

بِسْمِ اللّٰهِ الرَّحْمٰنِ الرَّحِيْمِ



دانشگاه صنعتی اصفهان

دانشکده مهندسی برق و کامپیوتر

طراحی و پیاده‌سازی سامانه اینترنت اشیا با استفاده از ساختار انتشار/اشتراک

گزارش پژوهه کارشناسی مهندسی کامپیوتر، گرایش نرم‌افزار

امیر خزاعی

استاد راهنمای

دکتر مجید نبی

فهرست مطالب

<u>صفحه</u>	<u>عنوان</u>
چهار	فهرست مطالب
پنج	فهرست تصاویر
۱	چکیده
۲	فصل اول: مقدمه
۲	۱-۱ تاریخچه اینترنت اشیا
۵	۱-۲ کاربردها و ضرورت
۶	۱-۳ مشکلات
۷	۱-۴ اهداف و دستاوردها
۷	۱-۵ ساختار گزارش
۹	فصل دوم: اینترنت اشیا
۱۱	۲-۱ دستگاهها
۱۲	۲-۱-۱ پروتکل‌های ارتباطی
۱۵	۲-۲ گذرگاهها
۱۶	۲-۲-۱ پروتکل‌های ارتباطی
۲۰	۲-۳ سرورها
۲۱	۲-۴ داشبوردها
۲۲	فصل سوم: سامانه اینترنت اشیا به کمک مفهوم انتشار/اشتراک
۲۲	۳-۱ بررسی و طراحی
۲۶	۳-۲ پیاده‌سازی
۴۱	مراجع

فهرست تصاویر

۱-۱	مدل OSI برای شبکه‌های کامپیوتری. [۵]	۳
۲-۱	FPGA با معماری نانو، ساخت شرکت IGLOO [۶]	۴
۳-۱	کامپیوتر کامل و جیبی Raspberry Pi 3 [۷]	۴
۱-۲	نمونه‌ای از اشیا به هم متصل. [۸]	۱۰
۲-۲	نمونه‌ای از یک شبکه توری که شامل یازده دستگاه است. هر دستگاهی به چندین دستگاه دیگر متصل می‌شود. [۹]	۱۳
۳-۲	مقایسه‌ای بین لایه‌های پروتکل‌های TCP/IP و 6LowPAN	۱۴
۴-۲	چند مثال از نحوه اتصال دستگاه‌های مختلف به شبکه و اینترنت برای جابه‌جایی اطلاعات.	۱۵
۱-۳	ساختار درختی سامانه طراحی شده در این مقاله.	۲۳
۲-۳	محصولات مختلف شرکت Arduino که به شکل آرم این شرکت چیده شده‌اند.	۲۷
۳-۳	بردهای شرکت Arduino با طراحی متن‌باز.	۲۸
۴-۳	اتصال تراشه ESP8266 به بردهای Arduino	۲۸
۵-۳	اتصال حسگرها به Arduino	۲۹
۶-۳	نحوه اتصال نمایشگر و Arduino	۳۰
۷-۳	تنظیمات اولیه برای اتصال ماژول ESP8266 به بُرد آردوینو	۳۱
۸-۳	ایجاد نقطه دسترسی و ارسال اطلاعات توسط ارتباط سریال به کامپیوتر	۳۲
۹-۳	اطلاعات ارسال شده از طریق ارتباط سریال به کامپیوتر	۳۳
۱۰-۳	دریافت درخواست و ارسال پاسخ از طریق ماژول ESP8266	۳۵
۱۱-۳	تنظیمات اولیه برای اتصال به واسطه پروتکل MQTT به کمک کتابخانه PubSubClient	۳۶
۱۲-۳	اتصال به واسطه، ارسال پیام در موضوع FromArduino و گرفتن اشتراک در موضوع ToArduino	۳۷
۱۳-۳	نمایش پیام Hello World! در نمایشگر گر با استفاده از کتابخانه LiquidCrystal	۳۷
۱۴-۳	خواندن اطلاعات کارت‌های RFID به کمک کتابخانه MFRC522	۳۸
۱۵-۳	فلوجارت اضافه شدن ویا حذف یک نود از سامانه	۳۸
۱۶-۳	ارسال بسته ای از دستگاه‌ها به سرور	۳۹
۱۷-۳	ارسال بسته ای از سرور به دستگاه‌ها	۳۹
۱۸-۳	نمونه پیاده‌سازی شده توسط ساختار معرفی شده.	۴۰

چکیده

اینترنت یکی از بزرگترین اختراقات بشیریت است. از طریق اینترنت می‌توان اطلاعات را به اشتراک گذاشت. سه دوره مختلف از اینترنت وجود دارد که دوره سوم اینترنت اشیا خواهد بود. اینترنت اشیا شبکه بسیار بزرگ و عظیمی از تمام اشیا مانند، قهوه‌جوش، چراغ قرمز چهارراه، حسگر رطوبت در خاک و ... است که از طریق اینترنت داده‌های خود را به اشتراک می‌گذارند.

در این مقاله به بررسی اینترنت اشیا خواهیم پرداخت. اول مفاهیم و اجزا این تکنولوژی و دوره جدید را بررسی می‌کنیم. پروتکل MQTT که از ساختار انتشار/اشتراک استفاده می‌کند را خواهیم دید و سپس از آن در طراحی سامانه اینترنت اشیا مطرح شده در این مقاله استفاده خواهیم کرد. در انتها نیز به طراحی و ساختار کلی نمونه‌ای از پیاده‌سازی این سامانه خواهیم پرداخت.

واژه‌های کلیدی:

اینترنت اشیا، اینترنت، وب اشیا، CoAP MQTT

فصل اول

مقدمه

۱-۱ تاریخچه اینترنت اشیا

در قرن بیست و یک زندگی می‌کنیم. ساهاست که در تلاش افزایش کیفیت زندگی انسان هستیم. به دنبال راهی برای بهره‌وری بیشتر از محیط اطراف خود می‌گردیم، مانند مصرف بهینه منابع طبیعی شامل آب، باد، نور خورشید و حتی زمان.

اختراع و پیداش کامپیوتر^۲ و به دنبال آن اینترنت^۳ از مهم‌ترین رویدادهای رخداده در هزاره گذشته هستند. با پیداش اینترنت در دهه ۱۹۷۰ [۱] میلادی، ارتباطات بین دستگاه‌ها شروع به شکل گرفتن کرد و سرعت پیشرفت جامعه به شدت بالا رفت. در سال‌های اولیه پیدایش اینترنت تنها می‌توانستیم از طریق ارسال درخواستی به وب سرور^۴ اطلاعاتی را دریافت نماییم. در این بازه زمانی به شبکه جهانی اینترنت وب نسخه ۵^۵ گفته می‌شد. بعدها شاهد افزایش تعامل در اینترنت بودیم. می‌توانستیم به راحتی پول جابه‌جا کنیم یا حساب‌های کاربری برای شبکه‌های اجتماعی داشته باشیم که در این بازه زمانی به شبکه جهانی اینترنت وب نسخه ۶^۶ گفته می‌شد.

²Computer

³Internet

⁴Web Server

⁵Web 1.0

⁶Web 2.0

OSI Model			
	Data unit	Layer	Function
Host layers	Data	7. Application	Network process to application
		6. Presentation	Data representation, encryption and decryption, convert machine dependent data to machine independent data
		5. Session	Interhost communication, managing sessions between applications
	Segments	4. Transport	Reliable delivery of segments between points on a network.
Media layers	Packet/Datagram	3. Network	Addressing, routing and (not necessarily reliable) delivery of datagrams between points on a network.
	Bit/Frame	2. Data link	A reliable direct point-to-point data connection.
	Bit	1. Physical	A (not necessarily reliable) direct point-to-point data connection.

شکل ۱-۱: مدل OSI برای شبکه‌های کامپیوتری. [۵]

امروزه با مطرح شدن مفهوم اینترنت اشیا، نام وب معنایی^۱ یا وب نسخه ۲۳ به شبکه جهانی اینترنت داده شده است. دلیل اصلی ایجاد تفاوت وب نسخه ۲ و وب معنایی در داده‌های معنادار است. در گذشته اینترنت شامل داده‌هایی بوده که تنها برای فرستنده و گیرنده معنا داشته‌اند و دیگران تنها اعداد صفر و یک را میدیدند. با وارد شدن به عرصه اینترنت اشیا بخش بزرگی از داده‌های تولید شده معنادار خواهد شد، یعنی بخشی از داده‌هارا فراداده^۲ تشکیل می‌دهند. فراداده‌ها اطلاعاتی درباره داده‌ها هستند که باعث افزایش چشمگیر کارآیی داده‌ها و سهولت استفاده از آنها خواهند شد.

مدل‌های لایه‌ای مختلفی برای شبکه ارتباطی بین دستگاه‌ها بوجود آمد که یکی از آنها مدل لایه‌ای OSI [۲] بود. ساختار کلی این مدل را می‌توانید در شکل ۱-۱ مشاهده کنید. سطح اول ارتباط فیزیکی بین دستگاه‌ها است که داده‌ها به شکل موج‌های رادیویی فرستاده می‌شوند. داده‌ها می‌توانند از طریق سیم و یا بدون سیم منتقل شوند. در سطح دوم ارتباطات نقطه به نقطه شکل می‌گیرند و می‌توان تنها با داشتن دو لایه اول، دو دستگاه را به هم‌دیگر متصل کرد و اطلاعات را بین آنها جابه‌جا نمود. با داشتن لایه سوم می‌توان بین چندین دستگاه، از طریق مسیریاب^۴ داده‌های مورد نظرمان را جابه‌جا نماییم. لایه چهارم نیز امکان برقرار کردن ارتباط قابل اطمینان^۵ و یک ارتباط دوطرفه را فراهم می‌کند. در این مقاله تمامی لایه‌های بالاتر را برای سادگی به اختصار لایه اپلیکیشن^۶ می‌نامیم.

در دهه اخیر تکنولوژی ساخت قطعات سخت‌افزاری نیز پیشرفت چشمگیری داشته است. در حال حاضر

¹Semantic Web

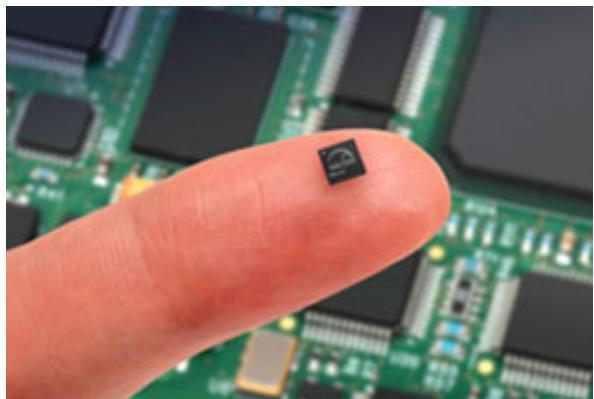
²Web 3.0

³Metadata

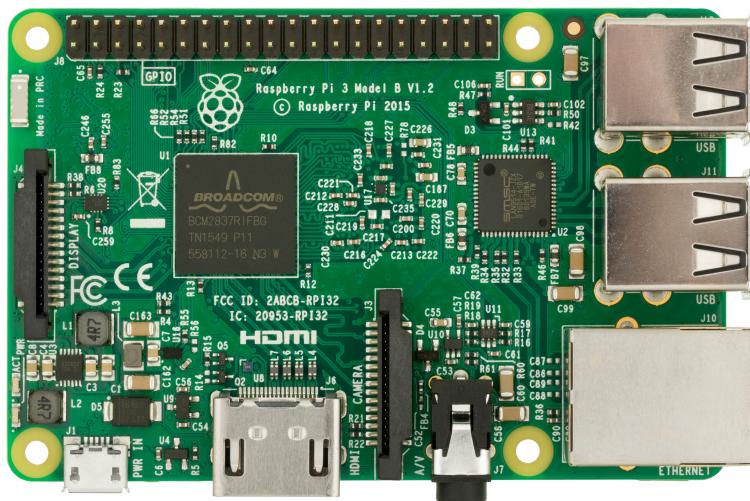
⁴Router

⁵Reliable

⁶Application



شکل ۱-۲: FPGA با معماری نانو، ساخت شرکت IGLOO [۶]



شكل ١-٣: كامبيوتر كامل و جيبي Raspberry Pi 3

شاهد جا گرفتن چندین میلیارد ترازیستور در مساحتی به اندازه یک سانتی متر مریع هستیم. نه تنها اندازه، بلکه مصرف انرژی قطعات سخت افزاری نیز بسیار کاهش پیدا کرده است. ساخت سنسورهای پیچیده‌ای مانند تشخیص نشت گاز و میزان آلودگی، هوا با تکنولوژی امروزه ممکن شده است.

شکل ۱-۲ نشاندهنده یک FPGA است که اندازه بسیار کوچکی دارد. FPGA را می‌توان سخت‌افزاری با قابلیت بروزرسانی طراحی نامید. در حقیقت اتصالات مدارهای درونی این سخت‌افزار قابل تغییر است.

شکل ۱-۳ یک کامپیوتر کامل که اندازه یک کارت بانکی است را نشان می دهد. این کامپیوتر توسط بنیاد Raspberry Pi طراحی و ساخته شده است. تمامی نقشه های طراحی، ساخت و همچنین سیستم عامل نرم افزاری این دستگاه به صورت متن باز در اختیار عموم قرار دارد. امروزه از این دستگاه در بسیاری از پروژه های اینترنت اشیا استفاده می شود. مصرف انرژی این دستگاه نسبت به یک کامپیوتر های عادی بسیار کمتر است اما قدرت پردازشی و کارآمدی آن نسبت به قیمت و اندازه اش حیرت انگیز است.

تمامی موارد مطرح شده تا این نقطه به شکل‌گیری و فراهم شدن زمینه برای بوجود آمدن اینترنت اشیا کمک کرده‌اند. دنیایی که در آن هر جسمی، از یک گلدان داخل راهرو گرفته تا یک ژنراتور برق یا حتی حیوانات یک دامداری می‌توانند اطلاعات و وضعیت خود را از طریق بستر اینترنت به اشتراک بگذارند. حتی می‌توان از طریق همان اینترنت به شکل دستی یا خودکار دستگاه‌های مختلف را کنترل کرد. حیوانی را تصور کنید که نیاز به تزریق روزانه دارویی به مدت چند هفته دارد تا بهبود پیدا کند. می‌توان با اتصال دستگاهی به او و دادن کنترل دستگاه به سیستمی هوشمند نیاز به نیرو انسانی را حذف و احتمال خطأ و اشتباه را کاهش داد. همانگونه که از اسم این تکنولوژی مشخص است، اینترنت اشیا به معنای متصل کردن اشیا مختلف از طریق اینترنت است. اینکه اشیا بتوانند اطلاعات خود را به اشتراک بگذارند و به اطلاعات دیگران دسترسی پیدا کرده و براساس آنها تصمیم بگیرند.

یکی از مهم‌ترین شیوه‌های مطرح ارتباطی در اینترنت اشیا، ارتباط ماشین با ماشین^۱ است. به عنوان مثال دستگاه قهوه‌جوشی که زمان بیدار شدن شما را از طریق ساعت زنگدارتان متوجه شود. همچنین با درخواست سوابق پزشکی شما می‌تواند بیماری دیابت شما را تشخیص دهد و از شکر در قهوه شما استفاده نکند. مثالی دیگر سیستم گرمایش یا سرمایش خانه است که از طریق برقراری ارتباط با موبایل شما می‌تواند متوجه بازگشت شما از سفر طولانی شده و دمای خانه به مقدار مطبوع تغییر دهد.

۱-۲ کاربردها و ضرورت

ماشین‌هایی که به واسطه هوش جمعی می‌توانند رانندگی کنند و از تصادفات اتومبیل با اتومبیل یا حتی دیگر اجسام جلوگیری نمایند. دانستن میزان ترافیک چند کیلومتر جلوتر ویا متوسط سرعت مجموعه ماشین‌های دیگر می‌تواند به کنترل سرعت و یکنواخت نگهداشتیش آن کمک کند و در نتیجه باعث کاهش مصرف سوخت و آلودگی بشود.

خانه‌های هوشمندی که می‌توانید از کیلومترها دورتر و زمانی که در سفر هستید وضعیتشان را بررسی کنید و از امنیت خانه مطمئن شوید. می‌توانید چراغ‌های حیاط را بدون خارج شدن از ساختمان و از داخل اتاق خواب خاموش کنید. حتی می‌توانید روشنایی خانه را به هوش مصنوعی محول کنید و دیگر نگرانش نباشد. یخچال شما با داشتن چند حسگر می‌تواند مواد غذایی داخلش را تشخیص دهد و به طور مثال با تمام شدن سبزی و یا شیر آن را به لیست خرید شما اضافه کند.

