

بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ



دانشگاه صنعتی اصفهان
دانشکده مهندسی برق و کامپیوتر

پیاده سازی اپلیکیشن یکپارچه در حوزه کنترل خانه های هوشمند

پایان نامه کارشناسی مهندسی کامپیوتر

حمیدرضا صادقی (۹۶۳۰۵۱۳)

استاد راهنما

دکتر مسعود رضا هاشمی

بهمن ۱۴۰۰

تشکر و قدردانی

سپاس خدای بزرگ را که مرا یاری رساند تا بتوانم این مقطع تحصیلی را به پایان رسانده و گامی در راستای اعتلای علم بردارم. از پدر و مادر مهربان تر از ”مهربان” که مرا در مسیر رشد و بالندگی راهنمایی کردند، تشکر میکنم. از استاد راهنمای گرانقدرم جناب **دکتر مسعود رضا هاشمی** که وجودشان همیشه قوتی برای انجام کارهایم بوده است و بدون شک انجام این پایانامه بدون کمک و راهنمایی‌های ارزنده آنها امکان پذیر نبوده است، کمال تشکر را دارم.

و در پایان از تمامی عزیزانی که در طول انجام این پروژه مرا یاری کرده‌اند کمال تشکر و قدردانی را ابراز می‌نمایم.

فهرست مطالب

عنوان	صفحه
فهرست مطالب	چهار
چکیده	۱
فصل اول: کلیات پژوهش	
۱-۱ مقدمه	۲
۲-۱ تعریف مساله و سوالات اصلی تحقیق	۴
۳-۱ نوآوری تحقیق و پژوهش	۴
۴-۱ ساختار پایان نامه	۴
فصل دوم: مفاهیم مورد نیاز	
۱-۲ کاربرد اینترنت اشیاء در خانه هوشمند	۵
۲-۲ ساختار اتوماسیون خانه هوشمند	۶
۳-۲ ساختار رایانش ابری	۷
۴-۲ ساختار محاسبات مه در بروز رسانی تجهیزات	۷
۵-۲ ساختارهای بروز در جهان واقعی	۸
۶-۲ سیستم های نرم افزاری تحت وب	۸
۱-۶-۲ وظایف برنامه سمت سرور	۹
۲-۶-۲ وظایف برنامه سمت مشتری	۹
۷-۲ معماری REST	۱۰
۸-۲ جنگو	۱۲
۹-۲ جنگو-رست	۱۳
فصل سوم: معماری	
۱-۳ طرح کلی مساله	۱۵
۲-۳ نقشه کلی وب سایت	۱۶
۱-۲-۳ Cloud	۱۶
۲-۲-۳ Home	۱۷
۳-۲-۳ Fog Node	۱۸

۱۹	۳-۳ معماری کلی پایگاه داده
----	----------------------------

فصل چهارم: پیاده سازی

۲۲	۱-۴ بررسی برخی API ها
۲۲	۱-۱-۴ احراز هویت و امور حساب ها
۲۴	۲-۱-۴ کنترل دستگاه ها
۲۵	۳-۱-۴ پنل ادمین
۲۵	۴-۱-۴ کنترل خانه
۲۷	۲-۴ اجرای سناریو معماری بروزرسانی شده

فصل پنجم: نتیجه گیری و آینده نرم افزار

۳۲	۱-۵ نتیجه گیری
۳۲	۲-۵ آینده نرم افزار

فصل ششم: مراجع و پیوست

۳۴	۱-۶ پیوست
۳۵	مراجع

چکیده

با افزایش روزافزون نیاز انسان‌ها به دستگاه‌های هوشمند در زندگی روزمره و وجود ترافیک سنگین داده‌ها در مسیر ارتباط بین کاربران و تجهیزات خانه‌هوشمند، نیاز به رایانش ابری در این محیط برای کاربران احساس خواهد شد. اما با این حال رایانش ابری ممکن است به تنهایی توانایی پردازش درخواست‌ها را، در زمان افزایش ترافیک داده‌ها نداشته باشد و نتواند سطح استاندارد از امنیت، ارتباط سریع و قابلیت در دسترس بودن را تضمین نماید. در این زمان قرار گرفتن محاسبات مه در کنار رایانش ابری خواهد توانست خدمات آن را تا انتهای شبکه گسترش دهد و خدمات محاسباتی را برای تجهیزات موجود در شبکه خانه‌هوشمند سرعت ببخشد. با این اقدام امنیت و کارایی سیستم بهبود خواهد یافت و میزان انتقال داده‌ها در سراسر شبکه به ابر برای پردازش و ذخیره سازی کاهش می‌یابد.

حال با افزایش دستگاه‌ها و سخت شدن کنترل و مدیریت آن‌ها، وجود یک سیستم کنترلی هوشمند و یکپارچه که بتواند علاوه بر کنترل و مدیریت دستگاه‌های خانه‌های هوشمند وظایفی از قبیل کنترل و انتقال داده‌ها، برقرای امنیت خانه، بروزرسانی و همگام سازی با تکنولوژی‌های روز دنیا و حتی انجام وظایفی که از قبل برنامه ریزی یا پیشبینی شده است، کاملاً احساس می‌شود.

در این مقاله، ما با توجه به سیستم پیشنهادی، سعی داریم یک سیستم منسجم و بروز را پیاده سازی کنیم تا نحوه ارتباطات و کارکرد سیستم به نمایش گذاشته شود و در آینده نزدیک وارد چرخه تولید شود.

کلمات کلیدی: خانه هوشمند، اتوماسیون خانه، محاسبات مه، پردازش ابری، پیاده سازی اپلیکیشن هوشمند، بروز رسانی سیستم عامل

فصل اول

کلیات پژوهش

۱-۱ مقدمه

مفهوم خانه هوشمند^۱، مدتهاست که در ادبیات تکنولوژی جهان مطرح شده است و امروزه به سرعت در حال رشد است. خانه هوشمند یک نمونه کوچک شده از فناوری جهانی اینترنت اشیا^۲ محسوب می شود که با استفاده از آن قادر خواهیم بود، تمامی ابزار الکترونیکی موجود در منزل را از راه دور کنترل کرده و هر لحظه از اتفاقاتی که درون منزل ما می گذرد، آگاهی داشته باشیم. خانه هوشمند مجموعه ای از تکنولوژی ها و سرویس ها در شبکه خانگی برای بهبود کیفیت زندگی انسان ها می باشد. این شبکه هوشمند شامل امکانات ارتباطی، سرگرمی، امنیتی، آسایشی و همچنین سرویس هایی برای افراد معلول و سالمند می باشد. خانه های هوشمند از توانایی آشکاري در راحت تر کردن زندگی و افزایش رفاه برای کاربران برخوردار می باشند و آسودگی خاطر را برای ساکنان آن میسر می نمایند. خانه هوشمند شمارا چه در محل کار باشید چه در تعطیلات، از آنچه در محیط خانه می گذرد مطلع می سازد. این سیستم از لحاظ امنیتی^۳ در مواقع اضطراری کمک بزرگی برای شما خواهد بود. برای مثال در موقع آتش سوزي علاوه بر این که ساکنان را با زنگ خطر مطلع می کند، کارهای لازم همچون باز کردن همه درها،

¹Smart Home

²Internet of Things

³Security

تماس با آتش نشانی، روشن کردن لامپ‌های راهروها برای ایمنی بیشتر و اقدامات مورد نیاز دیگر در آن زمان را انجام می‌دهد. سیستم خانه‌هوشمند همچنان دارای مشکلات بسیاری می‌باشد که روز به روز با پیشرفت تکنولوژی در حال برطرف شدن و بروز شدن است، اما با این حال هنوز مسائل باز بسیاری در زمینه خانه‌هوشمند وجود دارد. خانه‌هوشمند باعث صرفه‌جویی در مصرف انرژی می‌شود، وقتی ساکنان اتاق را ترک می‌کنند روشنایی اتاق‌ها به طور اتوماتیک خاموش می‌شود و دمای اتاق‌ها براساس اینکه کسی در محل هست یا نه تنظیم می‌گردد.

در اینجا ما سعی داریم یک اپلیکیشن^۱ پیاده سازی کنیم و در آن بتوانیم دستگاه های خانه های هوشمند خود را کنترل کنیم، از آنها گزارش بگیریم و در صورت نیاز نرم افزار چارچوب^۲ آنرا به روز رسانی کنیم. شما فرض کنید که در حال استفاده از تجهیزات یک خانه‌هوشمند هستید و بعد از مدتی این تجهیزات دچار ایرادات امنیتی یا نرم افزاری می‌گردد، حال در حالت معمول شما مجبور خواهید بود در بهترین حالت به صورت دستی با بروزرسانی نرم‌افزاری، تجهیزات خود را از نظر امنیتی ارتقا دهید که برای این کار نیاز به حضور یک متخصص و انجام بروزرسانی به صورت دستی خواهد بود و یا در بدترین حالت مجبور به تعویض و خرید تجهیزات جدید با توجه به نیازهای جدید بوجود آمده برای خود می‌باشید که تمامی این کارها شامل هزینه‌های بالا برای کاربران می‌باشد.

در این سیستم ما قصد داریم با توجه بازخورد کاربران و همچنین توجه به این امر که برای پویایی یک نرم افزار لازم به این است که هر چند وقت یکبار آن نرم افزار بروز رسانی شود، یک سیستمی پیاده سازی کنیم که بتواند با توجه به دستوراتی که از سرورهای بالادست خود میگیرد، نسخه های جدیدتر خود را از سرورهای شرکت دانلود کرده و خود را همزمان بروز رسانی کند.

با توجه به توضیحات ارائه شده در این بخش و نیاز به امکان بروزرسانی سیستم‌های خانه‌هوشمند^۳، ما با استفاده از مباحثی از قبیل رایانش ابری^۴، محاسبات مه^۵ و اتوماسیون خانه‌هوشمند قصد در ارائه یک نرم افزار^۶ کاربردی از سیستم خانه‌هوشمند برای بروزرسانی کلیه تجهیزات موجود در این شبکه را خواهیم داشت که علاوه بر بهبود امنیتی و کارایی، امکان تغییرات کلی در سیستم و همچنین کاهش زمان پاسخ گویی وجود داشته‌باشد.

