



درس آزمایشگاه سخت افزار

دکتر اجلالی

گزارش پروژه

## سیستم جامع مدیریت اینترنت اشیا

اعضا:

محمد معین صمدی آزاد

محمد شایان شعبانی

امیرحسین عزیزی

## تعریف پروژه:

در این پروژه، هدف ارائه یک پنل برای مدیریت و کنترل دستگاه های مختلف به کاربر است. این پنل اطلاعات مربوط به حسگر های اشیا مختلف را به کاربر ارائه می دهد. همچنین قابلیت کنترل عملگر های مختلف در اشیا را از راه دور به کاربر می دهد.

با توجه به نیاز های موجود، به چهار بخش کلی برای اجرای پروژه نیاز است. اولین بخش شامل آماده سازی اشیایی که می خواهیم به در سیستم مدیریت کنیم است. بخش بعدی ارتباط بین این دستگاه ها و سرور اصلی است. بخش سوم، ارائه اطلاعات و کنترل ها به کاربر و بخش چهارم ارائه رابط کاربری مناسب به کاربر است.

## راه کار های در نظر گرفته شده:

با توجه به تعریف پروژه، از برد های ESP32 به عنوان نمونه کامپیوتر هایی که در اشیا قرار گرفتند استفاده کردیم. این برد ها به چندین حسگر و عملگر مختلف به عنوان نمونه متصل شدند. برای بخش دوم، از پروتکل MQTT و بروکر Mosquito به دلیل سادگی و سبکی در عین برآورده کردن نیاز های پروژه، برای ارتباط برد ها استفاده شد. برای بک اند تصمیم به استفاده از جنگو گرفتیم. دلیل این امر سادگی جنگو، آشنایی بیشتر با آن و همچنین وجود کتابخانه های پایتونی برای ارتباط با MQTT است. در انتها برای رابط کاربری از Vue.js به دلیل زیبایی استفاده کردیم.

## مشکلات و چالش های مرتبط با راه کار ها:

بخش دوم، پروتکل MQTT:

در این قسمت، مشکلات جزئی وجود داشتند. از جمله آن ها میتوان به عدم پشتیبانی از دریافت لیست دستگاه های مرتبط که به حفظ وضعیت برد های مختلف کمک میکرد و همچنین وجود یک حساب مشترک و برابر برای تمام دستگاه ها اشاره کرد.

بخش سوم و چهارم:

ابتدا در این قسمت تصمیم به استفاده از streamlit گرفته بودیم. نقطه قوت این کتابخانه، سادگی بسیار زیاد توسعه و یک پارچه شدن بک اند و فرانت اند بود. البته که این سادگی، با محدودیت نیز همراه بود، به طوری که نیاز نمایش نمودار اطلاعات حسگر به طور زنده روی آن ساده نبود، و همچنین این سادگی به قیمت عدم قابلیت شخصی سازی و ایجاد تغییرات ساختاری در ظاهر تمام می شد. به همین دلیل تصمیم به استفاده از vue.js گرفته شد.

البته در طی اجرای پروژه، مشکلاتی برای استفاده از Vue.js پیش آمد، و باعث شد برای مدتی تصمیم به استفاده از Django template بگیریم که خوشبختانه در نهایت مشکلات حل شدند و توانستیم از Vue استفاده کنیم که قابلیت توسعه بیشتری دارد.

همچنین در ابتدا تصمیم به استفاده از redis برای نگه داری اطلاعات گرفتیم، که بعدا آن را با دیتابیس sqllight جایگزین کردیم تا دیتا را امن تر و طولانی تر نگه داریم.

## نیازمندی های پروژه:

برای اجرای پروژه، به 6 برد ESP32 ، حداقل یک سرور برای پایگاه داده، بروکر MQTT و Django نیاز داریم، همچنین 3 عدد سنسور میکروفون، نور و ultrasonic و 3 عملگر LED ، Buzzer و Servomotor استفاده شد.

## اجرا:

اجرای پروژه در چهار مرحله انجام شد.