دستگاه‌های پوشیدنی مانند ساعت یا کمربند هوشمند نقش مهمی را در سلامت شما ایفا می‌کنند. می‌توانید با

¹ Machine to Machine (M2M)

استفاده از آنها میزان تحرک خود و کالری مصرف شده را بسنجید. به عنوان مثال، برای مسن‌ترها می‌تواند حکم فرشته‌ی نجات را داشته باشد، در لحظات حساسی که بیمار قلبی دچار مشکلی شده و شخصی در اطرافش برای یاری اون حضور ندارد، این دستگاه‌ها می‌توانند با حسگرهای ضربان قلب وجود مشکل را تشخیص دهند و برای بیمارستان درخواست کمک ارسال کنند. در آینده با پیشرفت فناوری نانو و تولید ربات‌های بسیار کوچکی که در خون حرکت می‌کنند می‌توان از طریق اتصال این ربات‌ها به بیمارستان و جابه‌جا کردن داده‌های بیمار، تنها از طریق اینترنت موجود در خانه و بدون نیاز به مراجعه به بیمارستان، مشکلات و بیماری‌های بسیاری را برطرف و حتی ریشه‌کن کرد.

در کشاورزی می‌توان میزان روطوبت و نیترات خاک را با استفاده از حسگرهای مخصوص بررسی کرد و به کمک سیستم‌های هوشمند آبیاری بهیه‌تری داشت. با استفاده از اینترنت اشیا و کنترل خودکار، می‌توان میزان محصولات تولید شده را افزایش داد و به مقابله با مشکل جهانی کمبود موادغذایی و گرسنگی پرداخت. با این روش نه تنها میزان نیاز به نیروی انسانی و هزینه تولید نیز کاهش پیدا خواهد کرد، بلکه از بیماری و مشکلاتی که برای کاگران در هنگام سماپاشی یا برداشت محصول پیش می‌آید نیز می‌توان جلوگیری کرد. صنایع و کارخانه‌ها، سیستم‌های آموزشی، شیوه زندگی روزمره، شهر هوشمند و مصرف انرژی تنها چند مثال کوچک از کاربردهای بی‌حد و مرز این تکنولوژی هستند. بنظر می‌رسد تنها محدود ایده‌پردازی ما است که این تکنولوژی را محدود می‌کند.

۳-۱ مشکلات

از چالش‌هایی که برای اینترنت اشیا وجود دارند، می‌توان به موارد زیر اشاره کرد.

- به علت گستردگی و کاربرد این تکنولوژی در تمامی زمینه‌ها، مقیاس بزرگ داده‌ها و تعداد بسیار زیاد دستگاه‌ها ایجاد مشکل خواهد کرد. بر اساس [۱۰] تا سال ۲۰۲۰ جمعیت کره زمین به ۷,۶ میلیارد نفر و تعداد اشیا متصل به اینترنت به بیش از ۵۰ میلیارد می‌رسد.
- مشکل دیگری که در حال حاضر تقریباً رفع گشته، کمبود تعداد آدرس اینترنتی موجود است. نسخه چهارم پروتکل IP می‌تواند تنها چهار میلیارد دستگاه را آدرس دهی کند. این مشکل با معرفی نسخه شش پروتکل IP برطرف شده اما به علت همه گیر نبودن نسخه شش فعلاً نمی‌توان از این راه حل استفاده کرد. امید است در چند سال آینده نسخه شش به طور کامل جایگزین نسخه چهار بشود.
- به علت عدم یکپارچگی و ساختار منحصر بفرد برای راه حل‌های اینترنت اشیا و وجود روش‌ها و پروتکل‌های

فراوان، داده‌های تولید شده توسط دستگاه‌ها نیز ساختار متفاوت و بسیاری خواهند داشت. [ج ۱۱] [۱۲] [۱۳]. جمع‌آوری و استفاده از چنین داده‌های پیچیدگی‌های فراوان به همراه دارد.

- امنیت مسئله‌ای بسیار مهم است. با اتصال و کنترل تعداد زیادی از دستگاه‌ها و اشیا، در صورتی که بخشی از سامانه مورد حمله‌ی سایبری قرار بگیرد، تمامی کاربران در خطر قرار خواهند گرفت. حفاظت اطلاعات کاربران بخشی از نیازهایی است که یک سامانه باید برآورده کند. خطرات امنیتی تنها شامل دنیای دیجیتال نمی‌شوند. نحوه استفاده کاربر از سامانه نیز می‌توان امنیت را اورا به خطر بیاندازد. طراحی یک سامانه جامع ایمن یکی از بزرگترین چالش‌های پیش‌رو اینترنت اشیا خواهد بود.
- وقتی شرکتی می‌تواند به داده‌ها و اطلاعات شما دسترسی داشته باشد، با آنها چه خواهد کرد؟ آیا شما امکان پاک کردن اطلاعات ماه گذشته را از روی پایگاه‌های اطلاعاتی خواهید داشت؟ چه اشخاص حقیقی یا حقوقی حق دسترسی به اطلاعات شما را دارند؟ نحوه محافظت از حریم شخصی شما از مواردی است که بر استفاده شما از محصول آن شرکت تاثیر فراوان خواهد داشت. ایجاد سیاست حفظ حریم خصوصی^۱ اشخاص که مناسب و جامعه‌پسند باشد نیز چایش دیگری است.

۴-۱ اهداف و دستاوردها

Kevin Ashton بیش از ۱۵ سال پیش مفهوم اینترنت اشیا^۲ مطرح نمود. در چند سال اخیر توجه‌ها به شکل چشمگیری به سمت این تکنولوژی رفته است. با وجود گذشت زمان زیاد از مطرح شدن این مفهوم، هنوز سامانه‌ای جامع و استاندارد برای این امر طراحی و پیاده‌سازی نشده است.

در این مقاله هدف اصلی پایه‌ریزی برای سامانه‌ای جامع و کاربردی است که بتواند با انعطاف‌پذیری کافی تمامی نیازهای موجود را برطرف نماید. این سامانه باید قابلیت استفاده در کوچکترین دستگاه‌ها که منابعی محدود دارند تا پیشرفته‌ترین آن‌ها را داشته باشد. باید سریع و ساده باشد و مهم‌ترین نیاز اینترنت اشیا، یعنی ارتباط ماشین با ماشین را به خوبی پشتیبانی کند.

۵-۱ ساختار گزارش

در فصل دوم به شکل دقیق‌تری به بررسی اینترنت اشیا می‌پردازیم. ساختاری که باید برای اینترنت اشیا استفاده شود به بخش‌های کوچک‌تری تقسیم می‌کنیم و هر بخش را جداگانه خواهیم دید. در نهایت روش‌ها و

¹Privacy Policy

²The Internet of Things

پروتکل‌های موجود برای ساخت یک سامانه را بررسی می‌کنیم و یکی از آن‌ها را برای طراحی سامانه جامع در این مقاله انتخاب می‌کنیم.

در فصل سوم به طراحی سامانه می‌پردازیم و ساختار نمونه پیاده‌سازی شده را خواهیم دید.

فصل دوم

اینترنت اشیا

مانند هر سامانه دیگری اینترنت اشیا نیز شامل بخش‌های مختلفی می‌شود. اولین بخش مربوط به خود اشیا و دستگاه‌ها است. اگر به شکل ۲-۱ نگاه کنید، تعداد زیادی از دستگاه‌های مختلف را می‌بینید. بعضی از آنها حاوی حسگرهایی^۲ هستند که وضعیت محیط اطراف را به شکل داده‌های دیجیتالی توصیف می‌کنند. بعضی دیگر هم شامل محركهایی هستند که بر روی وضعیت محیط تاثیر می‌گذارند و آن را تغییر می‌دهند.

حال که اشیا و دستگاه‌ها را می‌شناسیم، می‌دانیم که نیاز به برقراری ارتباط و جابه‌جایی اطلاعات دارند. استانداردها و پروتکل‌های ارتباطی موجود نیز بخش دیگری از اینترنت اشیا را تشکیل می‌دهند. پروتکل‌های ارتباطی مانند HTTP^۳، TCP^۴، لایه‌های مختلف شبکه و ... اجزا تشکیل دهنده این بخش هستند.

حال که اشیا می‌توانند با یکدیگر صحبت کنند. تنها صحبت کردن آنها کافی نخواهد بود. باید بتوان داده‌های تولید شده را جمع‌آوری کرد و تحلیل نمود. این امر مستلزم ارتقا دادن یکی از دستگاه‌ها و اختصاص قابلیت‌های پیشرفته مانند میزان حافظه فراوان و سریع، توان پردازشی بالا و ... است که آن را سرور^۵ می‌نامیم. اگر حجم و مقدار داده‌ها افزایش یابد داشتن یک سرور کافی نخواهد بود. حتی داشتن چند سرور در یک نقطه جغرافیایی

²Sensors

³Hyper Text Transfer Protocol

⁴Transmission Control Protocol

⁵Server



شکل ۲-۱: نمونه‌ای از اشیا به هم متصل. [۸]

نیز کافی نخواهد بود. زیرا داده‌های سراسر دنیا باید به یک نقطه ارسال شوند که این امر باعث افزایش ترافیک اینترنتی شدیدی بر خطوط ارتباطی آن نقطه جغرافیایی خواهد شد. به همین دلیل باید چندین سرور در نقاط مهم ارتباطی وجود داشته باشد و بین آنها راه ارتباط اختصاصی سریع و قوی برقرار باشد.

گذرگاه‌ها در ارسال داده‌ها به محل ذخیره‌سازی و همچنین مدیریت آنها کمک می‌کنند. وجود گذرگاه باعث کاهش پیچیدگی و نیاز به سرورهای قدرتمند می‌شود. همچنین امکاناتی نظیر کارکرد درست سامانه به شکل محلی در صورت قطع ارتباط با سرور و امکان ارتباط ماشین با ماشین به شکل محلی و عدم نیاز به ارسال داده‌ها تا سرور و دریافت‌شان را برای کاربر به ارمغان می‌آورد.

از داده‌های جمع‌آوری شده با کمک دیگر تکنولوژی‌ها مانند یادگیری ماشین، می‌توان نیازهای کاربر را برطرف نمود. در این بخش اطلاعات و داده‌های انباسته شده را در سرویسی دیگر استفاده خواهیم کرد. به طور کلی به هر نوع سرویس که از داده‌های جمع‌آوری شده استفاده کند، داشبورد^۱ خواهیم گفت. پس در اینجا اینترنت اشیا را به چهار بخش مجزا شکستیم.

- دستگاه‌ها^۲ که شامل حسگرهای محرک ها هستند.
- گذرگاه‌های عبور داده که باعث مدیریت بهتر داده و افزایش قابلیت اطمینان می‌شوند.
- سرورهایی که اطلاعات را دریافت و ذخیره می‌کنند و با استفاده از رابطه‌های برنامه نویسی^۳ مناسب آن را ارائه می‌دهند.
- داشبوردهایی که با استفاده از رابطه‌های برنامه نویسی موجود به اطلاعات دست پیدا کرده و خدمتی جدید ارائه می‌دهند.

¹ Dashboard

² Devices

³ Application Programming Interface (API)

۱-۲ دستگاه‌ها

برای طراحی و ساخت یک را حل اینترنت اشیا باید از این بخش آغاز کنیم. اولین مشخصه‌ای که باید در نظر بگیرید انرژی مصرفی است. حسگرها و محرک‌های شما ممکن با استفاده از انرژی محدود باتری کار کنند. در این حالت شما باید میزان ارسال و دریافت اطلاعات را کاهش بدهید. همچنین از انجام پردازش‌های غیرضروری جلوگیری کنید. یکی از ویژگی‌های مهم که باید برای مصرف کمتر انرژی در نظر گرفت، به خواب رفتن است. در مواقعي که پردازنده کاری برای انجام ندارد، می‌توان آن را به خواب فرستاد تا مصرف انرژی کاهش یابد. با پیدايش تکنلوجی سیستم روی یک تراشه^۱ تمامی بخش‌های مختلف یک دستگاه مانند پردازش‌گر مرکزی، فرستنده و گیرنده‌های رادیویی، کنترل کننده USB و ... بر روی یک تراشه قرار دارند. حال نه تنها می‌توان واحد پردازش‌گر مرکزی، بلکه می‌توان تمامی سیستم روی یک تراشه را به حالت خواب برد و مصرف انرژی را به شکل چشمگیری کاهش داد.

در حرکت بودن یا عدم حرکت نیز بسیار مهم است. اگر دستگاه در حرکت باشد نیاز به استفاده از تکنلوجی‌های ارتباطی دور برد وجود خواهد داشت. در این حالت با شبکه‌هایی روبرو هستیم که سرعت ارسال داده‌ها می‌تواند بسیار کم باشد. فاصله زیاد احتمال بوجود آمدن تصادم در داده‌های ارسالی را بسیار افزایش می‌دهد. برای پی بردن به مختصات دستگاه‌های در حرکت نیاز به موقعیت‌یاب وجود دارد. هر دستگاهی اطلاعاتی را منتشر می‌کند. تعداد دفعات منتشر کردن یه داده جدید اهمیت فراوانی دارد. همانطور که پیش از این به خواهیدن دستگاه اشاره کردیم، می‌توان در بین بازه‌های زمانی ارسال اطلاعات دستگاه به خواب برود. همچنین تعداد دفعات و حجم داده‌های ارسالی می‌تواند در مشخص کردن تکنلوجی رادیویی که باید استفاده بشود کمک کند. به عنوان مثال در دستگاهی که نیاز به ارسال داده تا نرخ چندین مگابایت بر ثانیه باشد نمی‌توان از تکنلوجی بلوتوث^۲ استفاده کرد. نرخ ارسال بلوتوث حداقل ۲۵۰ کیلوبايت بر ثانیه بیشتر نخواهد بود [۱۴].

ساختار داده‌ها نیز اهمیت فراوانی دارد. اطلاعات ممکن است پیچیده باشند، مانند اطلاعات بدن یک بیمار که شامل فشار خون، تعداد ضربان قلب بر دقیقه، متوسط فعالیت روزانه است. دمای اتاق که توصیف شر برابر با یک عدد در واحد سلسیوس^۳ است، داده‌ای ساده محسوب می‌شود.

طول مسافت امواج ارسالی دستگاه‌هایی که ارتباط بیسیم برقرار می‌کنند محدود است. به اندازه مسافتی که امواجشان را می‌توانند ارسال کنند، گستره گفته می‌شود. گستره^۴ دستگاه‌هایی که در حال استفاده هستیم

¹ System on a Chip

² Bluetooth

³ Celsius

⁴ Range

می‌توانند از چند سانتی‌متر مانند کارت‌های اتوبوس که از تکنولوژی RFID استفاده می‌کنند، تا چند کیلومتر مانند دستگاه‌های ردیابی که به حیوانات متصل می‌کنند، متغیر باشد. هر چقدر میزان گستره دستگاه افزایش یابد، احتمال وجود دستگاه دیگر در آن محدوده مکانی نیز افزایش می‌باید. وجود دو دستگاه در این محدوده می‌تواند بر کیفیت ارتباطی هر دوی آنها تاثیر بگذارد. همانطور که پیش از این گفتیم، در چند ساله آینده با انبوهی از دستگاه‌ها مواجه خواهیم بود. حال اگر هر کدام بخواهدن از تکنولوژی‌هایی با گستره بالا استفاده کنند، در محدوده گستره آن‌ها تعداد زیادی دستگاه دیگر وجود خواهد داشت. افزایش اختلال در ارتباطات رادیویی باعث عدم کارآیی این تکنولوژی خواهد شد. به همین دلیل بهتر است در هنگام انتخاب دستگاه و راه حل ارتباطی کوتاه‌بردترین تکنولوژی که نیازهای پروژه ما را برآورده می‌کند انتخاب کنیم.

۱-۱-۱ پروتکل‌های ارتباطی

تکنولوژی‌های ارتباطی دستگاه‌ها بسیار متنوع هستند [۲۵]. ابتدا استانداردهای مربوط به لایه ارتباط داده^۱ را بررسی می‌کنیم. همانطور که مطرح شده بود، با استفاده از این استانداردها می‌توانید دو دستگاه را به یکدیگر متصل کنید.