¹ Application

² Application Firmware

³ Smart Home Automation

⁴ Cloud Computing

⁵ Fog Computing

⁶ Software

۲-۱ تعریف مساله و سوالات اصلی تحقیق

در حال حاضر بسیاری از کاربرانی که در حال استفاده از تجهیزات خانه هوشمند می باشند، با مشکلاتی در خصوص امنیت، کارایی و سلیقه ای نبودن تجهیزات خانه هوشمند روبرو می باشند که در حال حاضر با وجود تجهیزات موجود این نیازها به صورت جامع قابل برطرف نمودن نمی باشد. ما سعی داریم با ارائه یک اپلیکیشن جامع دسترسی راحت و فراگیر به دستگاه های هوشمند را فراهم کنیم و با قابلیت بروزرسانی که برای این پروژه پیاده سازی میشود، عمر سیستم افزایش یابد.

۳-۱ نوآوری تحقیق و پژوهش

نوآوری در این پژوهش در آنجایی است که این سیستم به عنوان یک سیستم مادر میتواند بر تعداد زیادی خانه هوشمند نظارت کند و امر امنیت و کارایی را بسیار راحت کند، همچنین در صورت جابجایی منزل، تمام اطلاعات افراد به راحتی به خانه جدید قابل انتقال می باشد.

۴-۱ ساختار پایان نامه

گزارش معرفی و پیاده سازی پژوهش در پنج فصل خلاصه می شود.
در فصل دوم به معرفی مفاهیم اولیه مورد نیاز برای انجام پژوهش می پردازیم.
در فصل سوم به معماری سیستم و نحوه پیاده سازی پایگاه های داده^۱ و ارتباط آنها و برخی از رابط های برنامه نویسی کاربردی^۲ می پردازیم.
در فصل چهارم به پیاده سازی چند سناریو اجرای سیستم می پردازیم.
در فصل پنجم نتیجه گیری از این پژوهش و راه کارهای بهبود آن و چشم انداز آینده آن را ارائه خواهیم کرد.

^۱Database

^۲API : Application Programming Interface

فصل دوم

مفاهیم مورد نیاز

در این فصل به طور مختصر به بررسی مفاهیم اولیه و مورد نیاز برای درک بهتر این سیستم می‌پردازیم. در ابتدا به مفاهیم اولیه اینترنت اشیاء و کاربرد آن در خانه‌هوشمند می‌پردازیم، در ادامه ساختار اتوماسیون خانه‌هوشمند، رایانش ابری و محاسبات مه را مورد بررسی قرار خواهیم داد و به طور مختصر به بیان آن‌ها می‌پردازیم. سپس به بررسی مفاهیم سیستم‌های نرم‌افزاری تحت وب و معماری REST^۱ می‌پردازیم. و در آخر درمورد چهارچوب نرم‌افزاری Django و Django-rest-framework خواهیم پرداخت.

۲-۱ کاربرد اینترنت اشیاء^۲ در خانه هوشمند

اینترنت اشیا به دستگاه‌هایی گفته می‌شود که به اینترنت متصل می‌شوند و اطلاعات را از طریق آن منتقل می‌کنند و می‌توانند از طریق آن دستورات خود را بگیرند. اینترنت اشیا کاربران را به یکدیگر متصل می‌کند تا زندگی را برای افراد آسان‌تر کند. مفهوم خانه هوشمند با کمک اینترنت اشیا راحتی را در زندگی ما به ارمغان می‌آورد [۱]. این دستگاه‌ها در خانه‌های هوشمند کارهایی را که ما برای آنها تعیین می‌کنیم را انجام می‌دهند و نیازی به

^۱Representational State Transfer

^۲Internet Of Things

حضور ما در آنجا نیست. در خانه های هوشمند، تمامی دستگاه های هوشمند، سنسورها و ... هرکدام بخشی از اینترنت اشیا هستند. به معنای دیگر، زیربنای خانه هوشمند، اینترنت اشیا است. در کل می توان گفت خانه هوشمند برای پیاده سازی نیاز به یک بستر زیرساخت خواهد داشت که این بستر توسط اینترنت اشیا فراهم می گردد.

۲-۲ ساختار اتوماسیون خانه هوشمند

امروزه نقش ماشین های هوشمند در زندگی ما بسیار ارزشمند است. با پیشرفت فن آوری های دیجیتالی مانند اینترنت اشیا و یادگیری ماشین^۱، بسیاری از تجهیزات برای کاربردهای مختلف توسعه یافته اند. حتی در خانه ها نیز، افراد برای انجام بسیاری از کارهای کوچک تا حیاتی مانند تمیز کردن، شستن، امنیت و غیره به ماشین ها نیاز خواهند داشت. در حال حاضر نیاز است تا سیستمی طراحی شود که بتواند پارامترهای مختلف مانند دما، رطوبت، شدت نور را کنترل کند. همچنین بتواند آتش و حرکت را تشخیص دهد و امنیت خانه را با کمک سیستم قفل درب اتوماتیک فراهم نماید. که تمامی این اقدامات با استفاده از اتوماسیون خانه امکان پذیر می باشد [۲].

مسائل مهم در سناریوی فعلی خانه هوشمند اتوماسیون و امنیت است. مشکل امنیت به دلیل اتصال شبکه دستگاه ها به اینترنت بوجود می آید. تمرکز به سمت ارائه محرمانگی^۲ و یکپارچگی داده های^۳ است که توسط اشیا ی خانه هوشمند تولید و رد و بدل می شوند. سربار^۴ نیز، یکی از موارد نگران کننده راه حل های خانه هوشمند خواهد بود. راحتی و امکان برطرف نمودن نیازهای کاربر با توجه به درخواست ها، نیاز اساسی برای اتوماسیون خانه هوشمند می باشد. اتوماسیون با یادگیری رفتار انسان نیز یکی از نگرانی های اصلی مفهوم خانه هوشمند می باشد [۱].

به همین خاطر در زمینه اتوماسیون خانه روش هایی پیشنهاد شده است که برای نمونه میتوان به کار آقای جاوای و همکاران در سال ۲۰۲۰ اشاره کرد که در آن یک سیستم اتوماسیون خانگی ارائه نموده اند که وضعیت خاموش و روشن شدن چراغ ها، فن ها، شیرهای آب و سیستم قفل درب را به طور خودکار و با کمک دستورات صوتی از طریق دستیار گوگل کنترل می کند. این سیستم پارامترهایی مانند حرکت، آتش، دما و رطوبت را نیز می تواند با استفاده از سنسورهای مربوطه تشخیص دهد. در این کار با استفاده از Node MCU و Google Assistant به طراحی و توسعه راه حل قابل اعتماد و کم هزینه برای اتوماسیون خانه و امنیت پرداخته می شود [۲].

¹Machine Learning

²Privacy

³Data Integrity

⁴Overhead

۳-۲ ساختار رایانش ابری

رایانش ابری، رایانه را در مکانی دور از کاربر و از طریق اینترنت پردازش می‌کند و تمام امکانات همچون فضای ذخیره‌سازی، پایگاه‌های داده و برنامه‌های مورد نیاز را در بستر خود جای داده‌است، بنابراین در صورت داشتن اینترنت می‌توان از هر دستگاهی به آنها دسترسی پیدا نمود و در این صورت نیازی به تهیه سرور یا استفاده از مرکز داده به صورت مجزا نمی‌باشد که این قابلیت‌ها می‌توانند هزینه‌ها را به میزان قابل توجهی کاهش دهند [۹]. همچنین امکان ذخیره حجم عظیمی از اطلاعات در فضای ابری وجود دارد. اما با وجود تمام این قابلیت‌ها به خاطر تاخیر بالای پردازش داده‌ها در ابر و ارسال آن به سمت کاربر این نیاز احساس می‌شود که تعدادی از فعالیت‌های مورد نیاز در لبه شبکه انجام شوند، این موضوع نیاز به محاسبات لبه را بهتر بیان خواهد نمود زیرا در زمان انجام پردازش‌ها در لبه شبکه تاخیر بسیار کمتر می‌شود، همچنین با توجه به این که محاسبات مه قدرت پردازش در شبکه را می‌تواند افزایش دهد، می‌توان با استفاده از محاسبات مه مقدار داده‌های ارسالی سمت ابر را بهینه نمود و کارایی سیستم را بهبود بخشید.

۴-۲ ساختار محاسبات مه در بروز رسانی تجهیزات

محاسبه مه یک گزینه امیدوار کننده برای به روزرسانی تجهیزات در خانه هوشمند می‌باشد. زیرا می‌تواند دوام شبکه را افزایش داده و تأخیرهای ارتباطی را در مقایسه با ابر کاهش دهد. در همین خصوص آقای اولاک و همکاران در سال ۲۰۲۱ یک الگوریتم و سیستم برای تجزیه و تحلیل الگوی ترافیک منطقه‌ای برای یک دوره زمانی خاص ارائه نموده‌اند که در مقایسه با الگوی اولیه بسیار سریع‌تر می‌باشد و همچنین تاخیر واگذاری، تاخیر انتشار، سرعت انتقال و تحرک خودرو را برای پیش‌بینی زمان بروزرسانی^۱ OTA کاهش می‌دهد [۵].

OTA به روشی از بروزرسانی گفته می‌شود که به طور مستقیم و از طریق ارتباط بی‌سیم^۲ توسط کمپانی منبع ارائه شده و هر کاربر بدون نیاز به استفاده از هیچ لوازم جانبی خاصی قادر به استفاده از آن خواهد بود در این تحقیق ما سعی داریم با الگوگیری از روش فوق تاخیر پاسخ گوئی را کاهش دهیم و همچنین اثربخشی بهتر را در نرم‌افزارهای خانه‌هوشمند با بروزرسانی سریع بررسی نماییم.

¹Over The Air

²Wireless

۲-۵ ساختارهای بروز در جهان واقعی

بروز رسانی نرم‌افزاری این امکان را میسر می‌کند تا بتوان با استفاده از تجهیزات موجود براساس نیاز شخص یا اشخاص کارایی‌های متفاوتی را از تجهیزات دریافت نمود به عنوان مثال با توجه به نیاز هر کاربر به صورت مجزا نور محیط را تغییر داد و یا در صورت بروز ایرادات و وجود حفره‌های امنیتی و یا مشکل در کارکرد تجهیزات در خانه‌هوشمند با بروزرسانی این مشکلات را برطرف نمود، در این خصوص آقای هانگ و همکاران در سال ۲۰۱۵ یک چهارچوب نرم‌افزاری و معماری سخت‌افزاری برای تبادل فایل‌های بروزرسانی از طریق تلفن همراه به دستگاه اینترنت‌اشیاء ارائه نموده‌اند تا بتوان از طریق گوشی همراه دستگاه‌ها را مجدداً برنامه‌ریزی نمود که در صورتی که به تجهیزات خانه‌هوشمند مستقر در محلی به صورت سیمی یا بی‌سیم متصل گردد امکان بروزرسانی تجهیزات را پیدا خواهد نمود [۶].