مرحله اول، راه اندازی برد ها و سنسور ها:

با توجه به اینکه ما 3 سنسور و 3 عملگر دریافت کردیم که هر یک متعلق به یکی از 6 برد ESP 32 ما می باشد، هر یک از این قطعات را به برد مربوطه وصل کردیم و از صحت عملکرد آن مطمئن شدیم. در ادامه، کد مناسب برای کار با حسگر یا عملگر را به صورت محلی نوشتیم، به این شکل که حسگر ها اطلاعات را به روی سریال نمایش می دادند و عملگر ها هر چند ثانیه عمل خود را انجام می دادند.



توسعه کد برد ها - سنسور ultrasonic

## مرحله دوم، راه اندازی بروکر، اتصال برد ها به بروکر و شروع توسعه بک اند:

در این مرحله، توانستیم سیستم را روی اینترنت ببریم. برای اینکار، ابتدا با داندلود بروکر mosquito و نصب و کانفیگ آن، با یک یوزرنیم و پسورد توانستیم به آن کانکت بشویم و پیام ها را انتقال دهیم، بعد از نصب آن مطابق سایت رسمی، تمامی دستگاه ها را به یک شبکه وای فای متصل کردیم. همچنین کد برد ها را تغییر دادیم تا به WIFI و سپس بروکر متصل شوند. علاوه بر این عملکرد آن ها را به روی پروتکل MQTT آوردیم به طوری که برای سنسور در کد آن ها در لوپ و برای عملگر ها کد آن در کال بک قرار گرفت.

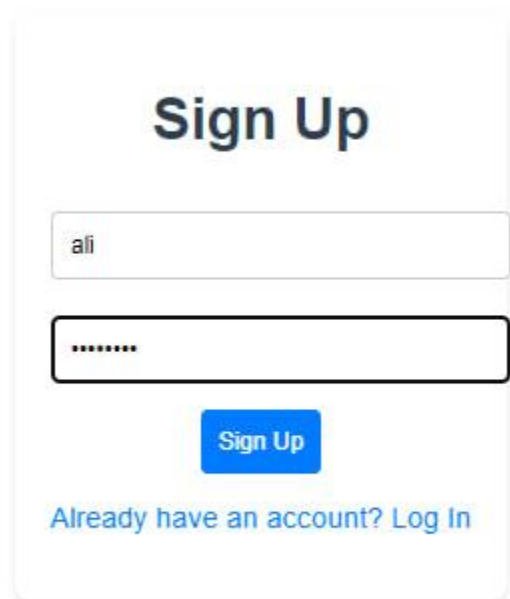
در ادامه تابع هایی با جنگو در سرور نوشتیم و آن را هم به بروکر متصل کردیم، این تابع ها وظیفه کنترل هر کدام از عملگر ها ، و همچنین دریافت اطلاعات هر کدام از سنسور ها را دارند .ما یک پروژه به وسیله جنگو بالا آوردیم که همزمان دو کار را میتواند انجام دهد. از یک سو می تواند به وسیله URL هایی که برای آن تعریف شده است، سیگنال های on یا off را برای Buzzer و LED و همچنین سیگنال position را برای motor servo از کاربر پشت سیستم دریافت کند و برای ESP32 بفرستد. از سوی دیگر، این سرور ما همواره روی تاپیک های Light ,Sound Ultrasound, دارد به ESP 32 گوش می کند و منتظر است تا اطلاعاتی را دریافت کند و سپس به کاربر نشان دهد.

## مرحله سوم، توسعه بک اند:

بعد از راه اندازی سنسور ها و عملگر ها، راه اندازی بروکر و سرور، و در نهایت اتصال این اجزا به یک دیگر، در این مرحله، به توسعه بک اند جنگو پرداختیم. به طوری که یک کلاس دستگاه ایجاد کردیم، که انواع دستگاه های مختلف از آن ارث خواهند برد. این کلاس ویژگی های مشترک دستگاه ها مثل نام، تاپیک، توکن و کاربر را در بر خواهد داشت. سپس سه نوع دستگاه از آن مشتق کردیم. یک کلاس برای سنسور ها که اطلاعات رسیده را با زمان ذخیره می کند، یک کلاس برای عملگر های بولین که دو استیت 0 و 1 دارند، می توان آن ها را در یکی از این دو قرار داد، و یک کلاس برای عملگر های عددی که با دریافت یک عدد به اندازه آن کاری انجام می دهند. همچنین در این قسمت کلاینت MQTT هم تغییر کرد و در یک ترد مجزا اجرا می شود.

