوط به میز را نگهداری می‌نماید. مدل Session نشست‌های حضور کاربران را ثبت می‌کند که شامل زمان شروع و پایان، وضعیت اسکن QR، هشدارهای مرتبط و امتیاز است. مدل Alert هشدارهایی مانند آلودگی صوتی یا عدم حضور را ثبت می‌کند و به میز، جلسه یا ناحیه مرتبط می‌شود. در نهایت، مدل Device اطلاعات مربوط به همه دستگاه‌ها و سنسورها را از جمله نوع، وضعیت، موقعیت مکانی و آخرین پیام ثبت می‌نماید. در فایل setup.py نیز با استفاده از این مدل‌ها، داده‌های اولیه مانند سنسورهای عمومی، ناحیه A، دو میز و سنسورهای حرکتی مرتبط با آن‌ها در پایگاه داده اضافه می‌شود.

توضیح کد های بخش بک اند سرور

توضیح app.py

این بخش با استفاده از فریمورک Flask در پایتون پیاده‌سازی شده است. این سرور بین کاربران، میکروکنترلرها و پایگاه داده نقش واسط را بازی می‌کند.

۱. اجزای اصلی برنامه:

- Flask: فریم‌ورکی برای ساخت APIهای سمت سرور.
- SQLAlchemy: برای تعامل با پایگاه داده (در اینجا sqlite).
- CORS: فعال کردن ارتباط بین سرور و کلاینت در شبکه محلی یا مرورگر.
- requests: برای ارسال درخواست‌های HTTP به میکروکنترلرها.
- Qr_code_reader: برای خواندن کد QR از فایل یا عکس.
- models: شامل مدل‌های دیتابیس مثل

۲.API Routes:

@app.route('/')
نمایش صفحه‌ی اصلی

@ app.route('/register', methods=['POST'])

ثبت‌نام کاربران جدید با دریافت شماره دانشجویی و رمز عبور.

@app.route('/login', methods=['POST'])
ورود کاربران با بررسی رمز عبور (با رمز هش‌شده در دیتابیس).

@app.route('/checkin_qr', methods=['POST'])
پردازش ورود از طریق کد QR:

- کاربر یک QR آپلود می‌کند یا آن را اسکن می کند.
- با خواندن QR، شناسه‌ی میز (desk_id) و ناحیه (zone) به دست می‌آید.
- اگر میز پیدا شود، یک Session ساخته می‌شود، وضعیت میز به "occupied" تغییر می‌کند و درخواست HTTP به آدرس IP میکروکنترلر آن میز فرستاده می‌شود (/reserve).

@app.route('/checkout', methods=['POST'])
خروج کاربر از میز:

- به پایان رساندن نشست و آزاد کردن میز.
- امتیاز دادن به کاربر.
- ارسال درخواست release به میکروکنترلر میز.

@app.route('/sensor-data', methods=['POST'])
دریافت اطلاعات از میکروکنترلرها :

1. اگر type برابر با "motion" باشد یعنی کاربر بدون خروج رسمی میز را ترک کرده:
    - نشست بسته می‌شود.

- کاربر جریمه می‌شود.
- میز آزاد می‌شود.
- درخواست HTTP /release به میکرو کنترلر میز ارسال می‌شود.
- یک هشدار در دیتابیس ثبت می‌شود.

2. اگر type برابر "sound" باشد:

- برای همه میزهای فعال بررسی می‌شود.
- کاربران جریمه می‌شوند و به میزها دستور sound_alert داده می‌شود.

3. اگر temperature و ppm فرستاده شده باشد:

- وضعیت محیطی در بخش comment میز ذخیره می‌شود.

@app.route('/desk-data')
دریافت اطلاعات میز رزروشده برای کاربر فعلی.

روش MQTT:

ابتدا با استفاده از کتابخانه paho.mqtt.client یک کلاینت MQTT ساخته می‌شود که با بروکر مشخص شده (mqtt.thingsboard.cloud) و پورت استاندارد ۱۸۸۳ متصل می‌گردد.

ACCESS_TOKEN برای احراز هویت روی این اتصال تنظیم می‌شود. وقتی اتصال با موفقیت برقرار شود (تابع on_connect)، سرور روی تاپیک کلی v1/devices/+/telemetry سابسکرایب می‌کند که شامل داده‌های سنسورهای تمامی دستگاه‌ها است.

وقتی پیام جدیدی از طریق MQTT دریافت می‌شود (تابع on_message)، payload آن به JSON تبدیل می‌شود و با توجه به نوع داده (که در type است) کارهای مختلفی انجام می‌گردد. برای مثال، اگر نوع پیام "motion" باشد یعنی کاربر بدون ثبت خروج رسمی میز را ترک کرده؛ در این صورت نشست کاربر بسته شده، امتیاز کاربر کاهش می‌یابد، میز آزاد شده و هشدار مربوطه در پایگاه داده ثبت می‌شود. اگر نوع پیام "sound" باشد، برای میزهای فعال جریمه امتیاز اعمال شده و هشدار "صدای بلند" ثبت می‌شود. همچنین اگر داده‌های محیطی مثل دما یا ppm ارسال شده باشد، در بخشcomment میز ذخیره می‌گردد.