حال با توجه به کارهای انجام شده در زمینه بروزرسانی تجهیزات می‌توان به این نتیجه رسید که با یک دیدگاه بروزرسانی ساده می‌توان با صرف کمترین هزینه از تجهیزات موجود مجدداً با توجه به نیاز افراد استفاده نمود و در صورت وجود هرگونه ایراد در الگوریتم‌های موجود به وسیله بروزرسانی این مشکلات را برطرف نمود.

۲-۶ سیستم‌های نرم‌افزاری تحت وب^۱

امروزه طیف گسترده‌ای از سیستم‌های نرم‌افزاری را سیستم‌های نرم‌افزاری تحت وب تشکیل می‌دهند. اکثر این سیستم‌های نرم‌افزاری از معماری کلاینت-سرور^۲ یا مشتری-سرویس دهنده استفاده می‌کنند. این سیستم‌ها به وسیله شبکه جهانی وب^۳ یا همان اینترنت^۴ با اجزای فعال حاضر در سیستم ارتباط برقرار می‌کنند. اکثر اوقات، هسته اصلی این سیستم‌های نرم‌افزاری روی سیستم سمت سرور^۵ یا سرویس دهنده قرار می‌گیرد که به آن برنامه سمت سرور نیز می‌گویند. این برنامه لزوماً مورد استفاده کاربران عادی نرم‌افزار یا سرویس گیرنده‌ها قرار نمی‌گیرد.

از طرف دیگر برنامه‌ای که کاربران این سیستم نرم‌افزاری با آن ارتباط برقرار می‌کنند و ورودی‌های مد نظر خود را از آن طریق به سرور ارسال می‌کنند و منتظر گرفتن سرویس هستند، برنامه سمت کلاینت^۶ یا مشتری می‌گویند. به برنامه سمت سرور اصطلاحاً برنامه بک-اند^۷ و به برنامه سمت کلاینت یا مشتری برنامه فرانت-اند^۸

^۱ Web-based Software Systems

^۲ Client-Server

^۳ World Wide Web

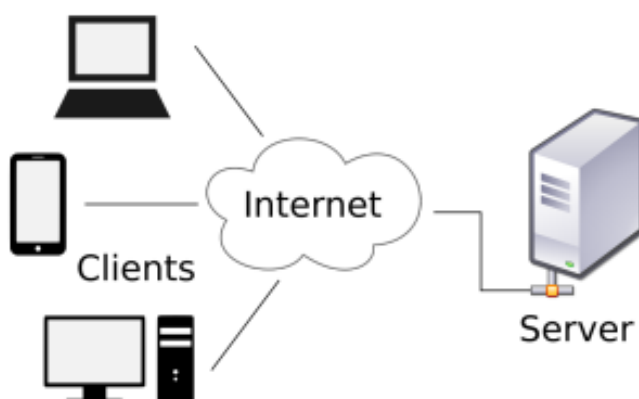
^۴ Internet

^۵ Server

^۶ Client

^۷ Back-end Application

^۸ Front-end Application



شکل ۲-۱: معماری کلاینت-سرور

نیز می گویند.

۲-۶-۱ وظایف برنامه سمت سرور

برنامه سمت سرور وظایف به خصوصی دارد که به اختصار آنها را نام می بریم:

۱. یکی از اصلی ترین وظایف برنامه سمت سرور، ایجاد وب سرویس های مناسب برای برنامه های سمت مشتری است تا برنامه سمت مشتری بتواند به راحتی از امکانات پیاده شده در سمت سرور استفاده کند.
۲. برنامه سمت سرور باید از امنیت بالایی برخوردار باشد و داده های ما را مورد خطر قرار ندهد.
۳. برنامه سمت سرور وظیفه مدیریت پایگاه داده در سمت سرور را بر عهده دارد.
۴. برنامه سمت سرور وظیفه دارد تا برای یک وب سرویس خاص بر اساس اینکه دسترسی های آن چگونه است، داده های مورد نظر را در اختیارش قرار دهد.

۲-۶-۲ وظایف برنامه سمت مشتری

برنامه سمت مشتری وظایف به خصوصی دارد که به اختصار آنها را نام می بریم:

۱. داشتن رابط کاربری مناسب و راحت برای کاربران انتهایی به نحوی که بتوانند کار های مورد نیاز خود را به راحتی انجام دهند.
۲. برقراری ارتباط امن و مطمئن بین کلاینت و سرور.

۳. استفاده از سرویس های تحت وب ارائه شده توسط برنامه سمت سرور به طوریکه تمام امکانات موجود را بر حسب نیاز پوشش دهد.

۴. استفاده بهینه از رابط های برنامه نویسی سمت سرور و وب سرویس های آن به طوری که عملکرد سیستم را مختل نکند.

۷-۲ معماری REST^۱

این نوع معماری یکی از گونه های مختلف معماری های برنامه های تحت وب می باشد. هر معماری در دنیای برنامه های تحت وب قوانین مخصوص به خودش را دارد که به اختصار به توضیح قوانین مهم حاکم بر معماری REST می پردازیم.

در ابتدا باید بگوییم که REST مخفف Representational State Transfer می باشد و در نوع خودش نسبت به معماری های دیگر مانند SOAP^۲ جدیدتر می باشد.

به طور کلی این معماری یک چهارچوب کلی برای توسعه دادن رابط های برنامه نویسی در برنامه سمت سرور یا همان وب سرویس های وب را ارائه می کند. به وب سرویس ها یا همان رابط هایی که طبق این معماری توسعه و عرضه می یابند، رابطه ها و وب سرویس های RESTful می گویند یعنی اینکه از قوانین REST تبعیت می کنند که اختصارا به آنها وب سرویس ها یا رابط های REST نیز می گویند.

در این معماری فرمت خروجی ها به صورت فرمت JSON^۳ (شکل ۲-۲) می باشد که مثلا در مقابل معماری SOAP بسیار متفاوت است زیرا که در آن معماری از فرمت XML^۴ (شکل ۲-۳) استفاده می شود.

^۱RESTful

^۲Protocol Access Object Simple

^۳Javascript Object Notation

^۴Extensible Markup Language


```
{
  "scores": [
    {
      "Away_Score": 2,
      "Away_Team": "Newcastle",
      "Home_Score": 2,
      "Home_Team": "Arsenal"
    },
    {
      "Away_Score": 2,
      "Away_Team": "Napoli",
      "Home_Score": 4,
      "Home_Team": "Liverpool"
    }
  ]
}
```

شکل ۲-۲: فرمت JSON

```
- <Parts>
- <Part>
  <Id>4478</Id>
  <Part_Name>1000 Ohm Resistor</Part_Name>
  <Total_Available>25000</Total_Available>
  <Price>0.01</Price>
</Part>
- <Part>
  <Id>3328</Id>
  <Part_Name>15000 Ohm Resistor</Part_Name>
  <Total_Available>75000</Total_Available>
  <Price>0.02</Price>
</Part>
- <Part>
  <Id>4725</Id>
  <Part_Name>555 Timer IC</Part_Name>
  <Total_Available>1500</Total_Available>
  <Price>0.25</Price>
</Part>
</Parts>
```

شکل ۲-۳: فرمت XML

اکنون به برخی از قوانین معماری REST می پردازیم :

۱. در معماری REST، از پروتکل HTTP^۱ استفاده می شود و بسته های ارسالی از جنس بسته های HTTP هستند.

۲. در معماری REST تلاش بر این است که از تمامی متد های رایج HTTP برای کار های مختلف استفاده کنیم. این متد ها شامل موارد زیر می باشد:

(آ) متد GET که برای واکنشی و گرفتن اطلاعات مورد استفاده قرار می گیرد.

(ب) متد POST که برای تغییر حالت برنامه سمت سرور مثلاً برای ذخیره یک داده جدید و یا یک عملیات جدید که باعث تغییر داده های سمت سرور می شود مورد استفاده قرار می گیرد.

(ج) متد PUT که برای بروزرسانی کلی داده ها در سمت سرور مورد استفاده قرار می گیرد.

(د) متد PATCH که برای بروزرسانی جزئی داده ها در سمت سرور مورد استفاده قرار می گیرد.

(ه) متد DELETE که برای حذف داده ها در سمت سرور مورد استفاده قرار می گیرد.

(و) متد HEAD که یک متد با بدنه خالی است و بیشتر برای مقاصد اشکال زدایی و اطلاع از فعال بودن رابط های سمت سرور مورد استفاده قرار می گیرد.

۳. برخی قوانین برای تعیین نام و مسیر های این رابط ها یا وب سرویس های REST نیز وجود دارد. به عنوان مثال قوانین REST بیان می کند که انتخاب نام برای URL های رابط ها نباید از فعل استفاده شود. و برخی از این قبیل قوانین که به توضیح آنها نمی پردازیم.

۸-۲ جنگو^۲

Django یک فریم ورک^۳ منبع باز^۴ و رایگان برای توسعه وب پایتون^۵ و ابزاری بسیار انعطاف پذیر برای توسعه وب است. که می تواند برای ایجاد هر نوع وب سایت یا برنامه ای که لازم است مورد استفاده قرار گیرد.

فریم ورک مجموعه ای از ماژول^۶ ها است که عناصر از پیش ساخته شده ای را فراهم می کند که کدگذاری^۷ را کارآمدتر و پایدارتر می کند.

^۱Hyper Text Transfer Protocol

^۲Django

^۳Framework

^۴Open Source

^۵Python

^۶Module

^۷Programming

جنگو برای هر پروژه توسعه وب یک انتخاب عالی است. این به ویژه برای سایت های رسانه های اجتماعی یا سایت های تجارت الکترونیکی که به یک پایه قوی و ایمن نیاز دارند بسیار مناسب است زیرا فریم ورک جنگو دارای ویژگی های داخلی است که برای محافظت از داده های حساس ، معاملات و تأیید اعتبار کاربر بسیار مناسب است.