## مرحله چهارم، توسعه فرانت اند، اتصال به بک اند و بهبود ساختار کلی:

در فرانت ابتدا با صفحه لاگین مواجه می شویم که در یک component در پروژه ما قرار داده شده است. این قسمت همچنین با نگه داشتن یک Boolean، می توانیم بین login و signup تغییر وضعیت دهیم. در صفحه اصلی نیز این قابلیت با قرار دادن یک لینک در پایین دکمه ارسال ایجاد شده که مود را از لاگین به ساین آپ تغییر می دهد.



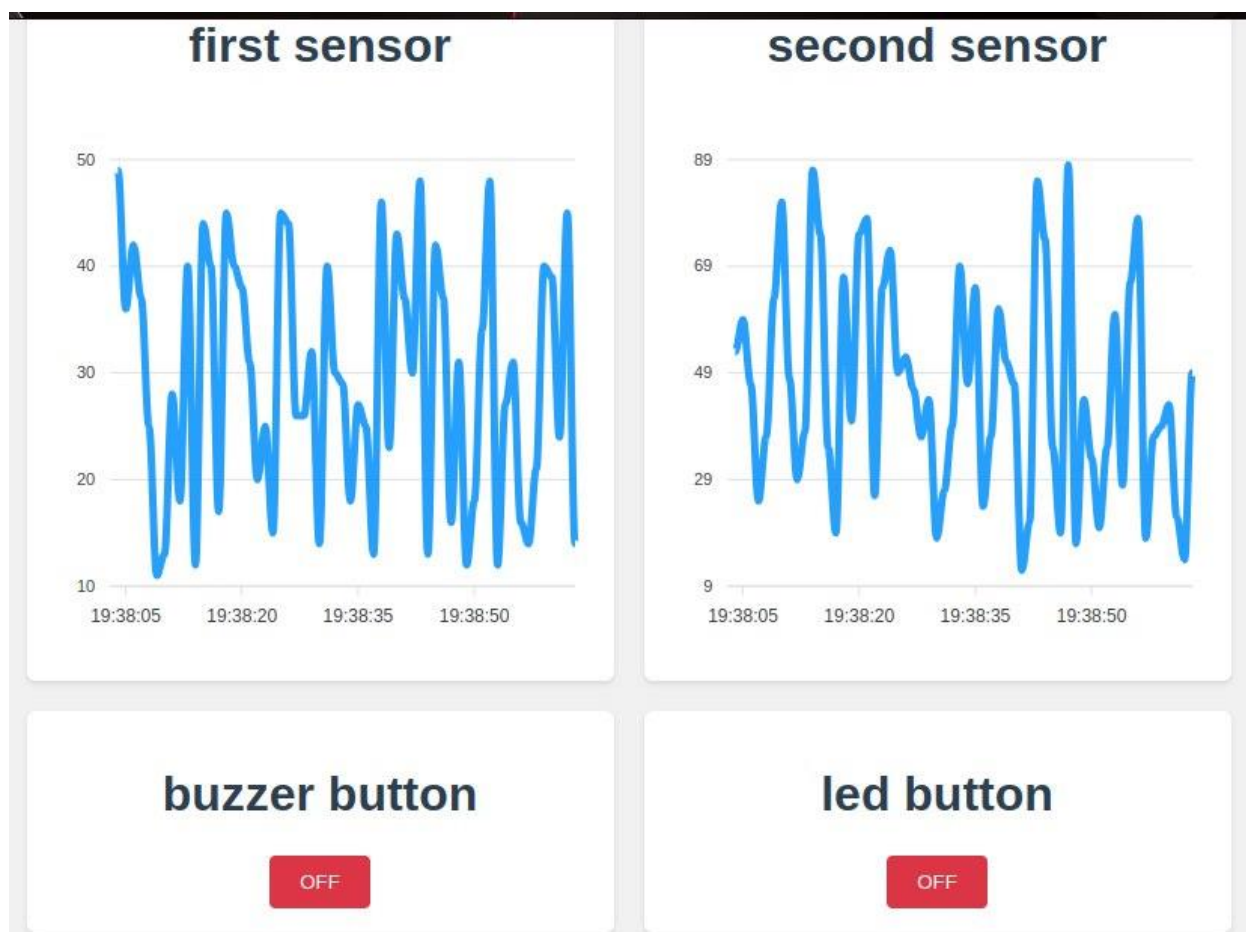
صفحه ساخت حساب

پس از ارسال نام کاربری و رمز عبور، در صورت موافقت سرور، به صفحه اصلی می رویم. در این صفحه، سه نوع کارت داریم.

نوع اول دکمه های Boolean می باشد، مانند دکمه خاموش و روشن کردن یک led، که با کلیک بر روی آن، به سرور میگوییم وضعیت آن دستگاه را تغییر دهد. وضعیت دستگاه در این حالت را نیز با یک Boolean نگه داری میکنیم و سپس نشان میدهیم.

نوع دوم برای servo motor است که صرفاً یک integer در فرم دریافت میکند و سپس به سرور ارسال میکند.

نوع سوم نیز نمودارهایی برای نشان دادن وضعیت سنسورهای ما می باشد. این نمودارها dynamic هستند و ماکسیمم آنها برابر با بزرگترین عدد دیده شده است. هر 5 ثانیه نیز یک ریکوئست به سرور زده می شود تا وضعیت جدید دریافت و نشان داده شود. در هر یک از این درخواستها به سرور نیز توکن یوزر را نیز میفرستیم تا مطمئن شویم این نتایج فقط در دسترس کاربرهای لاگین شده قرار بگیرد.