یکی از این تکنولوژی‌ها NFC^۲ است که برد ارتباطی کمتر از ۲۰ سانتی‌متر دارد. نرخ ارسال داده‌ها در این تکنولوژی پایین بوده و در حدود ۴۰۰ کیلوییت بر ثانیه است [۲۶]. امروزه از این تکنولوژی برای کارت‌های شناسایی و یا در تلفن‌های همراه برای پرداخت‌های هوشمند استفاده می‌شود. مصرف فوق العاده پایین دستگاه‌های دارای NFC باعث شده است که استفاده از این فرستنده و گیرنده برای دستگاه‌هایی که انرژی خود را از طریق باتری تامین می‌کنند، راه حل مناسبی باشد.

تکنولوژی ارتباطی بلوتوث بردی در حدود چند متر دارد [۱۴]. با معرفی نسخه کم مصرف^۳ این تکنولوژی دستگاه‌های ارتباطی به این پروتکل رو آورده‌اند. نرخ جابه‌جایی داده در حدود یک مگابیت بر ثانیه است. از این تکنولوژی در فضاهای کوچک و بسته، مانند خانه هوشمند می‌توان استفاده کرد. همچنین با استفاده از شبکه‌های توری^۴ می‌توان فضای بزرگتری را با استفاده از این تکنولوژی پوشش داد.

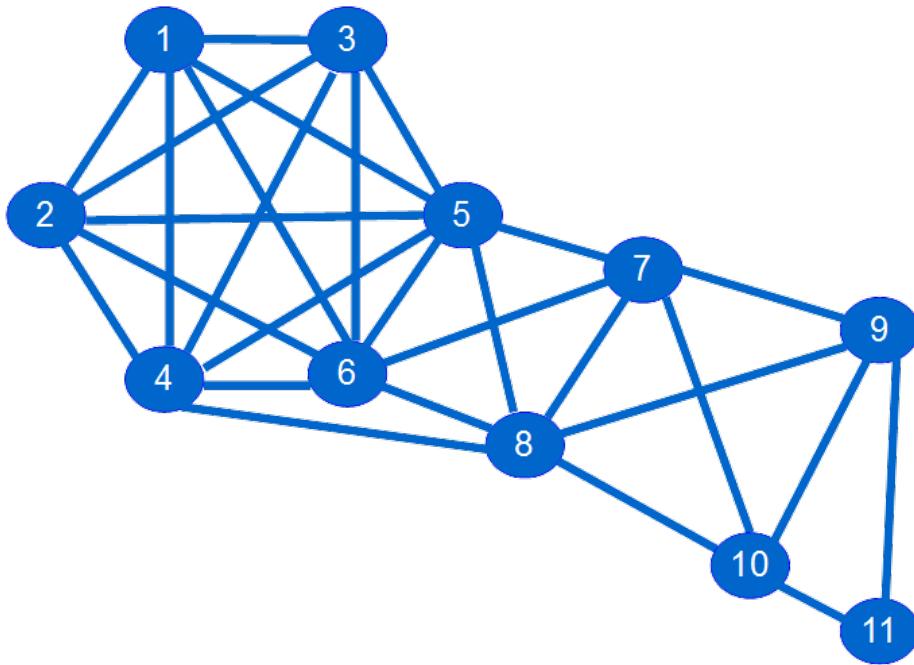
در شکل ۲-۲ نمونه‌ای از شبکه‌های توری را مشاهده می‌کنید. در این نوع شبکه‌ها ارتباط بین دستگاه‌ها ساختار درختی ندارد و هر دستگاه به چند دستگاه دیگر متصل خواهد بود و می‌تواند اطلاعات خود را از طریق آن‌ها ارسال یا دریافت نماید.

¹Data Link Layer

²Near Field Communication

³Bluetooth Low Energy

⁴Mesh Networks



شکل ۲-۲: نمونه‌ای از یک شبکه توری که شامل یازده دستگاه است. هر دستگاهی به چندین دستگاه دیگر متصل می‌شود. [۹]

یکی از پروتکل‌های ارتباطی که به شکل روزمره از آن استفاده می‌کنیم، استانداردهای شبکه‌های بی‌سیم خانگی^۱ هستند[۴]. به کمک این استانداردها برد تا حدود یک کیلومتر افزایش پیدا خواهد کرد. برخلاف دو مورد قبلی مصرف انرژی در این تکنولوژی نسبتاً بالاست و عمر باتری دستگاه را کوتاه خواهد نمود. پروتکل‌های ارتباطی بی‌سیم خانگی سرعت ارسال و دریافت داده‌ها را به بیش از ۵۰۰ مگابیت بر ثانیه افزایش می‌دهند. استفاده از این تکنولوژی برای صنایع و سازمان‌ها مناسب خواهد بود.

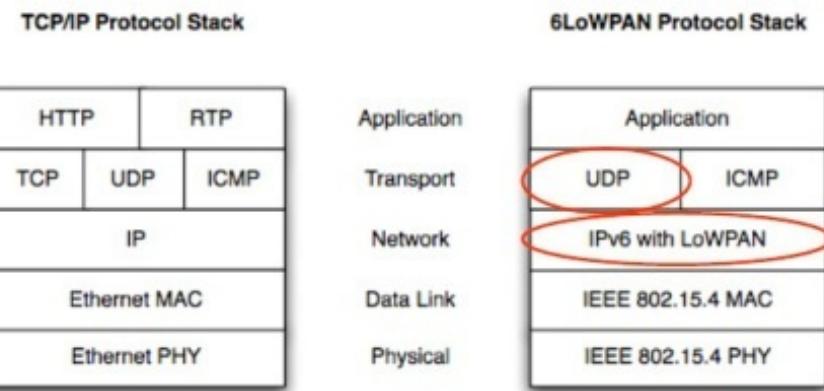
با پیدایش فناوری تکامل بلندمدت^۲ دسترسی به اینترنت همراه بسیار آسان‌تر شد. این فناوری پیش‌زمینه اتصال تعداد زیادی دستگاه را به یک دکل مخابراتی فراهم نموده و می‌توان از آن به عنوان یک راهکار ارتباطی در اینترنت اشیا استفاده کرد. کاهش میزان مصرف انرژی به ازای هر بیت ارسالی در این تکنولوژی دلیل اصلی افزایش استفاده از آن نسبت به نسخه‌های پیشین آن می‌باشد. با استفاده از این فناوری برد موثر تا چندین کیلومتر می‌تواند افزایش پیدا کند. سرعت ۳۰۰ مگابیت بر ثانیه جایه‌جایی اطلاعات نیز از دیگر مزیت‌های این فناوری است. همانند شبکه‌های بی‌سیم خانگی، مصرف انرژی متوسط و حتی زیاد در این فناوری از معایب آن به حساب می‌آید.

به غیر از پروتکل‌های لایه ارتباط داده، به پروتکل‌های لایه شبکه^۳ نیز برای جایه‌جایی داده‌ها بین چند دستگاه نیاز داریم. این استانداردها امکان مسیریابی و آدرس‌دهی را در شبکه محیا می‌کنند.

^۱IEEE 802.11 a/b/g/n/ac/ax/ah

^۲Long Term Evolution (LTE)

^۳Network Layer



شکل ۲-۳: مقایسه‌ای بین لایه‌های پروتکل‌های TCP/IP و 6LoWPAN.

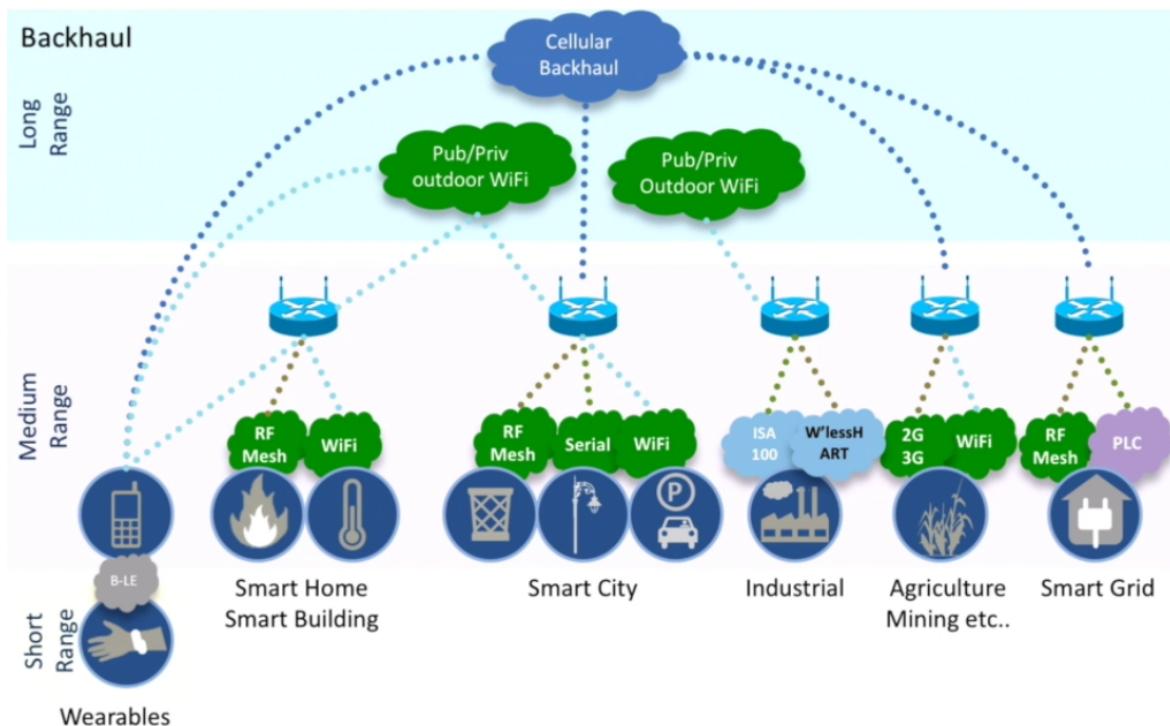
پروتکل اصلی که در حال حاضر کل اینترنت بر آن بنا نهاده شده، پروتکل اینترنت^۱ است. این پروتکل دو نسخه به شماره ۶ و ۴ دارد. نسخه چهار قدیمی و در حال منسوخ شدن می‌باشد. در این نسخه تعداد آدرس‌های ممکن تنها چهار میلیارد است که در مقایسه با نیاز به آدرس‌دهی بیش از ۵۰ میلیارد دستگاه در سال ۲۰۲۰ بسیار کم است. بنابراین نسخه شش این پروتکل معرفی شد که فضای آدرسش ۴ میلیارد برابر بزرگ‌تر است. به گفته طراحان این نسخه، نه تنها می‌توان تمامی اتم‌های کره زمین را با این پروتکل آدرس‌دهی کرد، بلکه می‌توان برای تمامی اتم‌های بیش از ۱۰۰ سیاره دیگر نیز آدرس اینترنتی در نظر گرفت. این استاندارد ساختاری پیچیده دارد و برای پیاده‌سازی و استفاده در دستگاه‌هایی که مصرف انرژی در آن‌ها اهمیت دارد، چندان مناسب نیست. از آنجایی که جامع‌ترین استاندارد موجود پروتکل اینترنت نسخه ۶ است نمی‌توان به راحتی آن را کنار گذاشت. براساس پروتکل اینترنت نسخه ۶، استانداردی به نام 6LoWPAN که برای دستگاه‌های با توان پردازشی و مصرف انرژی پایین ساخته شد^[۱۵]. این استاندارد در حقیقت با در نظر گرفتن استاندارد بلوتوث با مصرف انرژی پایین^۲ که مربوط به لایه فیزیکی است، طراحی شد. در شکل ۲-۳ مقایسه‌ای بین ساختار موجود در اینترنت و ساختار پیشنهادی با استفاده از 6LoWPAN را مشاهده کنید.

در شکل ۲-۴ می‌توانید موارد مختلف استفاده از اینترنت اشیا را مشاهده کنید. اسمی بیشتری از آنچه در این مقاله به بررسی آن‌ها پرداختیم در تصویر قابل مشاهده هستند.

¹Internet Protocol (IP)

²IEEE 802.15.4 BLE

The IoT Wireless Technologies Landscape



شکل ۲-۴: چند مثال از نحوه اتصال دستگاه‌های مختلف به شبکه و اینترنت برای جابه‌جایی اطلاعات.

۲-۲ گذرگاهها

در دنیا میلیاردها دستگاه خواهد بود. در هر ثانیه نزدیک به چندین گیگابایت اطلاعات تولید می‌شود و برای گذراندن چنین حجمی از اطلاعات از یک سیستم مرکزی کار مناسبی نیست. گذرگاه‌ها^۱ نقش مهمی را ایفا می‌کنند. تصور کنید می‌خواهید اطلاعات چند حسگر که داخل یه سالن ورزشی نصب شده اند را برای ذخیره‌سازی به سرور ارسال کنید. به طور مثال اگر ۱۰۰ حسگر حرارتی و کربن‌منواکسید وجود داشته باشد، نیازی به ارسال داده‌های تولید شده توسط تمامی آن‌ها نخواهد بود. می‌توانید میانگین این ۱۰۰ مقدار را حساب کرده و در نهایت دما و میزان کربن‌منواکسید سالن را به سرور ارسال کنید. سیستم‌های هشدار حریق براساس سنسورهای کربن‌منواکسید تصمیم‌گیری می‌کنند. اگر داده‌های سنسور مستقیماً به سرور اصلی رفته و سیستم هشدار نیز داده‌ها را از آنجا دریافت نماید، در صورت اختلال در اتصال اینترنت و ایجاد آتش‌سوزی، سیستم هشدار کارآمد نخواهد بود.

چند وظیفه کوچک یک گذرگاه در مثال قبل مشخص شده است. یکی از آنها پردازش اولیه داده‌ها است. در گذرگاه داده‌های ۱۰۰ سنسور میانگین گرفته خواهد شد. به این شکل نه تنها توان پردازشی سرور و فضای

^۱Gateway

ذخیزه‌سازی کاهش پیدا می‌کند، بلکه خطاب تولید شده توسط چند سنسور تاثیری در نتیجه نهایی نخواهد داشت. یکی دیگر از وظایف گذرگاه، بافر^۱ کردن اطلاعات است. اگر اطلاعات ۲۴ ساعت اخیر در گذرگاه ذخیره شود، با اختلال در اتصال گذرگاه به اینترنت نه تنها کارآمد سیستم هشدار حريق دچار مشکل نخواهد شد، بلکه می‌توان بعد از برقراری مجدد ارتباط، داده‌هایی که به سرور اصلی ارسال نشده بوده را ارسال کند. وظایفی که می‌توان برای گذرگاه‌ها درنظر گرفت به مسئله مطرح شده بستگی خواهد داشت. این وظایف براساس نیازهایی مانند قابلیت اطمینان^۲، امنیت اطلاعات^۳، راهاندازی سامانه به شکل محلی^۴ و ... تعریف خواهند شد.

۱-۲-۲ پروتکل‌های ارتباطی

بر روی لایه‌های معرفی شده در بخش قبلی، لایه چهارم به نام انتقال^۵ قرار دارد. دو استاندارد مهم در این لایه استاندارد های TCP^۶ و UDP^۷ وجود دارد.

استاندارد UDP بدون نیاز به برقراری یک رابطه و ارتباط^۸، تنها با استفاده از آدرس اینترنتی که در لایه سوم مشخص می‌شود، می‌تواند داده‌هایی را ارسال کند یا منتظر دریافت داده‌ای بماند. این پروتکل از رسیدن داده ارسالی اطمینان حاصل نمی‌کند و ممکن است بسته ارسالی هرگز به مقصد نرسد.

برخلاف استاندارد UDP، استاندارد TCP یک ارتباط دوطرفه^۹ را برقرار می‌کند. این پروتکل به ازای تمامی بسته‌های ارسالی اطمینان حاصل می‌کند که بسته به دست مقصد رسیده باشد. علاوه بر آن، صحت و عدم تغییر اطلاعات ارسالی را نیز تضمین می‌کند. به دلیل الگوریتم‌های نسبتاً پیچیده‌ای که در این استاندارد استفاده می‌شود، استاندارد TCP نسبت به استاندارد UDP مقداری سنگین‌تر است و باید در استفاده از آن محظوظ بود. بر روی لایه انتقال، لایه اپلیکیشن^{۱۰} را داریم. برای اینترنت اشیا استانداردهای لایه اپلیکیشن بسیار زیاد هستند. ما از بین آن دو مورد متفاوت و مهم را مورد بررسی قرار خواهیم داد.