این فریم ورک امنیت را به طور جدی ایفا می کند و به توسعه دهندگان کمک می کند تا از بسیاری از خطاهای امنیتی مشترک جلوگیری کنند.

برخی از شلوغ ترین سایت ها در وب، توانایی این چارچوب را به سرعت و انعطاف پذیری در مقیاس می گیرند.

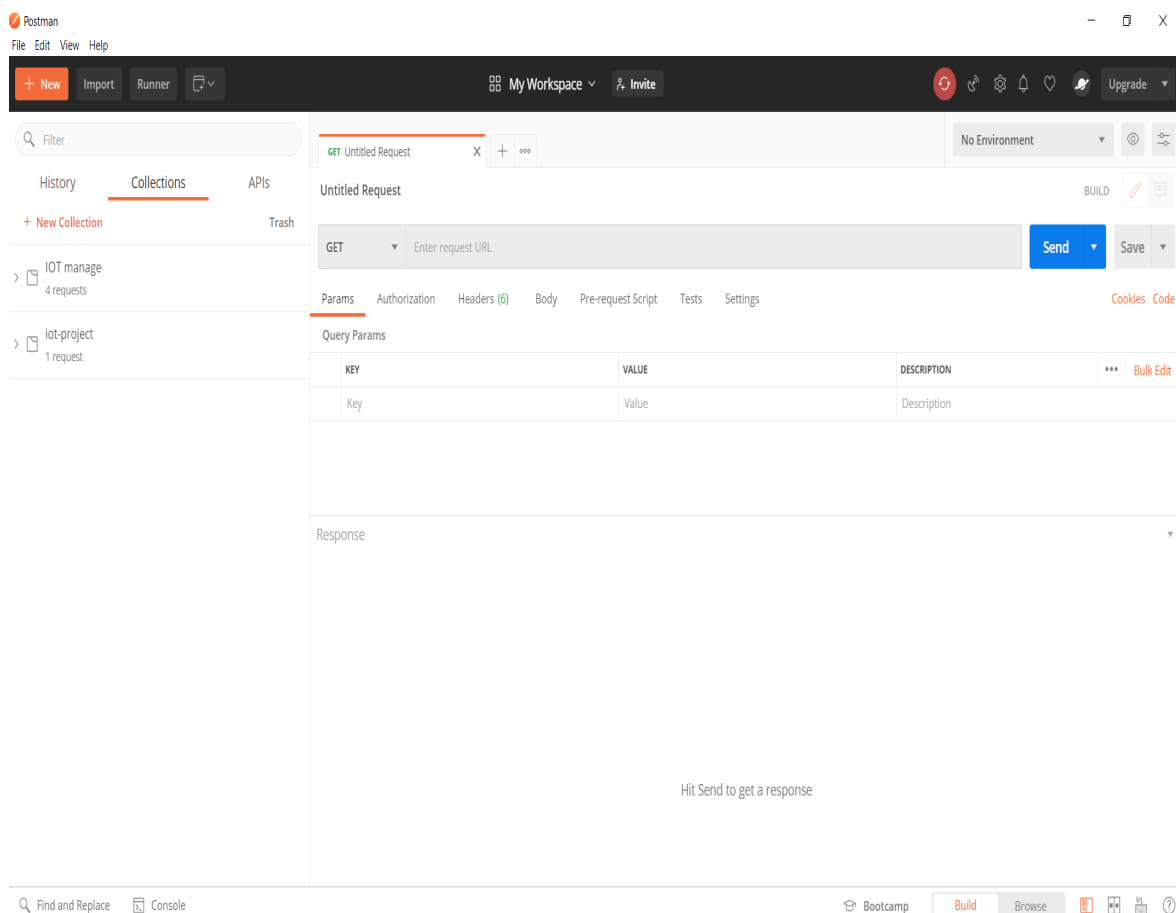
۹-۲ جنگو-رست^۱

جنگو رست فریم ورک که به اختصار به آن DRF گفته می شود، یک فریمورک فوق العاده قدرتمند و کاربردی برای ایجاد API های وب می باشد. به صورت کلی API ابزاری است که بنا به نیاز ما اطلاعات را از وبسایت استخراج می کند. بنابراین API ها از اهمیت بالایی در برنامه نویسی برخوردار می باشند. جنگو رست فریمورک یکی از ابزارهای قدرتمند پیاده سازی WebApi ها می باشد که با زبان برنامه نویسی پایتون توسعه داده شده است.

در جنگو رست فریم ورک ما می توانیم بدون اینکه فرانت-اند را در اختیار داشته باشیم، بک-اند اپلیکیشن خود را پیاده سازی و تست کنیم.

بدین صورت که می توان توسط URL یا نرم افزارهایی که میتوانند درخواست HTTP بدهند، با بک-اند در ارتباط باشیم و آن را تست کنیم. در این پروژه ما از نرم افزار POSTMAN برای تست کردن پروژه استفاده می کنیم. شکل ۲-۴ تصویری از محیط Postman است.

¹ Django-rest-framework



شكل ٢-٤ : Postman

فصل سوم

معماری

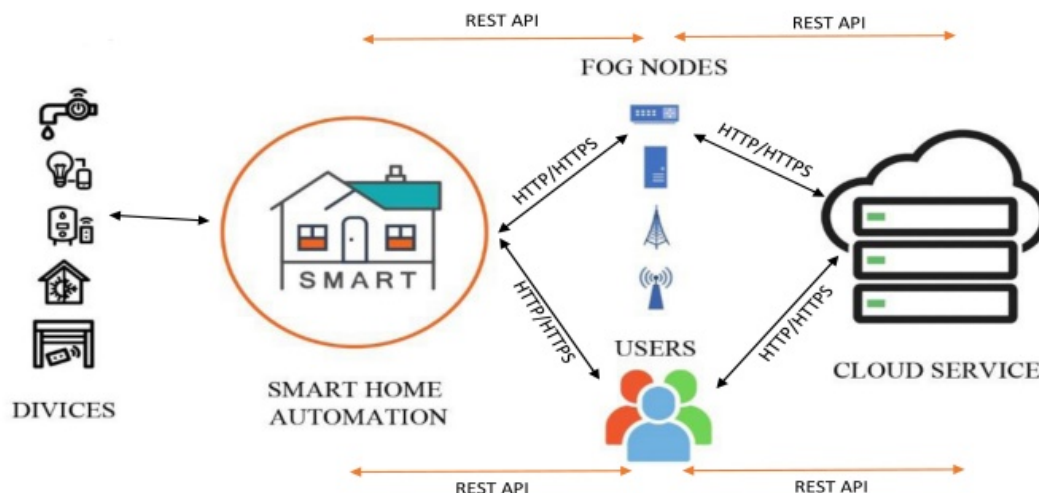
۳-۱ طرح کلی مساله

معماری کلی سیستم در شکل ۳-۱ قابل مشاهده است. در این شکل تمامی اجزایی که برای سیستم فرض کرده ایم در تصویر وجود دارند.

همچنین پروتکل هایی که در این معماری فرض شده اند نیز قابل مشاهده هستند.

در این پروژه ما پیاده سازی بخش بک-اند نرم افزار سیستم را برعهده داریم و در این حوزه، برای سرورهای Smart Home Automation , Fog Nods , Cloud Service نرم افزار بک-اند مورد نیاز آنها را پیاده سازی میکنیم. علت بیان معماری پیشنهادی ارائه یک سیستم هوشمند یکپارچه خواهد بود که قابلیت بروزرسانی تجهیزات موجود در خود را داشته و این امکان را در بستر خانه هوشمند ایجاد می نماید که تنها با یک بروزرسانی سیستم، از حملات امنیتی در آینده، که سیستم برای آنها راهکاری در نظر نگرفته است جلوگیری و خطاهای موجود در سیستم را، که از دید طراحان سیستم در زمان طراحی مخفی مانده است برطرف نماید. همچنین این معماری امکان استفاده برای مدت زمان بیشتر از تجهیزات موجود را در اختیار کاربران قرار می دهد که باعث کاهش هزینه و استفاده بلند مدت از تجهیزات می گردد.

روند کلی سیستم به این شکل خواهد بود که تمامی Device ها داخل اتوماسیون خانه هوشمند قرار دارند و



شکل ۳-۱: معماری کلی سیستم

کاربر میتواند به صورت مستقیم آنها را کنترل کند.

بخش Cloud وظیفه آنالیز کردن داده هایی که از سمت خانه های هوشمند می آیند را برعهده دارد و بخش Fog Nodes برای کاهش فشار اطلاعات سرور تعبیه شده و بخشی از محاسبات را انجام میدهد و نتایج خود را برای سرور بیان میکند.

۳-۲ نقشه کلی وب سایت

در این بخش معماری کلی بخش های اصلی بک-اند سیستم پیاده سازی شده قابل مشاهده می باشد. (شکل ۳-۲)

حال وظایف هر بخش را به طور کاملاً خلاصه بیان می کنیم.

۳-۲-۱ Cloud

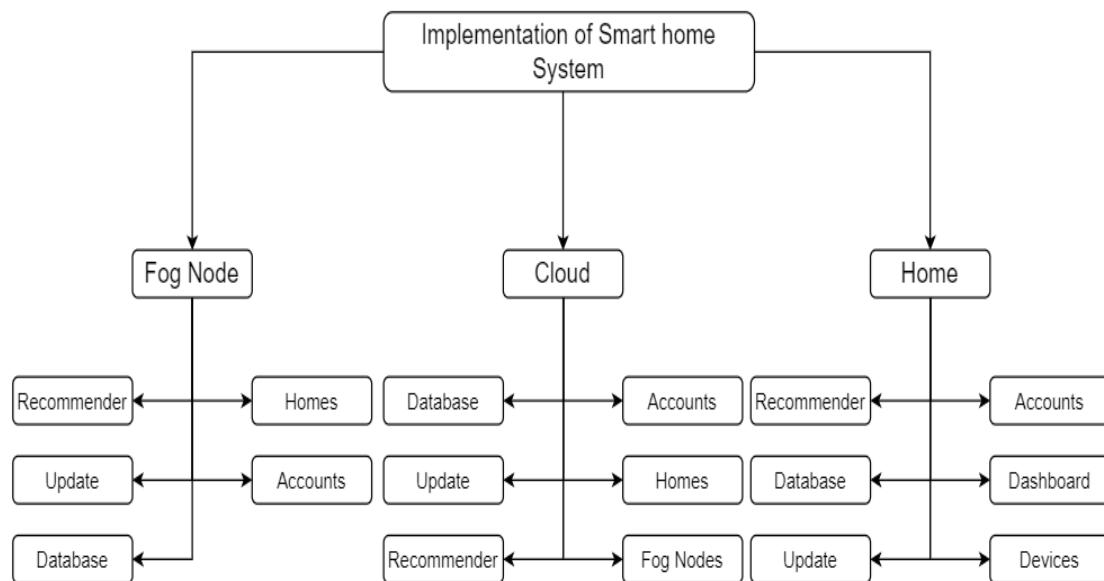
۱. Database: پایگاه داده اصلی سیستم در این بخش قرار دارد و شامل تمامی نود های مه و خانه ها می باشد و همچنین محل ذخیره سازی و بررسی صحت تمام اطلاعات کاربران و خانه ها و دستگاه ها می باشد.