در مسیرهای API مانند checkin_qr و checkout، سرور برای تغییر وضعیت میزهای رزرو شده، با ارسال پیام‌هایی به کانال‌های MQTT مربوط به هر میز، به دستگاه‌ها دستور reserve یا آزادسازیrelease

را می‌دهد.

توضیح Qr_code.py

تابع generate_qr(data, filename) برای تولید کد QR از یک string استفاده می‌شود. این تابع یک شیء QRCode می‌سازد، داده متنی را به آن اضافه می‌کند و تصویری از QR ایجاد کرده و در یک فایل با نام دلخواه ذخیره می‌کند.

مثلاً در مثال‌ها، برای میزهای شماره ۱ و ۲ در ناحیه A، دو QR با نام‌های A1.png و A2.png ساخته می‌شوند.

توضیح Qr_code_reader.py

تابع read_qr(image_path) به‌عنوان روش اصلی برای خواندن QR عمل می‌کند. ابتدا تصویر را با استفاده از OpenCV بارگذاری می‌کند. اگر تصویر پیدا نشود یا QR در آن شناسایی نگردد، پیام خطا چاپ شده و سپس تابع کمکی read_qr_pillow برای بررسی مجدد تصویر استفاده می‌شود.

تابع read_qr_opencv(image_path) روش ساده‌تری است که فقط از قابلیت تشخیص QR در OpenCV استفاده می‌کند. این تابع با کمک cv2.QRCodeDetector تلاش می‌کند کد QR را شناسایی کرده و محتوای آن را استخراج کند. اگر موفق شود، داده را برمی‌گرداند و در غیر این صورت، پیام شکست در تشخیص چاپ می‌شود.

تابع سوم یعنی read_qr_pillow(image_path) با استفاده از کتابخانه Pillow تصویر را باز کرده و به فرمت RGB تبدیل می‌کند، سپس با کمک pyzbar.decode تلاش می‌کند QR را رمزگشایی کند. این روش به‌عنوان راه‌حل پشتیبان عمل می‌کند و در مواقعی که روش OpenCV شکست می‌خورد، مفید است.

توضیح کد های بخش فرانت اند سرور

بخش فرانت اند سرور با استفاده از CSS ،HTML و JavaScript نوشته شده و دارای رابط کاربری به زبان فارسی است. این سامانه امکاناتی مانند ثبت‌نام، ورود، ثبت حضور از طریق اسکن QR، مشاهده میز رزرو شده و خروج از سیستم را فراهم می‌کند.

در بخش <head>، فونت فارسی Vazirmatn از طریق CDN بارگذاری شده و استایل‌هایی برای بدنه، فرم‌ها، دکمه‌ها، جدول‌ها و المان‌های نمایشی تعریف شده است.

صفحه دارای پس‌زمینه‌ای با تصویر و رنگ آبی روشن است و فرم‌ها در باکسی به نام main-box قرار گرفته‌اند که با box-shadow و border-radius ظاهر زیبایی دارند. تمام فرم‌ها راست‌چین و کاملاً فارسی‌سازی شده‌اند.

دو فرم وجود دارد:

ورود: شامل شماره دانشجویی و رمز عبور است.

ثبت‌نام: شامل شماره دانشجویی، نام کامل، رمز و تأیید رمز است.

- از جاوااسکریپت برای نمایش فرم مناسب استفاده شده (با توابع showLogin() , showRegister()).
- در صورت وارد کردن اطلاعات نادرست یا خالی، پیام خطا نمایش داده می‌شود.

ثبت حضور با QR Code:

کاربر بعد از ورود می‌تواند با اسکن QR Code میز یا بارگذاری عکس QR، حضور خود را ثبت کند:

- از کتابخانه html5-qrcode برای اسکن استفاده شده است.
- QR باید شامل اطلاعات مانند zone:1;desk:2 باشد که توسط تابع parseQRData تجزیه می‌شود و صحت آن چک می‌شود.
- اطلاعات به سرور ارسال می‌شوند تا حضور ثبت شود.

اگر کاربر میز فعالی داشته باشد، سیستم با درخواست /desk-data اطلاعات میز را نمایش می‌دهد (مانند شماره میز و ناحیه).

- اگر میز رزرو شده باشد، فرم آزادسازی میز (checkout) نمایش داده می‌شود.
- کاربر می‌تواند با زدن دکمه "آزاد کردن میز"، حضور خود را خاتمه دهد.

در بارگذاری مجدد صفحه (window.onload)، با درخواست /me بررسی می‌شود که آیا کاربر قبلاً وارد شده یا خیر.

- اگر وارد شده باشد، اطلاعات او (نام، شماره دانشجویی، امتیاز) در بخش بالای صفحه نمایش داده می‌شود.
- اگر میز فعالی دارد، فرم checkout نمایش داده می‌شود؛ در غیر این صورت فرم check-in نشان داده می‌شود.

هشدارها و پیغام‌ها:

- سیستم دارای امکان نمایش هشدارها است که از مسیر /alerts گرفته شده و با استفاده از SweetAlert2 نمایش داده می‌شوند.
- این هشدارها به صورت لیست تاریخ‌دار و فارسی در پنجره‌ای نمایش داده می‌شوند.