¹Buffer

²Reliability

³Data Security

⁴Localized Platform

⁵Transport Layer

⁶Transmission Control Protocol

⁷User Datagram Protocol

⁸Connectionless

⁹Connection Oriented

¹⁰Application

پروتکل CoAP

پروتکل CoAP^۱ [۲۰] یک نسخه سبک و ساده از استاندارد HTTP^۲ است که در سال ۲۰۱۳ طراحی شده است. برخلاف استاندارد HTTP که از TCP استفاده می‌کند، در این استاندارد از UDP استفاده می‌شود. هدرها^۳ نقش کلیدی را در بسته‌های داده ایفا می‌کنند. با کمک اطلاعاتی که در هدرها وجود دارد، هر لایه‌ای که پیش‌تر معرفی کردیم می‌تواند کار خود را بدرستی انجام دهد. در اینترنت اشیا چون حجم داده‌های ارسالی به ازای هر بسته بسیار کم است، اندازه هدرها مهم می‌شود. بسته‌ای را فرض کنید که شامل یک داده ۵۰ بایتی باشد اما اندازه هدر آن ۱۵۰ بایت باشد! در این حالت، دستگاه ما ۷۵ درصد زمان و انرژی اش را صرف ارسال هدر می‌کند تا صرف ارسال داده اصلی. اندازه هدر در CoAP کمتر از ۱۰ بایت است در صورتی که متوسط آن در HTTP برابر ۸۰۰ بایت خواهد بود[۱۶]

هر بسته‌ای که از طریق پروتکل HTTP جابه‌جا شود، یک نوع دارد. بسته‌ی ارسالی از نوع GET، POST، PUT و یا ... هستند. تعداد نوع‌های بسته‌ها در پروتکل HTPP زیاد است و برای همین پیاده‌سازی و استفاده از آن در دستگاه‌های با منابع محدود مناسب نخواهد بود. اما نوع بسته‌ها در CoAP به موارد GET، POST و DELETE محدود شده است که خود باعث کاهش پیچیدگی‌های پیاده‌سازی خواهد شد.

استانداردهای HTTP و CoAP از ساختار درخواست^۴ و پاسخ^۵ استفاده می‌کنند. در این ساختار مشتری^۶ درخواستی را برای خدمت‌رسان^۷ ارسال می‌کند. نوع درخواست در بسته ارسالی ذکر می‌شود. در نهایت خدمت‌رسان پاسخی را برای مشتری باز می‌گرداند.

همانطور که در بخش‌های پیشین بررسی کردیم، دستگاه‌ها در دو نقش متفاوت می‌توانند ظاهر شوند. حسگرهایی که اطلاعات محیطی را دریافت و با اعداد توصیف می‌کنند. یا محرک‌هایی که با دریافت دستورهایی بر محیط اطراف خود تاثیر می‌گذارند.

برای دسترسی به اطلاعات حسگر گذرگاه در نقش مشتری قرار می‌گیرد و با فرستادن درخواستی از نوع GET داده‌های خواسته شده را در پاسخ، دریافت می‌نماید. برای ارسال دستوری جدید می‌توان همان ساختار قبلی اما اینبار از درخواستی با نوع PUT استفاده کرد و دستور را به همراه بسته برای دستگاه که نقش خدمت‌رسان را دارد ارسال کرد. دستگاه می‌تواند انجام درست یا وجود مشکلی برای به پایان رساندن دستور را به عنوان پاسخ برای گذرگاه ارسال کند.

¹Constrained Application Protocol

²Hyper Text Transfer Protocol

³Header

⁴Request

⁵Response

⁶Client

⁷Server

معمولاً اطلاعات دستگاهها باید در بازه‌های زمانی کوتاهی دوباره درخواست شوند تا داده‌های موجود بروزرسانی گردد. برقراری ارتباط دوطرفه به ازای بازه‌های زمانی کوتاه باعث مصرف انرژی بالایی در دستگاه می‌خواهد شد. CoAP در درخواست نوع GET گزینه‌ای دارد به نام Observe. در صورتی که این گزینه فعال باشد، می‌توان یک بازه زمانی را در بسته ارسالی به دستگاه قرار داد و سپس ارتباط ایجاد شده دیگر توسط دستگاه بسته نمی‌شود، بلکه به ازای مدت زمان تعريف شده داده‌های درخواست شده از دستگاه به گذرگاه ارسال خواهد شد.

پروتکل UDP تضمینی برای کیفیت ارتباط بین دو نقطه ندارد. استاندارد CoAP بخارط استفاده از بستر UDP گزینه‌هایی را درنظر گرفته تا کیفیت سرویس^۱ را افزایش دهد. به عنوان مثال با قرار دادن گزینه Con-firmable در بسته‌های ارسالی، گیرنده پس از دریافت آن‌ها باید بسته‌ای را برای فرستنده ارسال کند تا فرستنده از رسیدن بسته ارسالی خود مطمئن گردد. عدم قرار دادن گزینه Confirmable در بسته ارسالی، باعث می‌شود تاییدیه‌ای از طرف گیرنده برای فرستنده ارسال نشود.

ساختار درخواست و پاسخی که در اینجا معرفی کردیم و توسط پروتکل CoAP پیاده‌سازی شده است برای ساختار اینترنت اشیا که بر پایه رویداد^۲ است، مناسب نیست[۲۲][۱۷]. همچنین پروتکل CoAP در ایجاد ارتباطات ماشین با ماشین سریار زیادی را برای دستگاه می‌تواند ایجاد کند. زیرا هر دستگاه برای برقراری ارتباط با دستگاهی دیگر نیاز به برقرار کردن یه ارتباط جدید خواهد داشت. همچنین ساختاری که با استفاده از پروتکل CoAP بتوان پیام‌های چندگانه^۳ و همگانی^۴ ارسال و دریافت نمود در نظر گرفته نشده است.

پروتکل MQTT

استاندارد MQTT^۵ [۱۹] عمر طولانی تری نسبت به CoAP دارد و در سال ۱۹۹۹ طراحی شده است. طراحی این استاندارد از ابتدا مطابق با نیازهای اینترنت اشیا بوده است. MQTT برخلاف CoAP از TCP استفاده می‌کند. در نظر طراحان آن پروتکل TCP نیازی به جایگزینی نداشته اما استفاده از مقاهم پروتکل HTTP یعنی TCP می‌کند. ساختار درخواست و پاسخ، به نظر اشتباہ بوده است. بیان این نکته ضروری است که MQTT با وجود استفاده از TCP عملکرد بسیار خوبی دارد. MQTT می‌تواند برای دستگاه‌هایی با ۶۴ کیلوبایت رم پیاده‌سازی و استفاده شود.[۱۸]

استاندارد MQTT از مفهومی به نام انتشار/اشتراک^۶ استفاده می‌کند که شامل اجزای زیر می‌باشد.

¹Quality of Service

²Event Based

³Multicast

⁴Broadcast

⁵Message Queue Telemetry Transport

⁶Publish/Subscribe

- یک واسط^۱ که اطلاعات را در موضوع‌های^۲ مختلفی نگه‌داری می‌کند.
- دستگاهی که اطلاعات جدیدی دارد می‌تواند آن‌ها به واسطه تحویل بدهد. این دستگاه نقش ناشر^۳ را در این مفهوم بازی می‌کند. موضوع اطلاعات منتشر شده از ناشر باید توسط خودش مشخص گردد تا واسطه بتواند اطلاعات جدید را بدرستی دست‌بندی نماید.
- مشترک^۴ نیز با گرفتن اشتراک موضوعی می‌تواند اطلاعات جدیدی که تحت آن موضوع به واسطه رسیده‌اند را دریافت نماید. واسطه به محض دریافت اطلاعات ارسال شده توسط ناشر آن‌ها را در اختیار مشترک قرار خواهد داد.

پروتکل MQTT به علت ساختار انتشار/اشتراک به خوبی می‌تواند برای اینترنت اشیا که ساختاری بر پایه رویداد دارد استفاده شود. اتصال ماشین‌های دیگر نیازی به ایجاد ارتباط^۵ جدید که منجر به افزایش سریار می‌شود، نخواهد داشت. با استفاده از این پروتکل دستگاه می‌تواند با ارسال پیام خود در موضوعی خاص به واسطه با دستگاه دیگر به برقراری ارتباط بپردازد. ارسال و دریافت پیام‌های چندگانه و همگانی نیز بسیار ساده خواهد بود. با مشترک شدن چند ویا تمامی دستگاه‌ها در موضوعی خاص می‌توان پیام‌های چندگانه ویا همگانی ارسال نمود.

از دیگر مزیت‌های این پروتکل، می‌توان به عدم نیاز به ارسال مجدد داده‌ها توسط دستگاه‌ها اشاره کرد. واسطه، که معمولاً یک کامپیوتر قدرتمند با منابع نامحدود در نظر گرفته می‌شود، وظیفه نگهداری از داده‌ها و ارسال و دریافت داده‌ها برای تمام دستگاه‌ها بر عهده دارد. به همین دلیل نسبت به CoAP پیاده‌سازی سبک‌تری خواهد داشت.

با استفاده از این پروتکل بخشی از نیاز پردازشی، حافظه، توان مصرفی و پیچیدگی‌های طراحی و پیاده‌سازی کاهش یافته ویا به واسطه منتقل می‌شود. توسعه پروره با MQTT ساده‌تر از استفاده از CoAP است. برای اضافه کردن یک دستگاه جدید به سیستم موجود و برقراری ارتباط به دستگاه‌های دیگر و سرور اصلی، کافی است دستگاه مشترک موضوع‌هایی که به اطلاعات‌شان نیاز دارد، بشود. اگر دستگاه اطلاعات جدیدی تولید می‌کند با در نظر گرفتن موضوعی جدید در واسطه برای این اطلاعات و ارسال آنها را به واسطه، می‌توان داده‌ها را برای دیگر دستگاه‌ها در دسترس قرار داد.

¹Broker

²Topic

³Publisher

⁴Subscriber

⁵Connection

یکی از ویژگی های جالب MQTT ویژگی به نام وصیت‌نامه^۱ است. هر ناشری که در حال ارسال اطلاعاتی در موضوع خاصی به واسطه است، اگر دچار مشکلی شود و ارتباط خود را با واسطه از دست بدهد واسطه از وصیت اون به عنوان آخرین پیام استفاده خواهد کرد. وصیت پیامی است که ناشر در زمان اتصال به واسطه او اعلام می‌کند. مثلاً دستگاه فشارسنجی را در نظر بگیرید که در اثر اتمام انژری الکتریکی باتری خاموش شده است. وصیت این ناشر میتواند «باتری تمام شده است. باتری را تعویض نمایید.» باشد.

استفاده از پروتکل TCP در لایه انتقال توسط پروتکل MQTT باعث می‌شود که یک ارتباط مطمئن داشته باشیم که تضمین‌کننده صحت و یکپارچگی اطلاعات در مقصد خواهد بود. استفاده از TCP علاوه بر مزیت، معایبی نیز دارد. به خاطر برقراری ارتباط دوطرفه، هر طرف مقدار زمانی را در نظر گرفته و اگر داده‌ای جدید قبل از پایان زمان در نظر گرفته شده، دریافت نشود ارتباط قطع خواهد شد. در نتیجه هر ارتباط TCP نیاز به ارسال اطلاعات در بازه‌های زمانی خاصی دارد تا ارتباط قطع نشود. این کار به معنا خوابهای محدود و کوتاه برای دستگاهها و در بعضی مواقع مصرف انژری بیشتر است. هر لینک ارتباطی TCP نیاز به نگهداری وضعیت دارد و این به معنای نیاز به حافظه است. به علت پیچیدگی ساختار TCP نسبت به UDP میزان حافظه مورد برای یک ارتباط TCP بیشتر از یک ارتباط UDP خواهد بود.

همانطور که بررسی کردیم، استانداردهای CoAP و MQTT دو راهکار کاملاً متفاوت را در پیش گرفته بودند. هر دو استانداردهای موفقی هستند که به شکل گسترده از آنها استفاده می‌شود.

۳-۲ سرورها

وظیفه اصلی سرور در اینجا نگهداری از اطلاعات و دردسترس قرار دادن آنها خواهد بود. داشتن یک سرور مرکزی به دلیل تعداد بالا دستگاهها سامانه را کند خواهد کرد که یکی از راهکارهای مناسب استفاده از گذرگاه‌ها خواهد بود. همچنین نحوه نگهداری از داده‌های بسیار عظیم، که هر ثانیه در حال تولید هستند نیز بسیار سخت خواهد بود.

دریافت اطلاعات از دستگاهها و ارسال دستورات جدید به آن‌ها توسط برقراری یک ارتباط دوطرفه TCP و استفاده از پروتکل MQTT امکان‌پذیر خواهد شد. در اینجا نیز با حسب مسئله‌ای که درحال طراحی راه حل آن هستیم، باید موضوع‌ها و نحوه ارتباط دیگر دستگاهها با این موضوع‌ها را مشخص کنیم. برای نگهداری داده‌ها می‌توان انواع پایگاه‌داده‌ها^۲ موجود استفاده کرد. علاوه‌بر پایگاه‌داده‌ها برای افزایش

¹Last Will

²Database

توان سرور می‌توان از ساختارهای ابری^۱ و یا از رایانش خوشه‌ای^۲ بهره برد.

۴-۲ داشبوردها

داشبورد در حقیقت بخشی از سامانه است که با کاربران در ارتباط خواهد بود. به عنوان مثال، یک اپلیکیشن تلفن همراه هوشمند که می‌توان لامپ‌های خانه را از طریق آن کنترل کرد و براساس متحوالی یخچال، لیست خریدی را به شما پیشنهاد بدهد. داشبوردها به اپلیکیشن‌های موبایل ختم نمی‌شوند. سیستم‌های هوشمند، برنامه‌های تحت وب^۳ و تمام نرم‌افزارهایی که از داده‌های ذخیره شده در سرور استفاده کنند، در دسته‌بندی داشبوردها قرار خواهند گرفت.

¹Cloud

²Cluster Computing

³Web Applications

فصل سوم

سامانه اینترنت اشیا به کمک مفهوم انتشار/اشتراک

۱-۳ بررسی و طراحی

بررسی سامانه‌های مشابه مانند [۲۲]، [۲۳] و [۲۴] به طراحی و بهینه‌سازی ساختار موجود کمک کرد.

در طراحی انجام شده چهار سطح مختلف وجود دارد.

- دستگاه‌ها^۲ که خود در برگیرنده حسگرها و عملگرها و اجزا ارتباطی مانند WiFi و بلوتوث هستند.
- گذرگاه‌ها که برای مدیریت، جمع‌آوری و ساده‌سازی اطلاعات در نظر گرفته شده اند.
- سرور اصلی اطلاعات را از گذرگاه دریافت، ذخیره و نگهداری می‌کند.
- یک داشبورد که کاربر از طریق آن بتواند اطلاعات را ببیند و دستگاه‌های مختلف را کنترل نماید.