۲. Accounts: همه کاربران در ابتدا برای استفاده از امکانات سیستم ابتدا باید احراز هویت^۱ شوند که این بخش شامل ۲ نوع کاربر می شود. مشتری^۲ و پیمانکار^۳ که هر کدام سطح دسترسی و نقش متفاوتی دارند.

^۱Authentication

^۲Customer

^۳Contractor



شکل ۳-۲: نقشه کلی سایت

۳. Update: هنگامی که بروزرسانی جدیدی برای هریک از قسمت های سیستم بیاید این بخش مسئولیت اطلاع رسانی به آن بخش و ارسال بروزرسانی جدید را برعهده دارد.

۴. Houses: تمامی خانه های موجود در سیستم در این بخش وجود دارند و احراز هویت و مدیریت می شوند.

۵. Recommender: این بخش تمامی نتایج سیستم توصیه کننده را در خود ذخیره می کند و همیشه با توجه به گزارش های بخش Fog به روزرسانی می شود. همچنین در صورت جا به جایی کاربر از یک خانه به خانه دیگر به راحتی تنظیمات کاربر را منتقل می کند تا کاربران نگرانی ای در مورد جا به جایی و از دست رفتن تنظیمات شخصی خود نباشد.

۶. Fog Nodes: تمامی نود های مه در این بخش به همراه تنظیماتشان قرار دارند تا گزارشات آنها دسته بندی و ارتباط با آنها آسان شود.

Home ۲-۲-۳

۱. Database: تمام اطلاعات کاربران خانه و دستگاه ها و دسترسی های موجود در خانه در این بخش قرار دارند. که میتوان آنرا بخش بسیار کوچکی از دیتابیس اصلی سیستم در قسمت ابر دانست.

۲. Accounts: همه کاربران در ابتدا برای استفاده از امکانات سیستم ابتدا باید احراز هویت شوند که این بخش شامل ۲ نوع کاربر می شود. مشتری و پیمانکار که هرکدام سطح دسترسی و نقش متفاوتی دارند.
۳. Update: هنگامی که بروزرسانی جدیدی برای سرور خانه و یا Device ها در دسترس باشد، در این قسمت نمایش داده می شود و با گرفتن دستور مشخص خود، اقدام به دانلود و بروزرسانی خود می کند.
۴. Dashboard: این بخش پنل کاربری و دستگاه های داخل خانه و وضعیت آنها را نشان می دهد و می توان از این طریق دستگاه هارا کنترل کرد.
۵. Recommender: این بخش گزارش تمام رخداد های دستگاه هارا برای بخش Fog ارسال می کند و نتایج محاسبات آنرا می گیرد و ذخیره می کند تا کاربر تجربه کاربری بهتری را تجربه کند.
۶. Devices: این بخش شامل تمام دستگاه های موجود در خانه است که شامل اطلاعات دستگاه و وضعیت کنونی آن می باشد. همچنین دارای یک بخش است که در آن نرم افزار اجرا کننده دستگاه^۱ واقع است که میتوان با بروزرسانی آن، بروزرسانی های جدید دستگاه را بر روی آن قرار داد و نصب کرد تا دیگر نیاز به اقدام مستقیم و یا فیزیکی برای بروزرسانی دستگاه ها نباشد.

۳-۲-۳ FOG NODE

۱. Database: در این بخش اطلاعات تمام خانه هایی که زیرمجموعه این گره Fog هستند ذخیره شده است
۲. Accounts: همه کاربران در ابتدا برای استفاده از امکانات سیستم ابتدا باید احراز هویت شوند که این بخش شامل ۲ نوع کاربر می شود. مشتری و پیمانکار که هرکدام سطح دسترسی و نقش متفاوتی دارند.
۳. Update: هنگامی که بروزرسانی جدیدی برای سیستم Fog از سمت سرور فرستاده شود، این بخش وظیفه بررسی و انجام فرآیند بروزرسانی خود را بر عهده دارد.
۴. Houses: تمامی خانه های موجود که زیرمجموعه این Fog هستند در این بخش وجود دارند و احراز هویت و مدیریت می شوند.
۵. Recommender: این بخش تمام گزارشاتی که از سمت خانه های هوشمند می آیند را جمع آوری می کند و با الگوریتم های سیستم توصیه ای خود نتایجی را برای کاربران و خانه های هوشمند متفاوت، با دستگاه های متفاوت بدست می آورد و یکبار برای ذخیره سازی برای ابر و یکبار برای پیاده سازی برای کنترلر خانه هوشمند ارسال می کند.

^۱ Firmware

۳-۳ معماری کلی پایگاه داده

در شکل ۳-۳ شمای کلی از جداول مهم پایگاه داده سیستم ما نمایش داده شده است. در اینجا جداول را به طور خلاصه بررسی خواهیم کرد.

۱. User: این جدول اطلاعات شامل اطلاعات کاربری تمامی کاربران احراز هویت شده^۱ است و هر نوع کاربر از قبیل Customer و Contractor از این جدول ارثبری می کنند.

۲. Contractor: این جدول مختص پیمانکاران سیستم خانه هوشمند است که عموماً خدمات سخت افزاری و پشتیبانی خانه های هوشمند را برعهده دارند.

۳. Customer: جدول مربوط به کاربران عادی سیستم که هرکدام یک رکورد از این جدول را تشکیل می دهند.

۴. Device: هر ورژن از هر دستگاهی که داخل اتوماسیون خانه هوشمند ما استفاده می شود، یک رکورد از این جدول را شامل می شود تا تمام اطلاعات لازم برای نصب و استفاده از دستگاه در آن ذخیره شود..

۵. Device In Used: هر دستگاهی که در هر خانه نصب می شود به صورت یک رکورد از این جدول ذخیره می شود. و از مهم ترین فیلدهای آن Device ، House هستند که توسط کلید خارجی^۳ به جداول House ، Device متصل شده اند و هرکدام به یک رکورد از آنها ارجاع داده می شود.

۶. House: این جدول شامل تمام خانه هایی می شود که داخل سیستم خانه هوشمند ثبت شده اند و تمامی اطلاعات آن داخل این جدول ذخیره می شوند. برخی فیلدهای مهم آن عبارتند از:

مالک خانه که با فیلد Owner مشخص شده است و یک کلید خارجی به جدول Customer است.

اعضای خانه که توسط یک رابطه چند به چند^۴ به جدول Customer متصل می شود.

و پیمانکار اصلی که مسئولیت نصب و پشتیبانی امور خانه را بر عهده دارد و به صوت یک کلید خارجی به جدول Contractor متصل می شود.

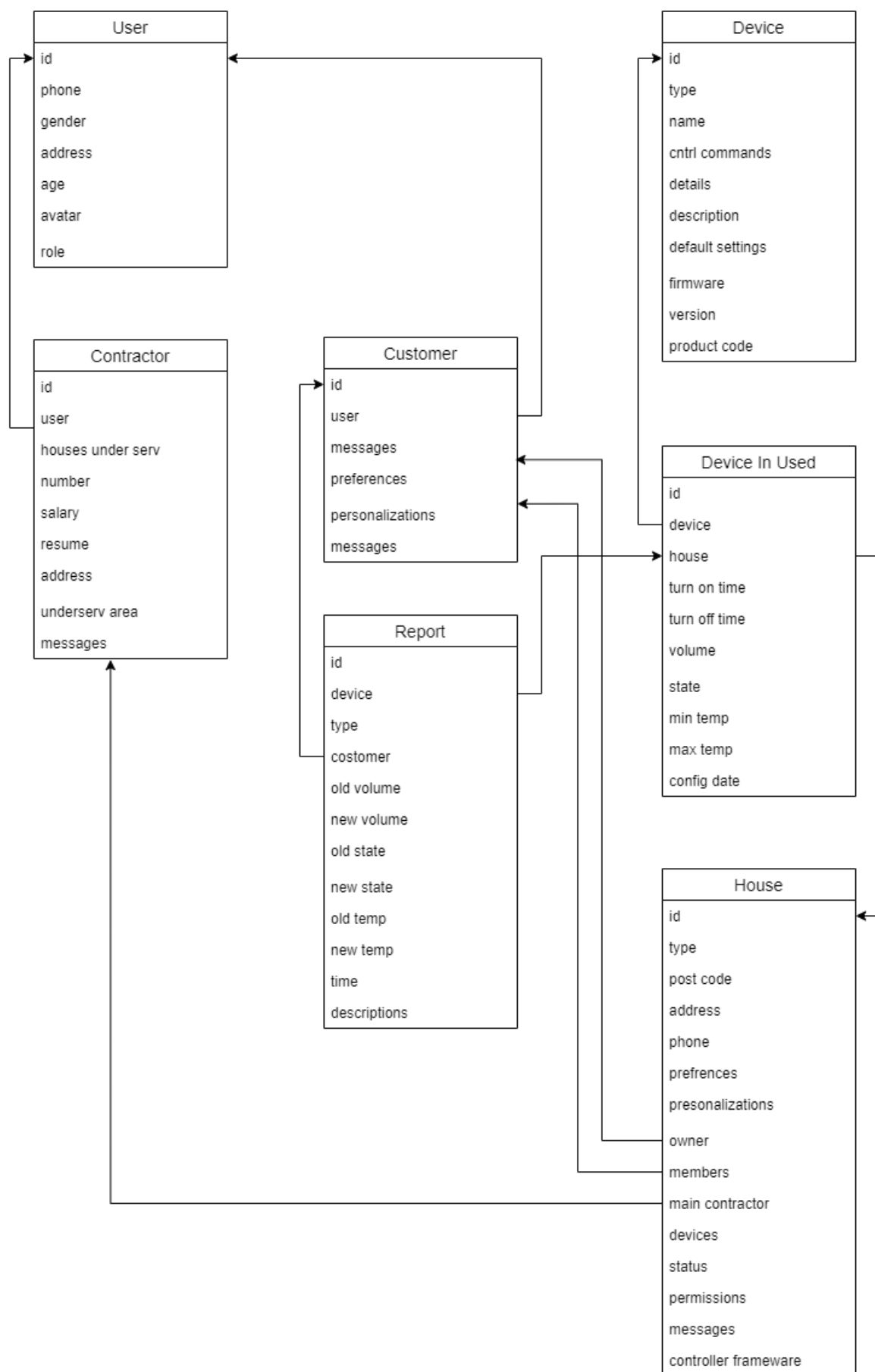
۷. Report: این جدول به این منظور طراحی شده است تا تمامی فعالیت های افراد بر روی دستگاه ها را ثبت کند تا با پردازش روی دیتاهای بدست آمده از این رکورد ها بتوان برای کاربران مختلف توسط یک

¹ Authenticated

² Field

³ Foreign Key

⁴ Many to Many Relationship



شکل ۳-۳: شمای کلی پایگاه داده

سیستم پیشنهاد دهنده تهیه کرد و مقداری از ترجیحات کاربر را برای رضایت مندی بیشتر آنها پیشبینی کرد. همچنین این جدول توسط یک کلید خارجی، با فیلد های Customer ، Device به جداول Device In Used ، Customer متصل می شود.