در ابتدای بالا آمدن صفحه اصلی، یک درخواست به سرور زده می شود و اطلاعات تمامی device های حاضر در سرور را دریافت می کنیم. در پاسخ این ریکوئست، علاوه بر اسم و توکن آن دستگاه، یک عدد به عنوان تایپ نیز دریافت می کنیم که میتواند 0 یا 1 یا 2 باشد. نوع 0 برای دکمه های بولین، نوع 1 برای سربو موتور و نوع 2 برای سنسور ها است. سپس با نگه داشتن آرایه ای از دستگاه ها، و با استفاده از دستور `v-if`، بر اساس تایپ های مختلف دستگاه ها را نشان داده و ریکوئست های مناسب را میزنیم.

نکته دیگر فرانت نیز استفاده از یک آرایه برای نگه داشتن `chartOption` های همه نمودار ها بود. برای اینکه بتوانیم `maximum value` هر نمودار را بر اساس داده های آن تعیین کنیم، باید برای هر نمودار از یک `chartOption` جداگانه استفاده کنیم. به همین دلیل به جای استفاده از یک `chartOption`، از یک آرایه از آن ها استفاده می کنیم که هر یک در زمان ایجاد آن نمودار، ایجاد میشود و مختص به آن نمودار می باشد. در این بخش، مطابق نیاز های ایجاد شده در فرانت اند، بک اند را تغییر دادیم، همچنین پیام ها را در قالب `json` بین بک اند و بورد ها انتقال داده و کد بورد ها را برای اتصال مجدد به وای فای و `MQTT` بروکر می دهیم.

## مرحله آخر، راه اندازی:

برای راه اندازی، از نرم افزار `Bore` که روی لینوکس، ویندوز و مک او اس در دسترس است استفاده کردیم، به این شکل که بعد از اجرای بروکر روی یک دستگاه، پورت آن را به یک پورت از دامنه `bore.pub` مپ کردیم، همچنین برای جنگو و فرانت نیز یک پورت دیگر از آن دامنه گرفته و با تنظیم اطلاعات وای فای و بروکر در سرور و بورد ها، در نهایت سیستم به روی اینترنت در دسترس قرار گرفت.