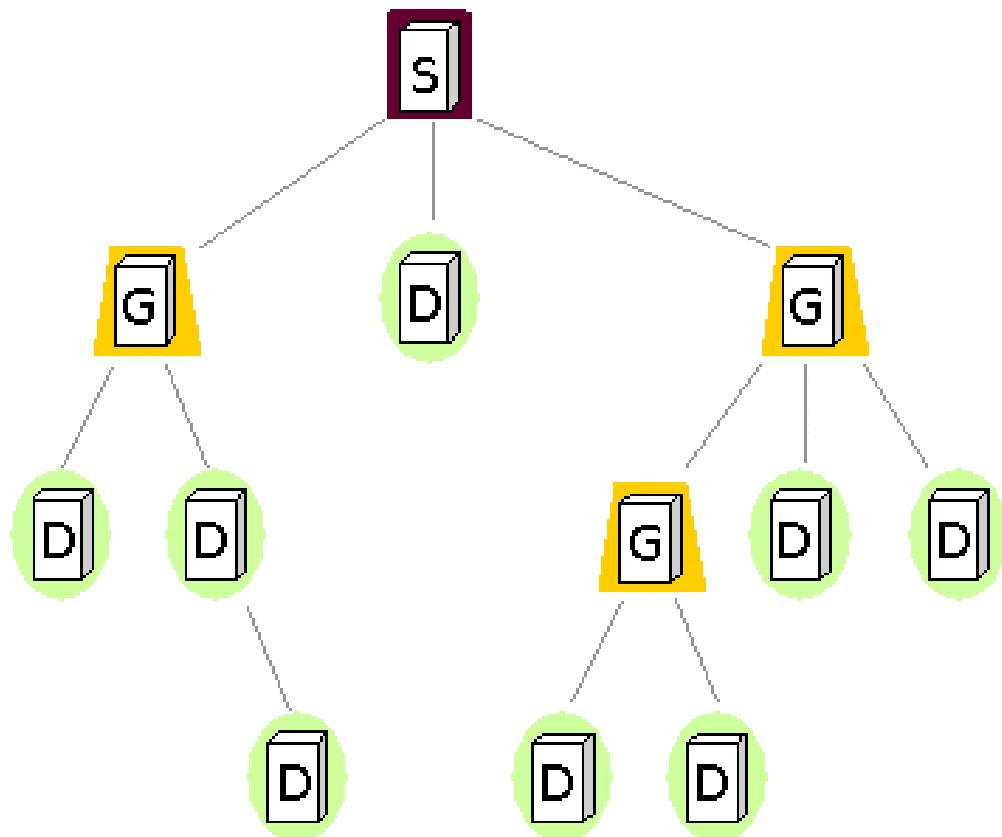
مفاهیم اصلی برای طراحی سامانه مورد نظر ما در این مقاله شامل گروه‌ها^۳، شناسه جامع و یکتا^۴، تنظیمات^۵ و ساختار شبکه‌های درختی می‌باشند.

²Devices

³Groups

⁴Universally Unique Identifier

⁵Configurations



شکل ۳-۱: ساختار درختی سامانه طراحی شده در این مقاله.

شکل کلی ارتباطات در این سامانه شبیه به یک درخت خواهد شد. در شکل ۳-۱ نمونه‌ای از اتصالات بین اجزا سامانه را مشاهده می‌کنید. این شکل شامل سه رنگ قرمز، زرد و سبز است که به ترتیب نشانده‌ند سرور، گذرگاه‌ها و دستگاه‌ها می‌باشد. اجزا مختلف این شبکه نود^۱ نام دارند. در این ساختار درختی، هر نود به نود بالاتری متصل است که نود پدر^۲ نامیده می‌شود. نودهای زیرمجموعه یک نود، نود فرزند^۳ نامیده می‌شوند. در سامانه مورد نظر هر نود دقیقاً یک پدر خواهد داشت و تنها نود بدون پدر نود سرور است. هر نود می‌تواند صفر، یک یا چند فرزند داشته باشد. ارتباط بین اجزا این سامانه بسیار انعطاف‌پذیر خواهد بود. هر دستگاه می‌تواند به دستگاهی دیگر، گذرگاه یا سرور متصل شود. هر گذرگاه نیز می‌تواند به گذرگاهی دیگر یا سرور متصل شود. در حقیقت در این سامانه هر نود چند نقش را همزمان ایفا می‌کند. نقش‌های واسط، ناشر و مشترک را نسبت به نودهای فرزندش و نقش‌های ناشر و مشترک را نسبت به نود پدر خود دارد.

دستگاه‌ها در سطح نود پدرشان داخل گروه‌های مختلف دسته‌بندی می‌شوند و داده‌های چند دستگاهی که در گروه قرار دارند با روش‌هایی مانند میانگین‌گیری داده‌ها و یا استفاده از حداقل یا حداقل‌تر به یک داده کاهش

¹Node

²Parent Node

³Child Node

پیدا می‌کند که به آن نماینده داده‌های آن گروه^۱ گفته می‌شود. از اهداف اصلی استفاده از گروه‌ها می‌توان کاهش از دیاد و پیچیدگی داده‌ها در نودهای بالاتر و درنهایت در سرور را نام برد. با این روش داده‌های ارسالی به نودهای بالاتر کمتر، دقیق‌تر و جامع‌تر خواهند بود.

در این طراحی هر دستگاه، گروه، گذرگاه و سرور از طریق یک شناسه جامع و یکتا شناخته خواهند شد. به همراه هر شناسه که به نودهای سامانه تخصیص داده می‌شود، مجموعه‌ای از تنظیمات^۲ را در نظر می‌گیریم. تنظیمات نودها شامل اطلاعاتی مانند شناسه نود، شناسه گروهی که نود در آن قرار دارد، توضیحی در مورد نود^۳، نوع نود^۴ و ... می‌باشد. با استفاده از این تنظیمات می‌توان اطلاعات منتشر شده توسط نود را دسته‌بندی نمود و اطلاعات درخواستی‌اش را به او رساند. همچنین ویژگی‌هایی مانند جست‌وجو و پیدا کردن حسگرها یا عملگرهای خاص بسیار ساده خواهد شد. مشخص کردن دقیق این تنظیمات به مسئله مطرح شده بستگی دارد. حداقل تنظیماتی وجود خواهد داشت که از این سامانه به عنوان راه حل بهره می‌برند، باید وجود داشته باشد.

تنظیمات نودها، که توسط شرکت سازنده مشخص می‌شوند و توسط کاربر قابلیت شخصی‌سازی دارند، در ایجاد ارتباط و استفاده از آن دستگاه یا گذرگاه اهمیت بسیاری دارند. دستگاه‌ها و گذرگاه‌ها در هنگام متصل شدن به دستگاه، گذرگاهی دیگر و یا سرور تنظیمات‌شان را در پیامی با نوع «سلام»^۵ به نود پدر خواهند فرستاد. در نتیجه هر نود از ساختار درختی نودهای زیرمجموعه‌اش آگاه خواهد بود. ارتباطات ایجاد شده بین نودهای سامانه یک ارتباط با پروتکل TCP برقرار می‌شود. در نتیجه در صورت ایجاد اختلال در ارتباط و یا قطع آن، نودها مطلع خواهند شد. در صورتی نودی از سامانه ارتباطش را با نود پدرش از دست بدهد، نود پدر با ارسال پیامی از نوع «خداحافظ»^۶ اجداش را از قطع شدن این نود از سامانه مطلع خواهد کرد.

دربافت اطلاعات از یک حسگر ملزم به دانستن شناسه دستگاهی است که اطلاعات آن حسگر را وارد سامانه می‌کند. شناسه دستگاه‌های متصل به سامانه از طریق پیام با نوع «سلام» برای هر کاربر در سرور مشخص شده است. برای هر نود در نود پدرش دو موضوع دیگر با نام INF:UUID و CMD:UUID وجود دارد. اطلاعات و گزارشات هر دستگاه با برچسب INF^۷ به همراه شناسه آن دستگاه به عنوان موضوع مشخص می‌شوند. جهت حرکت اطلاعات از دستگاه به سرور است و در ساختار درختی این سامانه هر نود فرزند اطلاعات با تگ INF را در نود پدرش منتشر می‌کند. همچنین هر نودی می‌تواند مشترک اطلاعات دستگاهی دیگر که با تگ INF

¹Group Indicator

²Configuration

³Node Description

⁴Node Type

⁵Hi

⁶Bye

⁷Information

مشخص شده است نیز باشد. تمامی دستورات با تگ **CMD^۱** مشخص می‌شوند که جهتی از سمت سرور به دستگاه خواهد داشت. برای بدست آوردن این اطلاعات هر نود متشرک موضوعی با تگ **CMD** در نود پدرسخواهد شد. حال گذرگاه تمامی اطلاعات منتشر شده از حسگرهایی که شناسه گروه یکسانی دارند را جمع کرده و عملیاتی نظیر میانگین‌گیری را روی داده‌ها انجام می‌دهد و در نهایت یک داده را به عنوان نماینده گروه در سرور با نام **INF:UUID^۲** ارسال می‌کند. حال کاربر با گرفتن اشتراک موضوع **INF:UUID** در سرور می‌تواند به اطلاعات منشر شده در هر گروه دسترسی پیدا کند. برای دسترسی به اطلاعات یک نود خاص نیز کافی است شناسه دستگاه و شناسه گروه آن نود با یکدیگر برابر باشد.

فرستادن دستور به دستگاه‌ها نیز براساس شناسه دستگاه شکل می‌گیرد. کاربر دستور خود در پیامی شامل شناسه دستگاه مورد نظر قرار داده و در موضوع دستورات^۳ در سرور منتشر می‌کند. با استفاده از پیام‌های اویله «سلام» و فرستادن تنظیمات از دستگاه تا سرور، تمامی دستگاه‌ها، گذرگاه‌ها و سرور از اینکه نوادگان هر کدام از فرزندانشان چه شناسه و ویژگی‌هایی دارند مطلع خواهند شد. حال سرور با بررسی شناسه دستگاه، دستور را در موضوع **CMD:UUID** گذرگاهی که از اجداد دستگاه موردنظر باشد، باز نشر می‌دهد. این روند آنقدر تکرار می‌شود تا در نهایت دستور به دستگاه برسد.

حداقل تنظیمات مورد نیاز برای کارکرد درست سامانه شامل موارد زیر خواهد بود.

- شناسه جامع و یکتای دستگاه که غیر قابل تغییر می‌باشد.
- شناسه گروهی دستگاه عضوی از آن خواهد بود. چند دستگاه می‌توانند شناسه گروهی یکسان داشته باشند. همچنین یک دستگاه می‌تواند با فرستادن چند پیام «سلام» مختلف عضو گروه‌های مختلفی باشد.
- نوع عملیاتی که در گروه باید بر روی تمامی داده‌ها انجام شده و حاصل نهایی به عنوان نماینده ارسال شود. این مقدار برای اعضای یک گروه باید برابر باشد.
- توضیحی کوتاه در مورد دستگاهی که قرار است به سامانه متصل شود.

تنظیمات دیگر مانند فاصله زمانی انتشار اطلاعات جدید توسط دستگاه اختیاری بوده و الزامی ندارند. به علت استفاده از ساختار انتشار/اشتراک استفاده کننده از این سامانه می‌تواند موضوع‌های دیگری با روند کار خاص یا گذرگاهی با عملکرد متفاوت ایجاد کند و با رعایت کردن ساختار معرفی شده آن را بکار گرفت.

¹Command

²Group UUID

³Commands

حال با اسفاده از مفاهیم معرفی شده در ساختار بالا، که ارتباطی دوطرفه با استفاده از پروتکل MQTT بین کاربر و هر دستگاه برقرار می‌کند، می‌توان مسئله‌ای را به عنوان نمونه بررسی و پیاده‌سازی کرد.

۲-۳ پیاده‌سازی

امروزه دسترسی به اینترنت و موتورهای جست‌وجو نقش بسیار مهمی در پیشرفت و آموزش ایفا می‌کنند. آینده‌ای را پیش‌رو داریم که افراد خود به ساخت و ابداع خواهند پرداخت. برنامه‌سازی بسیار ساده‌تر شده و از یک تخصص به دانش عمومی تبدیل گشته است. به همین دلیل شاهد تولید شدن بُردها^۱ و سیستم روی تراشه^۲‌های بسیاری هستیم. تصمیم گیری برای انتخاب بُردهای مورد استفاده در پیاده‌سازی با بررسی [۲۱] بوده است.

در شکل ۲-۳ شاهد محصولات شرکت Arduino^۳ هستیم. پروژه‌های این شرکت نرم‌افزار و سخت‌افزارهای متن‌بازی است که با قیمت بسیار ناچیزی می‌توان خریداری کرد. شرکت‌ها و پروژه‌هایی از این دست باعث شده‌اند که برنامه‌سازی و ساخت ابزارها ساده‌تر بشود و افراد مختلف را علاقه‌مند کرده است. در شکل ۳-۱ شاهد محصول شرکت Raspberry Pi هستیم. این بُرد نسبت به محصولات شرکت Arduino پیچیدگی و امکانات بیشتری دارد. بر روی بُرد Raspberry Pi ۳ به راحتی سیستم عامل محبوب و کارآمد لینوکس^۴ قابل نصب و اجرا خواهد بود.

می‌خواهیم شرایط واقعی که در اینترنت اشیا به طور معمول پیش می‌آید را شبیه‌سازی کنیم. برای اینکار از چند بُرد مختلف، حسگرها و عملگرهایی استفاده می‌کنیم. سه بُرد Arduino Micro Pro و یک بُرد Uno نقش کنترل‌کننده حسگرها و عملگرها را خواهند داشت و در حقیقت همان دستگاه‌های مطرح شده در فصل دوم هستند. این بُردها توان مصرف پایینی دارند. از پردازنده ATmega32 با سرعت 16MHz استفاده می‌کنند. یک نمونه از این بُردها در شکل ۳-۳ آورده شده است.

دستگاه‌های ما باید بتوانند به شبکه متصل شوند. برای ایجاد ارتباط از تراشه‌ای به نام ESP8266، تولید شده توسط کمپانی Espressif Systems، استفاده می‌کنیم. این تراشه را می‌توانید در شکل ۳-۴ مشاهده نمایید. این تراشه در حقیقت لایه‌های اول تا چهارم مطرح شده در دو فصل اول را برای ما فراهم می‌کند. نحوه اتصال این تراشه به هر کدام از بُردها را می‌توانید در شکل ۳-۴ مشاهده نمایید.

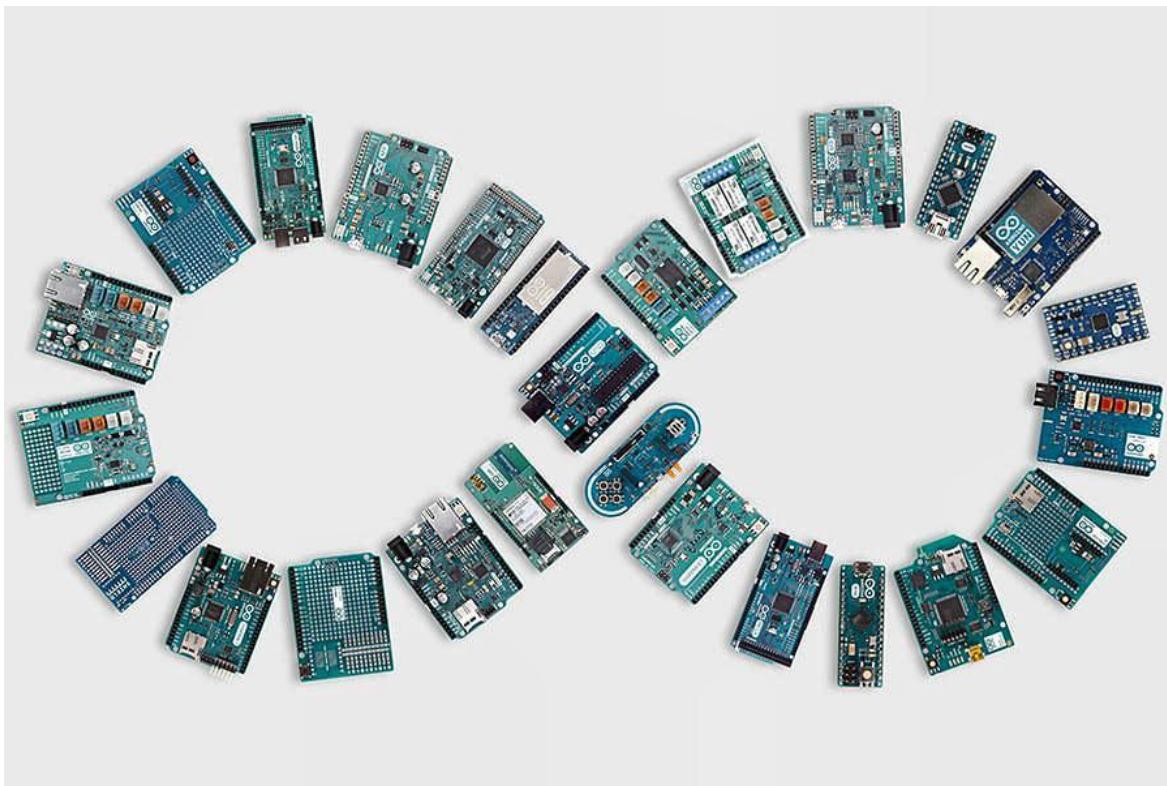
سیستم ما برای درستی شبیه‌سازی نیاز به چند حسگر و عملگر نیز دارد. از سه دسته حسگر استفاده خواهیم داشت که شامل موارد زیر هستند و در شکل ۳-۵ نحوه اتصال آن‌ها به دستگاه‌های Arduino به تصویر کشیده

¹Board

²System on a Chip(SoC)

³<http://arduino.cc>

⁴Linux



شکل ۲-۳: محصولات مختلف شرکت Arduino که به شکل آرم این شرکت چیده شده‌اند.