فصل چهارم

پیاده سازی

در این فصل سعی داریم برخی از قسمت های پیاده سازی شده را بررسی و تست کنیم.

۴-۱ بررسی برخی API ها

در این بخش ابتدا تعدادی از API های مهم برنامه را بررسی می کنیم.

۴-۱-۱ احراز هویت و امور حساب ها

این API ها عموماً مربوط به ثبت نام^۱، ورود^۲ خروج^۳ و ... می باشد. در شکل ۴-۱ API های کلی در مدل Accounts که مربوط به احراز هویت می شوند نمایش داده شده اند.

^۱Registration

^۲Login

^۳Log out

```
accounts/ dj-rest-auth/ password/reset/ [name='rest_password_reset']
accounts/ dj-rest-auth/ password/reset/confirm/ [name='rest_password_reset_confirm']
accounts/ dj-rest-auth/ login/ [name='rest_login']
accounts/ dj-rest-auth/ logout/ [name='rest_logout']
accounts/ dj-rest-auth/ user/ [name='rest_user_details']
accounts/ dj-rest-auth/ password/change/ [name='rest_password_change']
accounts/ dj-rest-auth/registration/
accounts/ customer/
accounts/ contractor/
accounts/ account/login/ [name='login']
accounts/ account/two_factor/setup/ [name='setup']
accounts/ account/two_factor/qrcode/ [name='qr']
accounts/ account/two_factor/setup/complete/ [name='setup_complete']
accounts/ account/two_factor/backup/tokens/ [name='backup_tokens']
accounts/ account/two_factor/backup/phone/register/ [name='phone_create']
accounts/ account/two_factor/backup/phone/unregister/<int:pk>/ [name='phone_delete']
accounts/ account/two_factor/ [name='profile']
accounts/ account/two_factor/disable/ [name='disable']
```

شکل ۴-۱: Accounts urls

طبق شکل ۴-۱ تمامی API های این بخش با پیشوند accounts شروع می شوند.

پس از آن، API هایی که با آدرس dj-rest-auth شروع می شوند مربوط به امور ورود، خروج، ثبت نام، تغییر رمز^۱ و ... می باشند.

به عنوان مثال برای ورود به حساب، باید یک درخواست^۲ از نوع HTTP به آدرس http://127.0.0.1:8000/accounts/dj-rest-auth/login/

به متد POST زد و در بدنه^۳ درخواست، نام کاربری^۴ و رمز اکانت را وارد کرد.

با انجام درست این کار، یک توکن^۵ برای کاربر اختصاص داده می شود که کاربر برای تایید هویت خود

همواره در درخواست های خود باید این توکن را در هدر^۶ درخواست خود قرار دهد.

همچنین کوکی^۷ و سشن^۸ در مرورگر (در صورت استفاده از مرورگر) نیز تنظیم می شوند.

در تصویر ۴-۲ یک کاربر با این روش با نرم افزار postman اقدام به لاگین می کند.

همچنین API هایی که با پیشوند Customers، Contranctor هستند، برای اختصاص نقش مشتری یا

پیمانکار به کاربران، استفاده می شوند.

¹Password

²Request

³Body

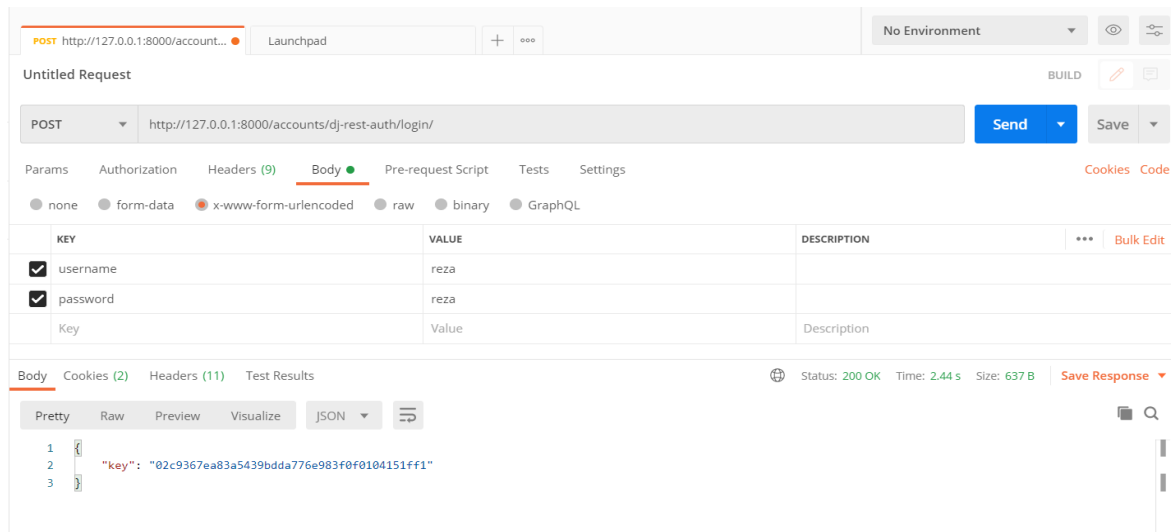
⁴Username

⁵Token

⁶Header

⁷Cookie

⁸Session



شکل ۲-۴: لاگین با کاربر reza

در نهایت API هایی که با پیشوند account/two-factor/ شروع می شوند مربوط به تنظیمات احراز هویت دومرحله ای^۱ برای امنیت بیشتر حساب ها می باشند.

نحوه کارکرد آن نیز به این گونه است که هرکاربر می تواند با استفاده از آدرس

http://127.0.0.1:8000/accounts/account/two-factor/setup/ یکبار نام کاربری و رمز خود را بزند و با اسکن کردن QRcode توسط یک نرم افزار تولید کننده رمز که میتوان آنرا به راحتی بر روی تلفن همراه خود نصب کند، آنرا پیکریندی^۲ کرده و به آن متصل شود. از آن پس هنگام ورود به سیستم باید رمز ۶رقمی تولید شده توسط نرم افزار را وارد کند تا اجازه لاگین را پیدا کند.

۲-۱-۴ کنترل دستگاه ها

این بخش مربوط به کنترل دستگاه ها و دستگاه های داخل خانه است.

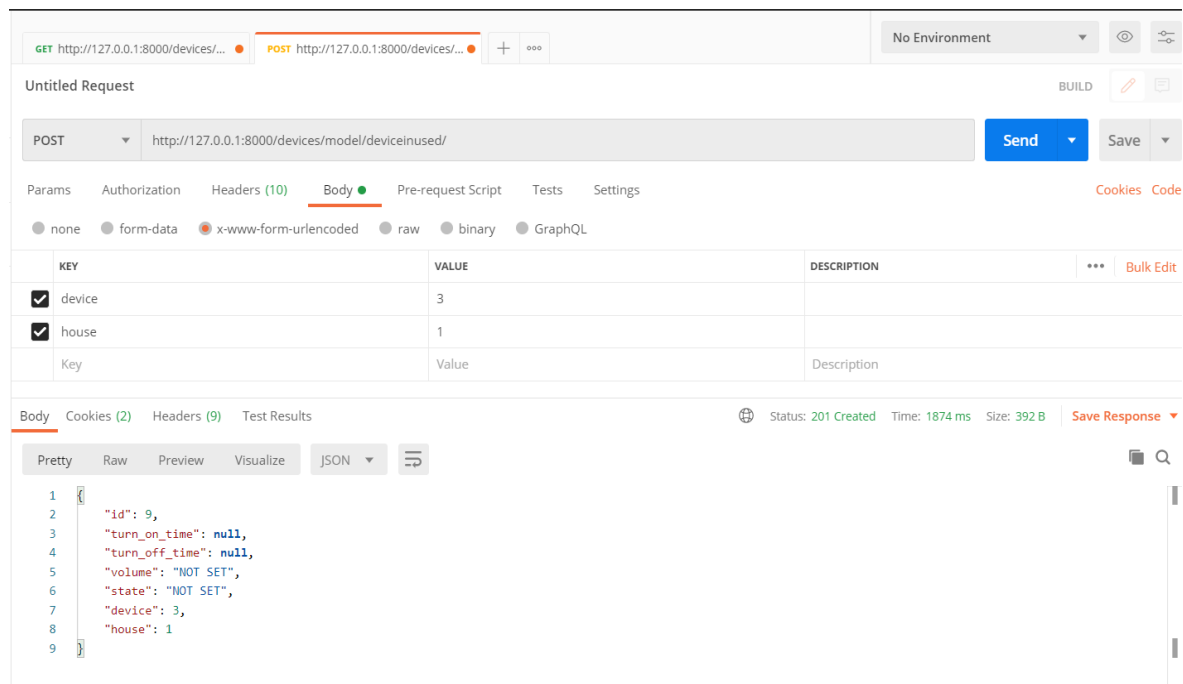
این API ها با آدرس http://127.0.0.1:8000/devices/model/ شامل ۲ API کلی می شوند که با متد های متخلف میتوانند روی جداول Device، DeviceInUsed، تغییراتی از جمله اضافه، حذف، بروزرسانی، گرفتن اطلاعات و ... را انجام دهند.

برای مثال در شکل ۳-۴ کاربر دستگاهی با آی دی^۳ شماره ۳ را به خانه با آی دی شماره ۱ اضافه کند، که در صورت درست انجام شدن، رکورد جدید به عنوان پاسخ در قالب JSON برگردانده می شود.

^۱Two-Factor Authentication

^۲Configuration

^۳Id



شکل ۴-۳: postman

۴-۱-۳ پنل ادمین

این بخش مربوط به پنل ادمین جنگو است که برای این سیستم شخصی سازی شده تا ادمین بتواند دسترسی مستقیم روی پایگاه داده داشته باشد و بر آن نظارت کند.

برای این بخش ۲ API در نظر گرفتیم، یکی از آنها با آدرس `http://127.0.0.1:8000/admin/` که در اصل یک خمره عسل^۱ است و برای یه دام انداختن هکر ها استفاده می شود.

و آدرس دیگری `http://127.0.0.1:8000/admin-panel/` است که پنل اصلی ادمین جنگو را بر روی این آدرس قرار داده ایم. شکل ۴-۴ تصویری از محیط ادمین جنگو سیستم پیاده سازی شده است.

۴-۱-۴ کنترل خانه

در این بخش میتوان با API های موجود، رکورد های خانه را نمایش داد، آپدیت کرد، اضافه کرد و ... آدرس این API ها با پیشوند `http://127.0.0.1:8000/homes/model/house/` می باشند.

به عنوان مثال، خروجی این API با متد GET لیست تمام خانه های موجود است. (شکل ۴-۵)

^۱Honey Pot

Django administration

Site administration

ACCOUNTS	
Contractors	+ Add Change
Customers	+ Add Change
Users	+ Add Change

ACCOUNTS	
Email addresses	+ Add Change

ADMIN_HONEYPOT	
Login attempts	Change

AUTH TOKEN	
Tokens	+ Add Change

AUTHENTICATION AND AUTHORIZATION	
Groups	+ Add Change

DEVICE_CNTRL	
Device in useds	+ Add Change
Devices	+ Add Change
Reportss	+ Add Change

Recent actions

My actions

- [ahmad4](#) User
- [ahmad4](#) User
- [ahmad4](#) User
- [ahmad4](#) User
- [+ ssss](#) User
- [+ hadi](#) Customer
- [✗ f9d6d8fd68237189168c3276868a...](#) Token
- [✗ 25f3819e0919fe09e45b9fbb8922f...](#) Token
- [erfan](#) User
- [erfan](#) User

Admin panel : شکل ۴-۴

Django REST framework

[Api Root](#) / [House List](#)

House List

[OPTIONS](#)
[GET](#)

GET /homes/model1/house/

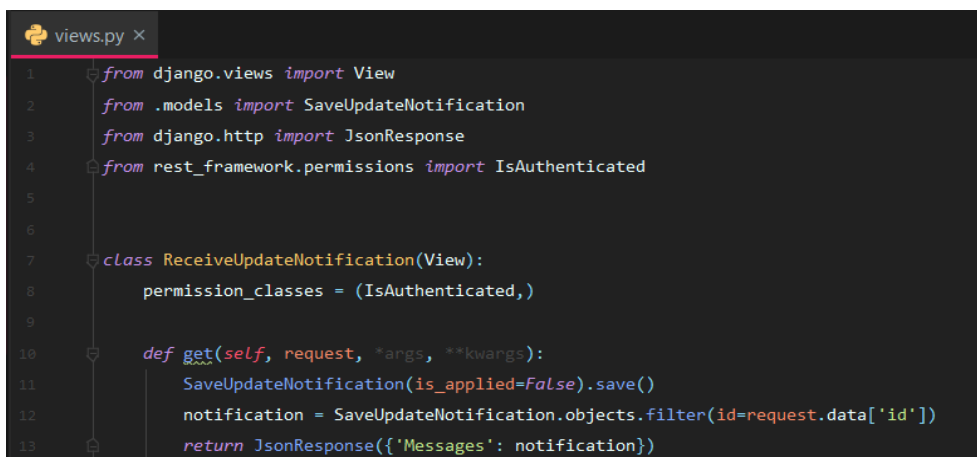
HTTP 200 OK
 Allow: GET, POST, HEAD, OPTIONS
 Content-Type: application/json
 Vary: Accept

```
[
  {
    "id": 1,
    "post_code": "53214123",
    "address": "teh,azadi",
    "phone": "",
    "is_active": true,
    "owner": 1,
    "main_contractor": null,
    "members": [
      2
    ]
  },
  {
    "id": 2,
    "post_code": "123456789",
    "address": "Tehran, Iran",
    "phone": "09111111111",
    "is_active": true,
    "owner": 3,
    "main_contractor": null,
    "members": []
  }
]
```

django rest interface in web browser : شکل ۴-۵

۴-۲ اجرای سناریو معماری بروزرسانی شده

در این سناریو با توجه به بازخوردی^۱ که توسط برخی کاربران در مورد سطح دسترسی کاربران عادی در خانه های هوشمند گرفته شد به این نتیجه رسیدیم که برخی دسترسی های مهم به سیستم کنترلر هوشمند خانه را محدود کنیم و آنرا تنها به برخی کاربران به عنوان مثال کاربر مالک^۲ خانه بدهیم. در سیستم پیاده سازی شده اولیه تمام اعضای عضو یک خانه^۳ میتوانند پیام های بروزرسانی را مشاهده کنند و حتی دستور بروز رسانی سیستم را بدهند. در این سناریوی فرضی، بازخوردهایی مالکان خانه ها به سیستم ما داده اند و از دسترسی بی چون و چرای اعضای خانه برای آپدیت سیستم، شکایت کرده اند. در اینجا ما تصمیم گرفتیم که یک بروزرسانی برای سیستم کنترلر خانه هوشمند منتشر کنیم و در آن، دسترسی به پیامهای بروزرسانی و انجام فرایند بروزرسانی را فقط در اختیار مالک خانه قرار دهیم. در ابتدا بلوک بروزرسانی سیستم از رابط شکل ۴-۶ برای نمایش پیام به کاربران استفاده میکرد.



```

views.py x
1 from django.views import View
2 from .models import SaveUpdateNotification
3 from django.http import JsonResponse
4 from rest_framework.permissions import IsAuthenticated
5
6
7 class ReceiveUpdateNotification(View):
8     permission_classes = (IsAuthenticated,)
9
10    def get(self, request, *args, **kwargs):
11        SaveUpdateNotification(is_applied=False).save()
12        notification = SaveUpdateNotification.objects.filter(id=request.data['id'])
13        return JsonResponse({'Messages': notification})

```

شکل ۴-۶: updates.views.py

برای رفع این مشکل ما یک سطح دسترسی جدید در سیستم طراحی میکنیم که در آن فقط به مالکان خانه اختصاص دارد.

در این قسمت از سطح دسترسی های استاندارد جنگو استفاده میکنیم.

در ابتدا در اپلیکیشن حساب ها^۴ یک فایل جدید^۵ اضافه میکنیم و دسترسی های خود را در آن تعریف و مدیریت میکنیم (شکل ۴-۷).

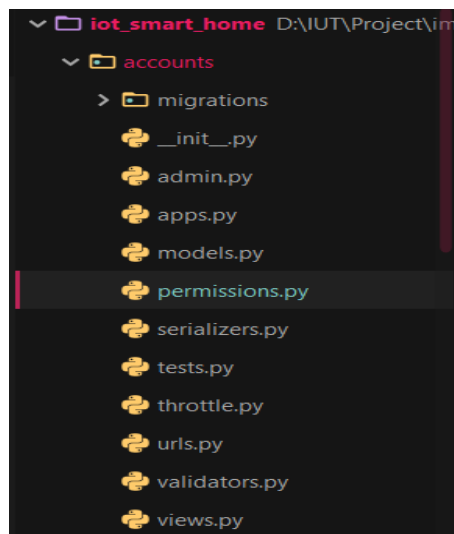
^۱Feedback

^۲Owner

^۳Members

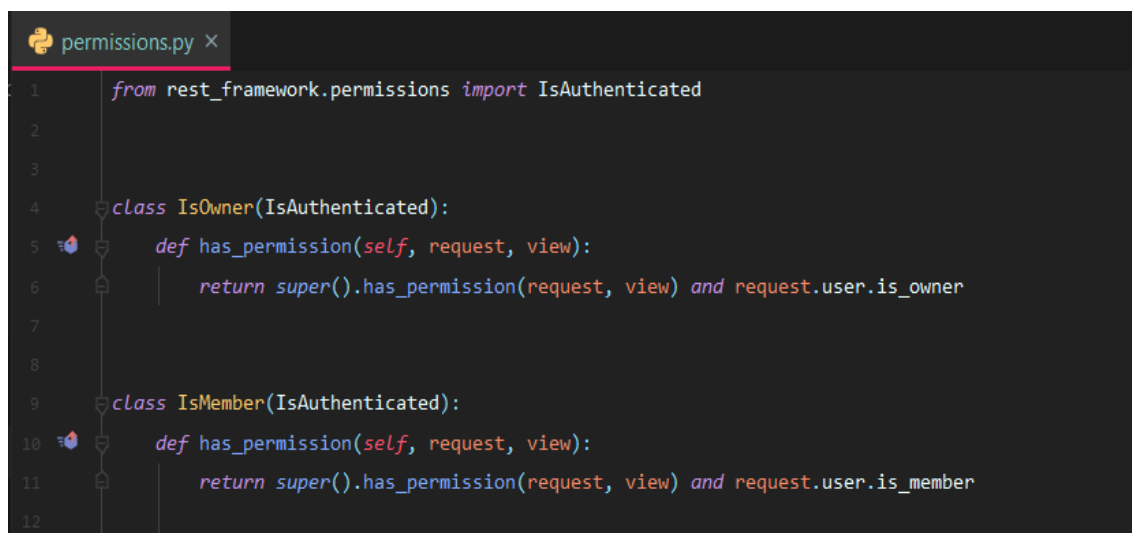
^۴Accounts

^۵permissions.py



شکل ۴-۷: Accounts

حال کلاس های دسترسی را با توجه به نیازهای سطح دسترسی به سیستم، پیاده سازی میکنیم (شکل ۴-۸).