شده است.

- سنسور نور^۱ که میزان نور محیط را بررسی کرده و به شکل یک عدد می‌تواند در اختیار دستگاه قرار دهد.
- سنسور دما^۲ که میزان دما محیط را با مقیاس عددی به دستگاه گزارش می‌دهد.
- خواننده شناسه از طریق موج‌های رادیویی^۳ که شناسه ذخیره شده در کارت‌های مخصوص این تکنولوژی را به دستگاه ارسال می‌نماید.

و همچنین از دو دسته عملگر زیر را برای تکمیل شبیه‌سازی استفاده خواهیم نمود. نحوه اتصال نمایشگر در شکل ۳-۶ قابل بررسی است.

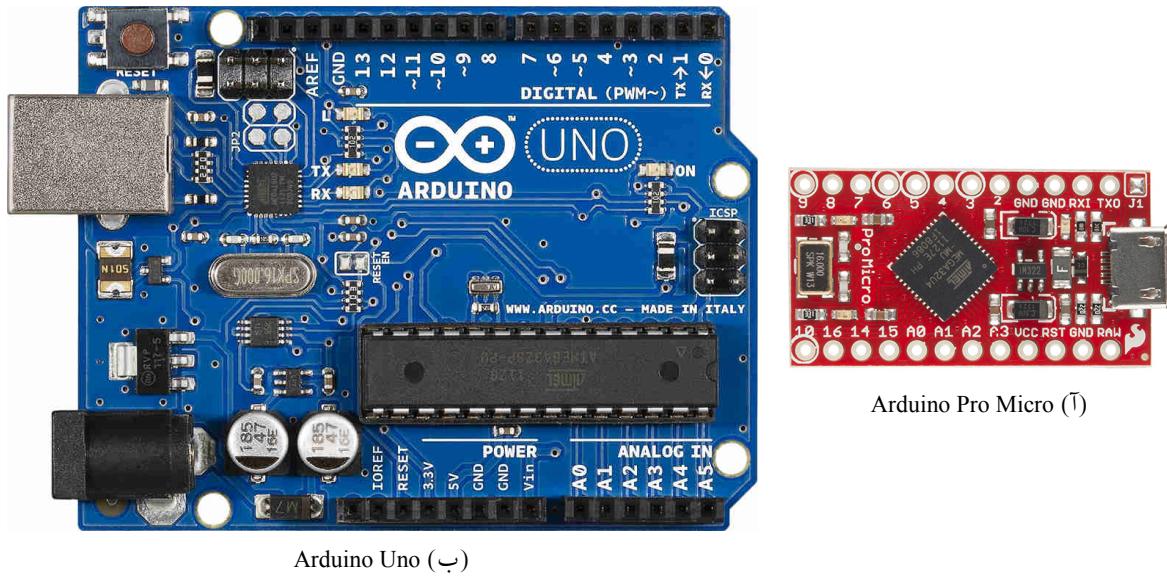
- نمایشگر که اطلاعات مختلف مانند میزان نور، دما و شناسه آخرین کارت استفاده شده را نمایش می‌دهد.
- یک لامپ^۴ که وضعیت دریافت صحیح کارت شناسه را اعلام خواهد کرد.

¹Light Dependent Resistor(LDR)

²LM35 Precision Centigrade Temperature Sensors

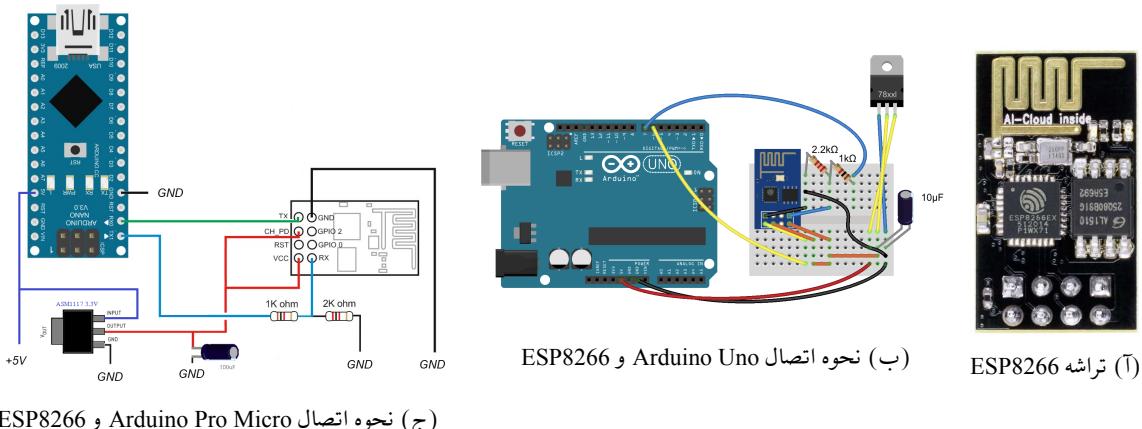
³RFID Reader RC522

⁴LED



Arduino Uno (ب)

شکل ۳-۳: بردهای شرکت Arduino با طراحی متن باز.



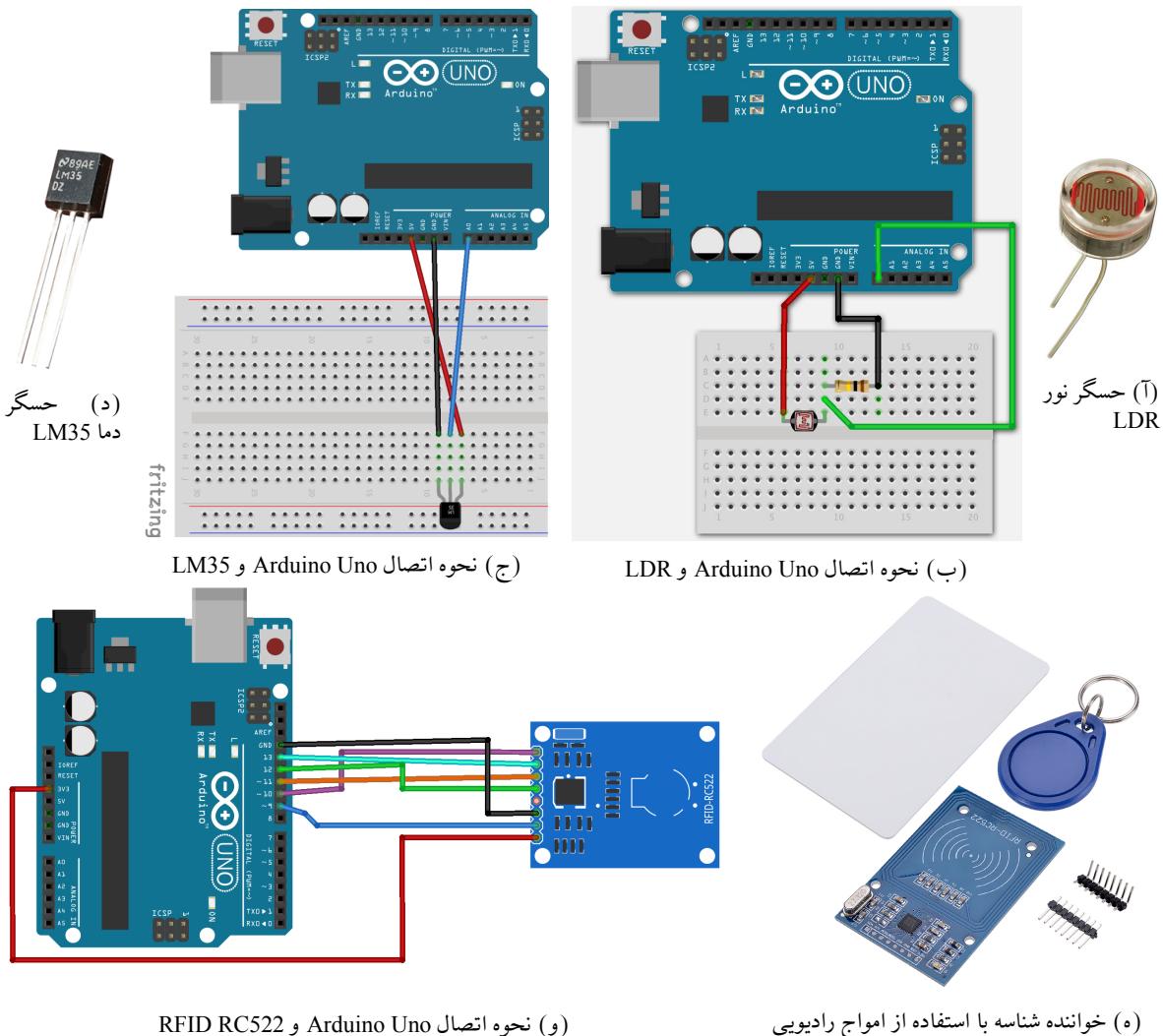
(ج) نحوه اتصال ESP8266 و Arduino Pro Micro

شکل ۳-۴: اتصال تراشه ESP8266 به بردهای Arduino

یک عدد Raspberry Pi 3 نیز به عنوان گذرگاه استفاده خواهیم نمود. اتصال تمامی دستگاهها به گذرگاه از طریق شبکه بیسیم WiFi است. هر دستگاهی بر اساس اطلاعات تولیدی و اطلاعات درخواستی اش ناشر و یا متشرک موضوعی در گذرگاه خواهد شد.

سپس گذرگاه از طریق اینترنت داده های خود را به سرور ارسال می نماید. داده ها در سرور ذخیره و نگهداری می شوند. داشبوردهای متن بازی باری ارسال و دریافت اطلاعات وجود دارند. یکی آن ها داشبورد تحت وب و متن بازی به نام FreeBoard^۱ است. این داشبورد امکان استفاده بر روی لپتاپ، تبلت و موبایل ها خواهد بود. داشبورد دیگری به نام MQTT Dash یک اپلیکیشن موبایل است که می توان برای کار کردن با واسطه های پروتکل MQTT از آن استفاده نمود.

¹ freeboard.io



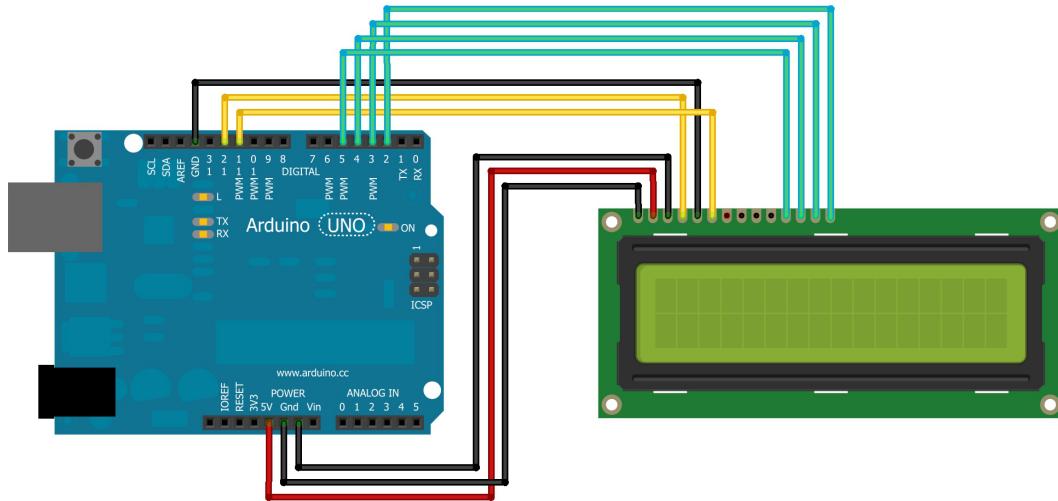
شکل ۳-۵: اتصال حسگرها به Arduino

سناریو پیاده‌سازی شده شامل بخش‌های زیر است.

- دستگاه اول شامل یک حسگر دما، نور، دو لامپ به رنگ‌های سبز و قرمز و خواننده شناسه از طریق امواج رادیویی است. در صورتی که اطلاعات کارتی خواننده شود، شناسه ذخیره شده در کارت به عنوان داده جدید نشر پیدا خواهد کرد. سپس دستگاه منتظر دریافت دستوری برای روشن کردن چراغ سبز به معنای صحت کارت استفاده شده ویا چراغ قرمز به معنای عدم صحت کار استفاده شده خواهد ماند و با دریافت دستور چراغ مورد نظر را روشن خواهد کرد.

- دستگاه دوم و سوم هر کدام شامل یک حسگر دما و نور خواهد بود.

- دستگاه چهارم شامل یک نمایشگر خواهد بود که اطلاعات دما، نور و آخرین شماره کارت خواننده شده را نمایش خواهد داد.



شکل ۳-۶: نحوه اتصال نمایشگر و Arduino

داده‌های حسگرهای دما و نور هر کدام در گروهی می‌روند و متوسط مقداری که از آن‌ها به دست می‌آید توسط گذرگاه به سرور ارسال می‌شود. همچنین کاربر می‌تواند تایید را رد یک کارت را اعلام نماید و ارتباط نمایشگر با دیگر دستگاه‌ها یک ارتباط ماشین با ماشین خواهد بود.

در ابتدا اتصالات و ارتباطات موجود را بررسی و برای هر کدام کدی نمونه برای کار کردن با آنها خواهیم دید. اتصالات در سطح دستگاه‌ها به موارد زیر شکسته می‌شوند.

- برقراری ارتباط Arduino با ESP8266 و LDR و LM35 و RFID-RC522 و نمایشگر کاراکتری
- مازول ESP8266 باید بتواند به یک نقطه دسترسی متصل شود و همچنین باید بتوان بر روی آن از پروتکل MQTT نیز استفاده نمود.

در سطح گذرگاه و سرور نیز باید بتوان ارتباطات و همچنین پروتکل MQTT را اجرا نمود. کدهای آردوینو به دو بخش کلی شکسته می‌شوند. در تابعی به نام `setup()` تمامی تنظیمات اولیه برای راهاندازی برد قرار می‌گیرد. در تابعی دیگر به نام `loop()` کدهای اصلی قرار خواهند گرفت. اجرای این کدها پس پایان تابع از سرگرفته خواهند شد. در این بخش به عنوان مثال کدهای بررسی و گرفتن اطلاعات از یک حسگر و ارسال آن قرار می‌گیرد. ابتدا با توجه به شکل ۳-۴ آردوینو را به ESP8266 متصل می‌نماییم. سپس با استفاده از کتابخانه متن باز WiFiEsp [۲۷] می‌توانیم از قابلیت‌های آن استفاده کنیم. به عنوان نمونه می‌خواهیم نقطه دسترسی^۱ ایجاد کنیم و پس از اتصال به آن بتوانیم یک صفحه وب شامل اطلاعات پایه A0 را مشاهده کنیم. برقراری ارتباط بین این دو سخت‌افزار از طریق ارتباط سریال خواهد بود که در آن آردوینو دستوراتی را به

^۱ Access Point

```

1 #include "WiFiEsp.h"
2
3 // Emulate Serial1 on pins 6/7 if not present
4 #ifndef HAVE_HWSERIAL1
5 #include "SoftwareSerial.h"
6 SoftwareSerial Serial1(8, 9); // RX, TX
7 #endif
8
9 char ssid[] = "Arduino";           // your network SSID (name)
10 char pass[] = "12345678";          // your network password
11 int status = WL_IDLE_STATUS;       // the Wifi radio's status
12 int reqCount = 0;                 // number of requests received
13
14 WiFiEspServer server(80);
15
16 // use a ring buffer to increase speed and reduce memory allocation
17 RingBuffer buf(8);

```

شکل ۳-۷: تنظیمات اولیه برای اتصال ماژول ESP8266 به برد آردوینو

ارسال می‌نماید و نتیجه‌اش را نیز از طریق ارتباط سریال دریافت خواهد کرد. بر روی برد دو پایه به نام‌های RX و TX وجود دارد. این دو پایه امکان برقراری ارتباط سریال را فراهم می‌کنند که به آن‌ها پایه‌های سخت‌افزاری برقراری ارتباط سریال گویند. ارتباط بین آردوینو و کامپیوتر نیز از طریق ارتباط سریال شکل می‌گیرد. اگر بخواهیم پیام‌هایی برای خطایابی و دنبال کردن اجرا برنامه‌های بارگزاری شده به کامپیوتر ارسال کنیم، باید پایه‌های دیگری را برای ارتباط ESP8266 و آردوینو انتخاب کنیم. در صورتی که بخواهیم ارتباط سریال از طریق دو پایه دیگر انجام شود باید از کتابخانه متن باز دیگری به نام SoftwareSerial استفاده کنیم. این کتابخانه متعلق به پروژه آردوینو می‌باشد.

مانند شکل ۳-۷ ابتدا کتابخانه‌های مورد نظر را به کد اضافه می‌کنیم و تنظیمات اولیه مانند نام و کلمه عبور نقطه را مشخص می‌کنیم. در تابع setup() دو ارتباط سریال، یکی برای فرستادن پیام به کامپیوتر و دیگری برای فرستادن دستورات و دریافت نتیجه آنها به ESP8266 را باز می‌کنیم. با معرفی سریال بازشده به تابع آغازگر^۱ کتابخانه WiFiEsp می‌توان از آن سخت‌افزار استفاده نمود و با فراخوانی تابع beginAP() نقطه دسترسی ایجاد خواهد شد. پس آن با استفاده از تابع printWiFiStatus() اطلاعات نقطه دسترسی ایجاد شده و صفحه وب در ارتباط سریال به کامپیوتر ارسال خواهد شد. کدهای مربوط به این بخش در شکل ۳-۸ مشخص شده است. در شکل ۳-۹ نیز تابع printWifiStatus() مشخص شده است. این تابع نام و آدرس اینترنتی نقطه دسترسی ایجاد شده و همچنین آدرس اینترنتی خدمت رسان ایجاد شده را با استفاده از ارتباط سریال به کامپیوتر ارسال می‌کند.

در تابع loop() دستگاه منتظر برقراری ارتباط می‌ماند و به محض درخواست مورد نظر پاسخی را توسط تابع

¹Initializer

```

1 void setup()
2 {
3     Serial.begin(115200); // initialize serial for debugging
4     Serial1.begin(9600); // initialize serial for ESP module
5     WiFi.init(&Serial1); // initialize ESP module
6
7     // check for the presence of the shield
8     if (WiFi.status() == WL_NO_SHIELD) {
9         Serial.println("WiFi shield not present");
10        while (true); // don't continue
11    }
12
13    Serial.print("Attempting to start AP ");
14    Serial.println(ssid);
15
16    // uncomment these two lines if you want to set the IP address of the AP
17    //IPAddress localIp(192, 168, 111, 111);
18    //WiFi.configAP(localIp);
19
20    // start access point
21    status = WiFi.begin(ssid, 10, pass, ENC_TYPE_WPA2_PSK);
22
23    Serial.println("Access point started");
24    printWifiStatus();
25
26    // start the web server on port 80
27    server.begin();
28    Serial.println("Server started");
29 }

```

شکل ۳-۸: ایجاد نقطه دسترسی و ارسال اطلاعات توسط ارتباط سریال به کامپیوتر

ارسال می‌کند و توسط این مازول به درخواست‌کننده برگردانده خواهد شد. در شکل ۳-۱۰ می‌توان کدهای مربوط به تابع `loop()` و `sendHttpResponse()` را مشاهده نمود.

حال نوبت به پیاده‌سازی پروتکل MQTT بر روی آردینو رسیده است. از پیاده‌سازی متن باز پروتکل PubSubClient به نام [۲۸] Nick O'Leary که توسط آقای Mosquitto [۲۹] انجام گرفته و بر روی دستگاه‌های آردینو نیز قابل استفاده است، بهره خواهیم برد. همچنین یک واسط متن باز به نام [۲۶] کارکرد این کتابخانه بر روی آردینو استفاده کردیم. تنظیمات اولیه در قطعه کد شکل ۱۱-۳ با گرفتن نام نقطه دسترسی، آدرس اینترنتی و پورت واسط و همچنین با مشخص کردن یک تابع برای دریافت پیام‌ها از موضوع‌هایی که در آنها مشترک شده‌ایم، انجام می‌شود. در قطعه کد شکل ۱۲-۳ پس از اتصال به واسط، در موضوع FromArduino دریافت پیام HelloWord! می‌کنیم و در موضوع ToArduino اشتراکی گرفته و منتظر دریافت پیام می‌ماند. تابع `callback()` وظیفه دریافت پیام‌ها را دارد.

حسگر LM35 با تغییر دما، اختلاف ولتاژ پایه خروجی اش نیز تغییر می‌کند. تغییر ولتاژ در واحدهای ۱۰mV رخ داده و برای تشخیص این تغییرات و درنهایت تغییرات دما، نیاز به استفاده از پایه‌های آنالوگ بُرد خواهیم داشت. این پایه‌ها می‌توانند اختلاف ولتاژ بین صفر تا پنج ولت را به ۱۰۰۰ بخش تقسیم کرده و حدود تفاوت

```

1 void printWifiStatus()
2 {
3     // print your WiFi shield's IP address
4     IPAddress ip = WiFi.localIP();
5     Serial.print("IP Address: ");
6     Serial.println(ip);
7
8     // print where to go in the browser
9     Serial.println();
10    Serial.print("To see this page in action, connect to ");
11    Serial.print(ssid);
12    Serial.print(" and open a browser to http://");
13    Serial.println(ip);
14    Serial.println();
15 }

```

شکل ۹-۳: اطلاعات ارسال شده از طریق ارتباط سریال به کامپیوتر

ولتاژ را گزارش دهند. Arduino Uno شش پایه و Arduino Pro Micro نیز چهار پایه آنالوگ دارند. پس از متصل کردن فیزیکی LM35 به یکی از این پایه‌ها با استفاده از شماره پایه و تابع (`analogRead()`) به راحتی می‌توان عددی بین ۰ تا ۱۰۰۰ دریافت کرد که نشان‌دهنده اختلاف ولتاژ است.

حسگر LDR یک مقاومت متغیر وابسته به نور است. هر چقدر شدت نور بیشتر باشد، مقاومتش کاهش پیدا کرده و هر چقدر شدت نور کمتر باشد مقاومتش بیشتر خواهد شد. ساختار و استفاده از حسگر نور نیز مشابه با حسگر دما خواهد بود. کافی است که مطابق شکل ۳-۵ آن را به آردوینو متصل کنید و در نهایت از تابع (`analogRead()`) مقدار خروجی این حسگر را بدست آورید.

راه‌انداختن نمایشگر نیز کار ساده‌ای خواهد بود. کافی است مطابق شکل ۳-۶ نمایشگر را به آردوینو متصل کنید و با کتابخانه متن‌باز LiquidCrystal آن را کنترل نمایید. نمونه‌ای از کد برای کار کردن با کتابخانه LiquidCrystal در شکل ۳-۱۳ نمایش داده شده است.

در انتهای می‌خواهیم نحوه اتصال خواننده شناسه از طریق امواج رادیویی به آردوینو را نیز بررسی کنیم. ابتدا هر کدام از پایه‌ها را بر اساس شکل ۳-۵ متصل نموده و سپس با استفاده از کتابخانه MFRC522 که متعلق به پروژه Arduino است، شناسه کارت را خواهیم خواند. ارتباط بین ماژول RFID-RC522 و آردوینو از طریق پروتکل SPI برقرار می‌شود. در شکل ۳-۱۴ کد مورد نظر ما قرار دارد.

در سطح گذرگاه و سرور از کتابخانه متن‌باز دیگری به نام HBMQTT [۳۰] استفاده خواهیم کرد. این کتابخانه بر اساس کتابخانه استاندارد asyncio برای زبان برنامه‌نویسی پایتون^۱ [۳۱] نوشته شده است. به کمک این کتابخانه می‌توان ناشر، مشترک و واسطه‌ایی به زبان پایتون نوشت.

با استفاده از فلوچارت، الگوریتم و ساختار مورد استفاده را نشان می‌دهیم. همانطور که قبله گفتیم، هر نواد

^۱Python

در خودش تنظیماتی دارد. اضافه و یا حذف شدن یک نوD تحت عناوین «سلام»^۱ و «خداحافظ»^۲ به اطلاعات اجداد آن نوD خواهد رسید. در تصویر ۱۵-۳ میتوان فلوچارت اضافه و یا حذف یک نوD از سامانه را مشاهده کنید.

داده‌های دریافتی از حسگرها توسط دستگاه به نوD دیگری ارسال می‌شوند تا در نهایت به سرور برسند. همچنین دستورات کاربر از سمت سرور باید به دستگاه رسیده و انجام گیرند. فلوچارت این دو به ترتیب در شکل ۱۶-۳ و شکل ۱۷-۳ قابل مشاهده است.

در انتها نمونه ساختار کلی پیاده‌سازی شده در شکل ۱۸-۳ قابل مشاهده خواهد بود. این ساختار شامل ارتباطات ماشین به ماشین است و شبیه‌سازی مناسبی از دنیای واقعی اینترنت اشیا را نشان می‌دهد. کدهای کل این پروژه به شکل متن باز در اختیار عموم قرار دارد.[۳۲]

¹Hi
²Bye

```

` void loop()
` {
`   WiFiEspClient client = server.available(); // listen for incoming clients
` 
`   if (client) { // if you get a client,
`     Serial.println("New client"); // print a message out the serial port
`     buf.init(); // initialize the circular buffer
`     while (client.connected()) { // loop while the client's connected
`       if (client.available()) { // if there's bytes to read from the client,
`         char c = client.read(); // read a byte, then
`         buf.push(c); // push it to the ring buffer
` 
`         // you got two newline characters in a row
`         // that's the end of the HTTP request, so send a response
`         if (buf.endsWith("\r\n\r\n")) {
`           sendHttpResponse(client);
`           break;
`         }
`       }
`     }
` 
`     // give the web browser time to receive the data
`     delay(10);
` 
`     // close the connection
`     client.stop();
`     Serial.println("Client disconnected");
`   }
` }

` void sendHttpResponse(WiFiEspClient client)
` {
`   client.print(
`     "HTTP/1.1 200 OK\r\n"
`     "Content-Type: text/html\r\n"
`     "Connection: close\r\n" // close connection after completion of response
`     "Refresh: 60\r\n" // refresh the page automatically every 60 sec
`     "\r\n");
`   client.print("<!DOCTYPE HTML>\r\n");
`   client.print("<html>\r\n");
`   client.print("<h1>Hello World!</h1>\r\n");
`   client.print("Requests received: ");
`   client.print(++reqCount);
`   client.print("<br>\r\n");
`   client.print("Analog input A0: ");
`   client.print(analogRead(0));
`   client.print("<br>\r\n");
`   client.print("</html>\r\n");
` }
` }
```

شکل ۳-۱۰: دریافت درخواست و ارسال پاسخ از طریق ماثول ESP8266

```

`#include <WiFiEsp.h>
`#include <WiFiEspClient.h>
`#include <WiFiEspUdp.h>
`#include "SoftwareSerial.h"
`#include <PubSubClient.h>
` 

`IPAddress server(192, 168, 1, 1); // IP Address of MQTT Broker
`char ssid[] = "SSID"; // your network SSID (name)
`char pass[] = "SECRET"; // your network password
`int status = WL_IDLE_STATUS; // the Wifi radio's status
` 

`// Initialize the Ethernet client object
`WiFiEspClient espClient;
` 

`PubSubClient client(espClient);
` 

`SoftwareSerial soft(8, 9); // RX, TX
`void setup() {
`    // initialize serial for debugging
`    Serial.begin(9600);
`    // initialize serial for ESP module
`    soft.begin(9600);
`    // initialize ESP module
`    WiFi.init(&soft);
` 

`    // check for the presence of the shield
`    if (WiFi.status() == WL_NO_SHIELD) {
`        Serial.println("WiFi shield not presented!");
`        // don't continue
`        while (true);
`    }
` 

`    // attempt to connect to WiFi network
`    while (status != WL_CONNECTED) {
`        Serial.print("Attempting to connect to WPA SSID: ");
`        Serial.println(ssid);
`        // Connect to WPA/WPA2 network
`        status = WiFi.begin(ssid, pass);
`    }
` 

`    // you're connected now, so print out the data
`    Serial.println("You're connected to the network");
` 

`    //connect to MQTT server
`    client.setServer(server, 1883);
`    client.setCallback(callback);
`}
` 
```

شکل ۳-۱۱: تنظیمات اولیه برای اتصال به واسطه پروتکل MQTT به کمک کتابخانه PubSubClient

```

1 //print any message received for subscribed topic
2 void callback(char* topic, byte* payload, unsigned int length) {
3   Serial.print("Message arrived [");
4   Serial.print(topic);
5   Serial.print("] ");
6   for (int i=0;i<length;i++) {
7     Serial.print((char)payload[i]);
8   }
9   Serial.println();
10 }
11 }

12 void loop() {
13   // put your main code here, to run repeatedly:
14   if (!client.connected()) {
15     reconnect();
16   }
17   client.loop();
18 }

19 void reconnect() {
20   // Loop until we're reconnected
21   while (!client.connected()) {
22     Serial.print("Attempting MQTT connection...");
23     // Attempt to connect, just a name to identify the client
24     if (client.connect("ArduinoClient")) {
25       Serial.println("connected");
26       // Once connected, publish an announcement...
27       client.publish("FromArduino", "HelloWorld!");
28       // ... and resubscribe
29       client.subscribe("ToArduino");
30     } else {
31       Serial.print("failed, rc=");
32       Serial.print(client.state());
33       Serial.println(" try again in 3 seconds");
34       // Wait 3 seconds before retrying
35       delay(3000);
36     }
37   }
38 }

39 }
40 }
```

شکل ۱۲-۳: اتصال به واسطه، ارسال پیام در موضوع FromArduino و گرفتن اشتراک در موضوع ToArduino

```

1 #include <LiquidCrystal.h>
2
3 // indicate which pin of LCD is connected to which pin of Arduino
4 const int rs = 12, en = 11, d4 = 5, d5 = 4, d6 = 3, d7 = 2;
5 LiquidCrystal lcd(rs, en, d4, d5, d6, d7);
6
7 void setup() {
8   // configure the object by passing number of columns and rows
9   lcd.begin(16, 2);
10  // print a message
11  lcd.print("Hello World!");
12 }
13
14 void loop() {
15 }
```