شکل ۴-۸: permissions.py

کلاس IsOwner : این کلاس از سطوح دسترسی^۱ احراز هویت شده^۲ در جنگو ارثبری^۳ میکند و یک صفت^۴ به آن اضافه میکند و به ما اجازه میدهد تا از آن در سیستم خود استفاده کنیم. و در صورت استفاده از آن، چیزی که به ما برمیگرداند این است که آیا این کاربری که درخواست^۵ را فرستاده است مالک^۶ خانه است

^۱permissions

^۲IsAuthenticated

^۳Inheritance

^۴Attribute

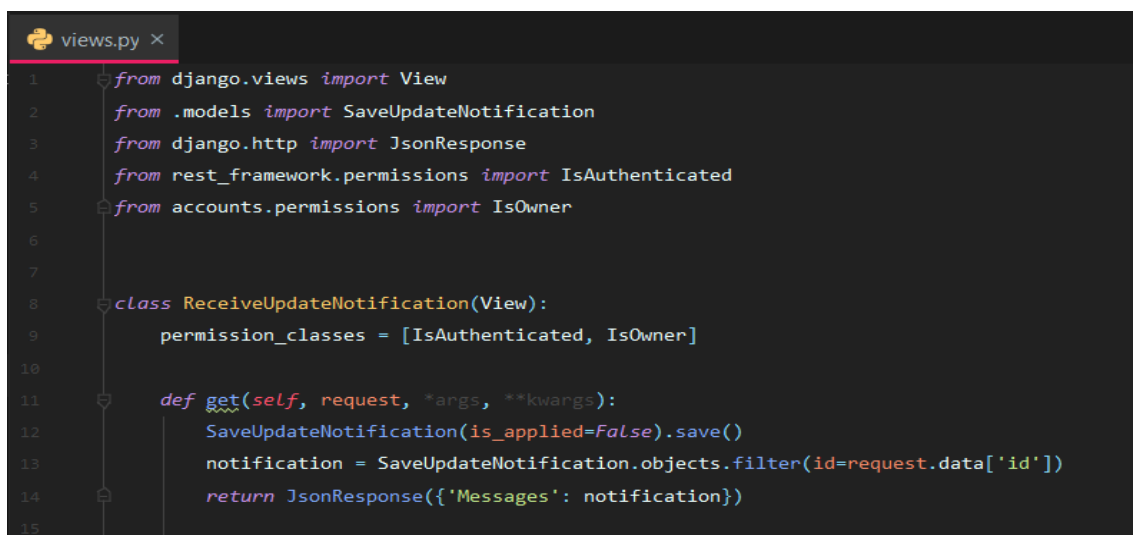
^۵request

^۶Owner

یا خیر.

کلاس IsMember : این کلاس از سطوح دسترسی احراز هویت شده در جنگو ارثبری میکند و یک صفت به آن اضافه میکند و به ما اجازه میدهد تا از آن در سیستم خود استفاده کنیم. و در صورت استفاده از آن، چیزی که به ما برمیگرداند این است که آیا این کاربری که درخواست را فرستاده است عضو^۱ خانه است یا خیر.

حال دسترسی به رابط را محدود به کاربران مالک خانه میکنیم و با ارثبری از احراز هویت جدیدی که برای کاربران مالک خانه تهیه کردیم، فقط کاربران مالک خانه دسترسی به بخش بروز رسانی سیستم خانه هوشمند خواهد داشت (شکل ۴-۹).



```

1 from django.views import View
2 from .models import SaveUpdateNotification
3 from django.http import JsonResponse
4 from rest_framework.permissions import IsAuthenticated
5 from accounts.permissions import IsOwner
6
7
8 class ReceiveUpdateNotification(View):
9     permission_classes = [IsAuthenticated, IsOwner]
10
11     def get(self, request, *args, **kwargs):
12         SaveUpdateNotification(is_applied=False).save()
13         notification = SaveUpdateNotification.objects.filter(id=request.data['id'])
14         return JsonResponse({'Messages': notification})
15

```

شکل ۴-۹ : update.views.py

و حالا نتیجه را بررسی می کنیم، در ابتدا در شکل (۴-۱۰) نتیجه ارسال درخواست برای مشاهده بروز رسانی توسط کاربر مالک خانه را مشاهده می کنیم.

حال نتیجه ارسال درخواست برای مشاهده بروز رسانی توسط کاربر عضو خانه در شکل (۴-۱۱) قابل مشاهده است.

^۱ Member

۳۰

GET http://127.0.0.1:8000/update/receive-update-notification

Send Save

Params Authorization Headers (8) Body Pre-request Script Tests Settings Cookies Code

Headers 7 hidden

KEY	VALUE	DESCRIPTION
Authorization	Token eead4303c2f5519386771d920684404216d30671	
Key	Value	Description

Body Cookies (2) Headers (7) Test Results Status: 200 OK Time: 140 ms Size: 284 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "Message": "New Update Available version: V.1.0.3"
3 }
```

شکل ۴-۱۰ : update.views.py

POST http://127.0.0.1:8000/update/receive-update-notification

Send Save

Params Authorization Headers (10) Body Pre-request Script Tests Settings Cookies Code

Headers 9 hidden

KEY	VALUE	DESCRIPTION
Authorization	Token 60cf18267bcb289c012faabc25a05b2d66b8ae4b	
Key	Value	Description

Body Cookies (2) Headers (10) Test Results Status: 401 Unauthorized Time: 5 ms Size: 372 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "detail": "Authentication credentials were not provided."
3 }
```

شکل ۴-۱۱ : update.views.py

نتیجه گیری:

در این بخش با دریافت بازخورد از کاربران توانستیم بخش کنترلر خانه هوشمند سیستم خود را بروزرسانی کنیم و سیستم را بهبود ببخشیم و ضعف های سیستم خود را پس از انتشار برطرف کنیم. با اینکه ما تلاش کرده بودیم که سیستم را استاندارد طراحی و پیاده سازی کنیم، بازهم حفره ها و نکاتی از قلم افتاده بودند. در اینجا اهمیت قابلیت بروزرسانی سیستم نمایان شد.

فصل پنجم

نتیجه گیری و آینده نرم افزار

۵-۱ نتیجه گیری

ارائه یک سیستم یکپارچه و ایمن که بتواند نیازهای کاربران را برطرف کند و همزمان به درستی با اجزای مختلف خود ارتباط برقرار کند، امری بسیار دشوار و پیچیده است. همچنین با توجه به اینکه پیشرفت تکنولوژی ها در حوزه های IOT و Smart Home بسیار سریع است، حتی بی نقص ترین سیستم ها هم بعد از مدت کوتاهی کارایی خود را از دست می دهند. اما اگر سیستمی وجود داشته باشد که بتواند تمام الگوریتم ها و ساختار خود را با توجه به نیاز های روز، بروزرسانی کند و همگام با پیشرفت تکنولوژی قدم برداد آنگاه می تواند در چرخه مصرف باقی بماند.

هدف از انجام این پروژه پیاده سازی یک سیستم کنترلر خانه هوشمند بود که بتواند با گرفتن دستورات لازم خودش را ارتقا دهد.

۵-۲ آینده نرم افزار

با توجه به نیاز روزافزون انسان ها به تکنولوژی، خانه هوشمند جزئی از نیاز های اساسی انسان ها در سالهای آینده خواهد بود. با تلاش بر روی تمامی جوانب نرم افزار و ارائه یک سیستم کارآمد و با امکان پشتیبانی، می

توان زمینه تجاری سازی آن را مهیا کرد.

فصل ششم

مراجع و پیوست

۱-۶ پیوست

تمامی کد های پروژه و دیاگرام های مهم سیستم بر روی ریپازیتوری گیت هاب بنده به آدرس :
<https://github.com/horrhamid/iot-smart-home-api> قرار دارند.

مراجع

- [1] T. Chaurasia and P. K. Jain, "Enhanced Smart Home Automation System based on Internet of Things," 2019 Third International conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC), 2019, pp. 709-713, doi: 10.1109/I-SMAC47947.2019.9032685.
- [2] K. S. S. Javvaji, U. R. Nelakuditi and B. P. Dadi, "IoT Based Cost Effective Home Automation and Security System," 2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT), 2020, pp. 1-5, doi: 10.1109/ICCCNT49239.2020.9225557.
- [3] S. Anand, M. U. Pranavya, G. S. Vaibhavi, R. Apoorva and S. R. Shenoy, "Efficient Model For Automated Home Management System," 2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE), 2020, pp. 1-4, doi: 10.1109/ic-ETITE47903.2020.489.
- [4] K. Karimi and S. Krit, "Smart home-Smartphone Systems: Threats, Security Requirements and Open research Challenges," 2019 International Conference of Computer Science and Renewable Energies (ICCSRE), 2019, pp. 1-5, doi: 10.1109/ICCSRE.2019.8807756.
- [5] N. Auluck, A. Azim, A. Singh and M. A. Maruf, "Faster Fog Computing based Over-the-air Vehicular Updates: A Transfer Learning Approach," in IEEE Transactions on Services Computing, doi: 10.1109/TSC.2021.3099897.

- [6] S. G. Hong, N. S. Kim and T. Heo, "A smartphone connected software updating framework for IoT devices," 2015 International Symposium on Consumer Electronics (ISCE), 2015, pp. 1-2, doi: 10.1109/ISCE.2015.7177805.
- [7] G. Kaur and R. S. Batth, "Edge Computing: Classification, Applications, and Challenges," 2021 2nd International Conference on Intelligent Engineering and Management (ICIEM), 2021, pp. 254-259, doi: 10.1109/ICIEM51511.2021.9445331.
- [8] P. Singh, A. Kaur, G. S. Aujla, R. S. Batth and S. Kanhere, "DaaS: Dew Computing as a Service for Intelligent Intrusion Detection in Edge-of-Things Ecosystem," in IEEE Internet of Things Journal, vol. 8, no. 16, pp. 12569-12577, 15 Aug.15, 2021, doi: 10.1109/JIOT.2020.3029248.
- [9] W. Shi, J. Cao, Q. Zhang, Y. Li and L. Xu, "Edge Computing: Vision and Challenges," in IEEE Internet of Things Journal, vol. 3, no. 5, pp. 637-646, Oct. 2016, doi: 10.1109/JIOT.2016.2579198.