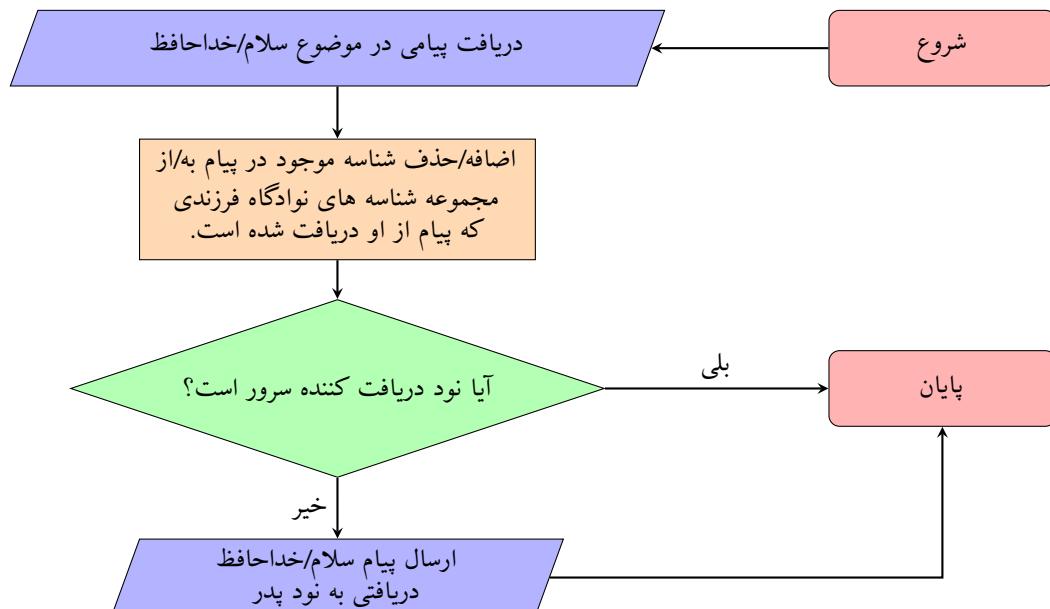
شکل ۱۳-۳: نمایش پیام Hello World! در نمایشگر گر با استفاده از کتابخانه LiquidCrystal

```

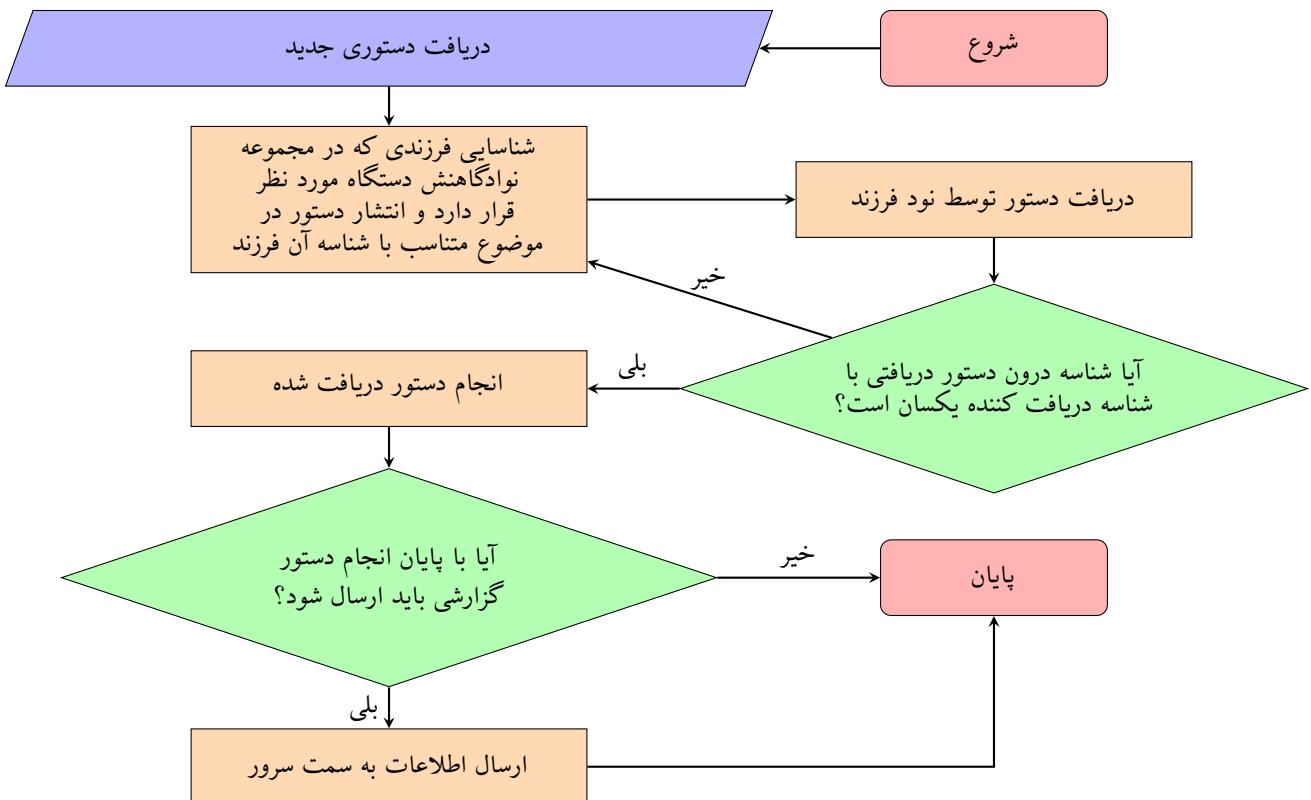
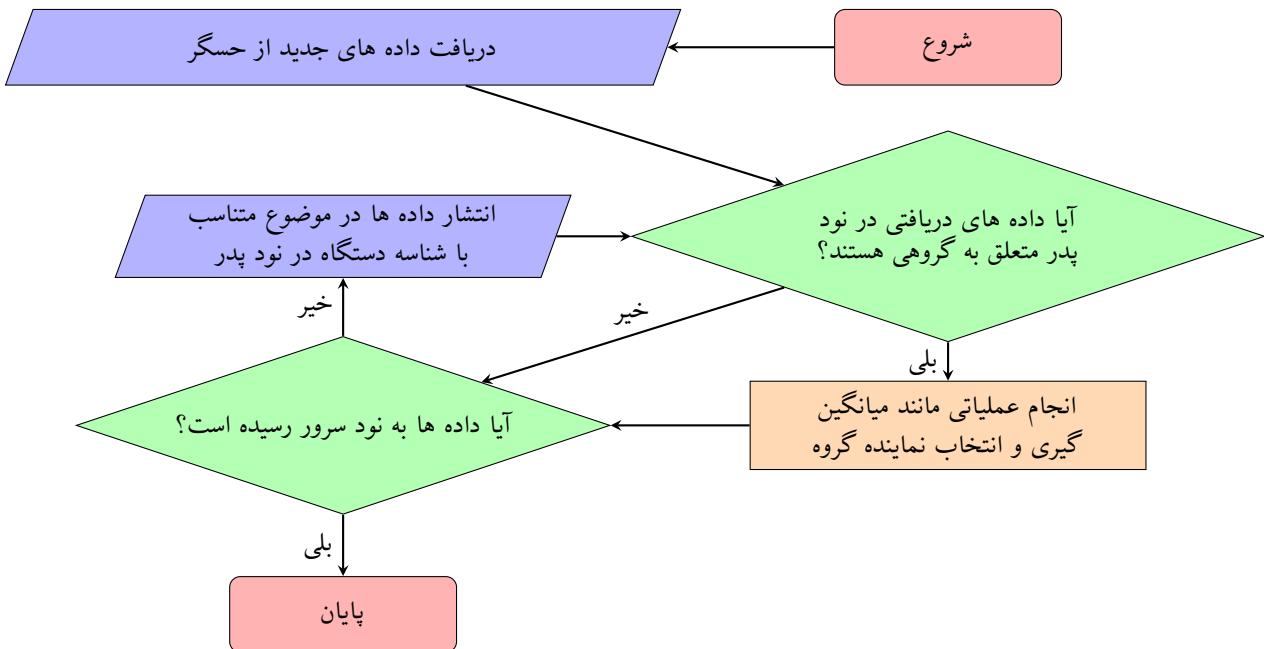
1 #include <SPI.h>
2 #include <MFRC522.h>
3
4 constexpr uint8_t RST_PIN = 9;
5 constexpr uint8_t SS_PIN = 10;
6
7 MFRC522 mfrc522(SS_PIN, RST_PIN); // Create MFRC522 instance
8
9 void setup() {
10   Serial.begin(9600); // Initialize serial communications with the PC
11   while (!Serial); // Wait until serial communications start
12   SPI.begin(); // Init SPI bus
13   mfrc522.PCD_Init(); // Init MFRC522
14   mfrc522.PCD_DumpVersionToSerial(); // Show details of MFRC522 Card Reader
15   Serial.println(F("Scan PICC to see UID, SAK, type, and data blocks..."));
16 }
17
18 void loop() {
19   // Look for new cards
20   if ( ! mfrc522.PICC_IsNewCardPresent() ) {
21     return;
22   }
23
24   // Select one of the cards
25   if ( ! mfrc522.PICC_ReadCardSerial() ) {
26     return;
27   }
28
29   // Dump debug info about the card
30   mfrc522.PICC_DumpToSerial(&(mfrc522.uid));
31 }

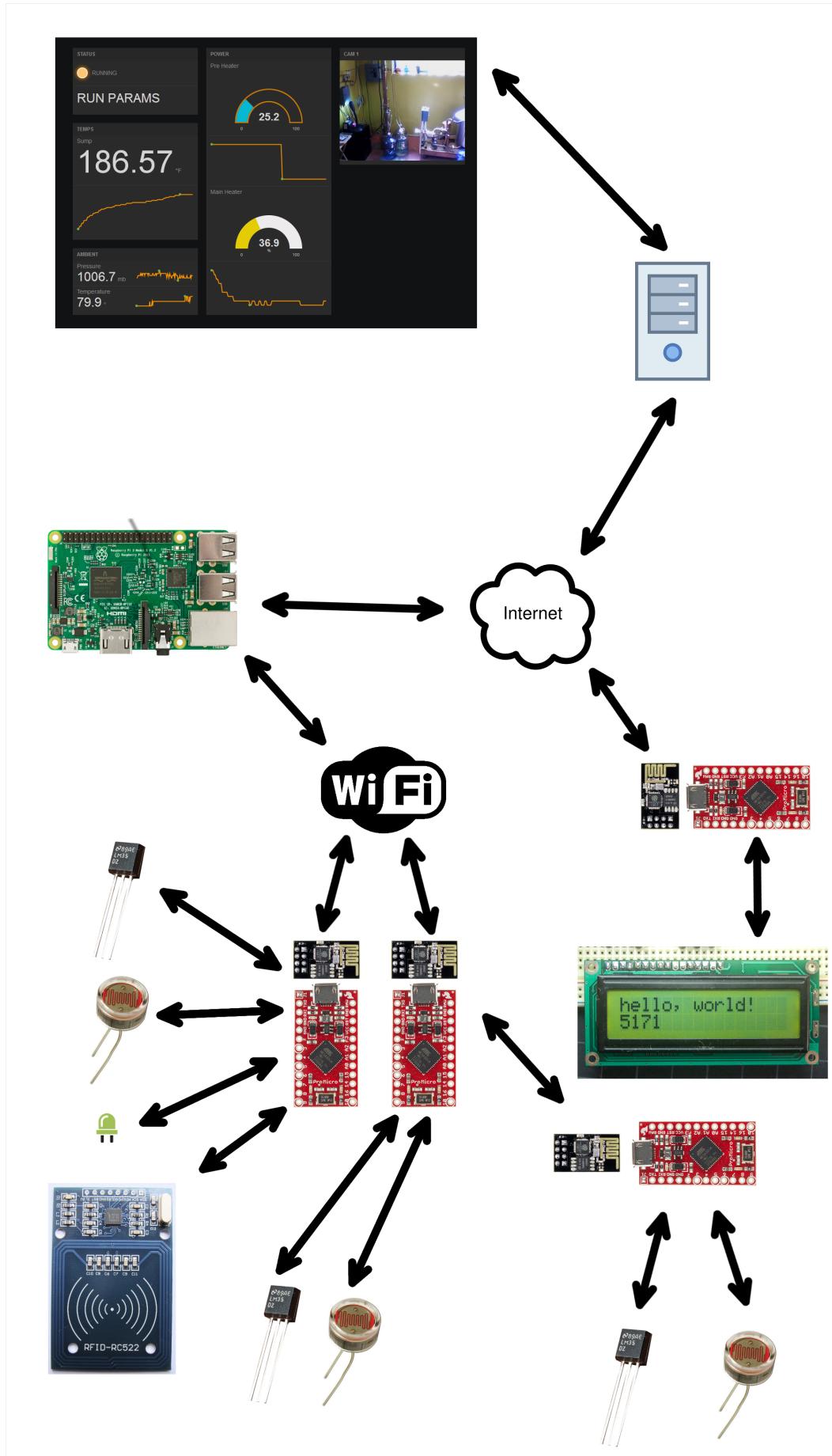
```

شکل ۱۴-۳: خواندن اطلاعات کارت‌های RFID به کمک کتابخانه MFRC522



شکل ۱۵-۳: فلوچارت اضافه شدن و یا حذف یک نود از سامانه





شکل ۳-۱۸: نمونه پیاده‌سازی شده توسط ساختار معرفی شده.

مراجع

- [1] Wikipedia: History of Internet https://en.wikipedia.org/wiki/History_of_the_Internet
- [2] Wikipedia: OSI Model https://en.wikipedia.org/wiki/OSI_model
- [3] Wikipedia: Kevin Ashton https://en.wikipedia.org/wiki/Kevin_Ashton
- [4] Wikipedia: IEEE 802.11 https://en.wikipedia.org/wiki/IEEE_802.11
- [5] IBM OSI Model Picture https://www.ibm.com/support/knowledgecenter/en/SSCVHB_1.1.0/glossary/npi_osi_model.html
- [6] IGLOO Nano FPGAs <https://www.microsemi.com/products/fpga-soc/fpga/igloo-nano>
- [7] Raspberry Pi 3 https://en.wikipedia.org/wiki/Raspberry_Pi
- [8] Top 10 internet of things stories of 2015 <http://www.computerweekly.com/news/4500260406/Top-10-internet-of-things-stories-of-2015>
- [9] Range Limitation Introducing Mesh Networks <https://blog.bluetooth.com/range-limitation-what-range-limitation-introducing-mesh-networks>
- [10] E. Dave, “White paper: the Internet of Things, how the next evolution of the Internet is changing everything” Cisco Internet Business Solutions Group (IBSG).

- [11] K. Zheng, Z. Yang, K. Zhang, P. Chatzimisios, W. Xiang and K. Yang, “*Big data driven optimization for mobile networks towards 5G*,” IEEE Netw., vol. 30, no. 1, pp. 44-51, Jan. 2016.
- [12] Y. Zhang, S. He and J. Chen, “*Data gathering optimization by dynamic sensing and routing in rechargeable sensor networks*,” IEEE/ACM Trans. Netw., vol. PP, no. 99, pp. 1-15, Jun. 2015.
- [13] N. Kumar, S. Zeadally, N. Chilamkurti and A. Vinel, “*Performance analysis of Bayesian coalition game-based energy-aware virtual machine migration in vehicular mobile cloud*, ” IEEE Netw., vol. 29, no. 2, pp. 62-69, Apr. 2015.
- [14] C Gomez, J Oller, J Paradells, “*Overview and evaluation of bluetooth low energy: An emerging low-power wireless technology*, ” Sensors, 2012 - mdpi.com
- [15] Geoff Mulligan, “*The 6LoWPAN Architecture*”
- [16] SPDY: An experimental protocol for a faster web <https://dev.chromium.org/spdy/spdy-whitepaper>
- [17] Vasileios Karagiannis, Periklis Chatzimisios, Francisco Vazquez-Gallego, Jesus Alonso-Zarate, “*A Survey on Application Layer Protocols for the Internet of Things*, ” ICSA Publishing, Transaction on IoT and Cloud Computing, 2015
- [18] Vangelis Gazis, Manuel Götz, Marco Huber, Alessandro Leonardi, Kostas Mathioudakis, Alexander Wiesmaier, Florian Zeiger, Emmanouil Vasilomanolakis, “*A Survey of Technologies for the Internet of Things*, ” International Wireless Communications and Mobile Computing Conference, 2015
- [19] Message Queuing Telemetry Transport (MQTT) OASIS Standard by Andrew Banks and Rahul Gupta, September 2014. <http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/mqtt-v3.1.1.html>
- [20] Internet Engineering Task Force (IETF), Request for Comments: 7252, The Constrained Application Protocol (CoAP) <https://tools.ietf.org/html/rfc7252>
- [21] Kiran Jot Singh, Divneet Singh Kapoor, “*A survey of IoT platforms: Create Your Own Internet of Things*, ” IEEE Consumer Electronics Magazine, April 2017
- [22] Prarna Dhar, Prof.Poonam Gupta “*Intelligent Parking Cloud Services based on IoT using MQTT Protocol*, ” International Journal of Engineering Research Volume No.5, Issue No.6, pp : 457-461, ISSN:2319-6890)(online),2347-5013(print) 1 June 2016

- [23] Georgios Pierris, Dimosthenis Kothris, Evangelos Spyrou, Costas Spyropoulos, “SYNAISTHISI: An Enabling Platform for the Current Internet of Things Ecosystem, ”, ACM, ISBN 978-1-4503-3551-5/15/10, 2015
- [24] Seong-Min Kim, Hoan-Suk Choi, Woo-Seop Rhee ”IoT home gateway for auto-configuration and management of MQTT devices, ”, IEEE Conference on Wireless Sensors, 2015
- [25] Livelessons Course: “Internet of Things Fundamentals” by Jerome Henry, Rob Barton
- [26] K Finkenzeller, “RFID handbook: fundamentals and applications in contactless smart cards, radio frequency identification and near-field communication, ” 2010
- [27] Open source Arduino ESP8266 Library. <https://github.com/bportaluri/WiFiEsp/>
- [28] Arduino Client for MQTT by Nick O’Leary <https://pubsubclient.knolleary.net/>
- [29] Mosquitto, An Open Source MQTT v3.1/v3.1.1 Broker <https://mosquitto.org/>
- [30] MQTT client/broker using Python asynchronous I/O <https://github.com/beerfactory/hbmqtt>
- [31] Python is a programming language that lets you work more quickly and integrate your systems more effectively. <https://www.python.org/>
- [32] Implementation of IoT platform with MQTT protocol. This was my Bachelor project. <https://github.com/733amir/Bachelor-Project>