|  |  |  |  |
| --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | |  | **پروژه طراحی و پیاده‌سازی پلتفرم اینترنت اشیاء** |  | |
|  |
| عنوان:  **طراحی نرم­افزار- نیازمندی­های کارکردی** |
| ارائه دهنده:  کارگروه پلتفرم، گروه پژوهشی اینترنت اشیاء، دانشگاه صنعتی امیرکبیر |
| کد سند:  **ISRC-AUT-970121.0** |
| تاریخ انتشار:  21/01/1397 |

حق مالکيت سند

اين سند در مالکيت کارگروه پلتفرم، گروه پژوهشی اینترنت اشیاء، دانشگاه صنعتی امیرکبیر به نشاني تهران، خيابان حافظ، دانشگاه صنعتی امیرکبیر، دانشکده مهندسی کامپیوتر و فناوری اطلاعات بوده و شامل اطلاعات محرمانه و تجاري است. مالکيت اين سند را نمي‌توان بدون کسب اجازه کتبي از آزمایشگاه اینترنت اشیاء به شخص حقيقي يا حقوقي ديگري انتقال داد. هيچ‌کدام از اقلام اين سند را نمي‌توان بدون اجازه کتبي از آزمایشگاه اینترنت اشیاء مورد استفاده قرار داد، مجددا استفاده نمود، يا منتشر کرد.

اطلاعات سند

|  |  |
| --- | --- |
| نام پروژه: | پروژه طراحی و پیاده‌سازی پلتفرم اینترنت اشیاء |
| عنوان سند: | طراحی نرم­افزار-نیازمندی­های کارکردی |
| کد سند: | ISRC-AUT-970121.0 |
| نگارش: | 0/1 |
| نام تهیه کنندگان: | تیم فنی |
| تاریخ تهیه: | 21/01/1397 |
| نام بازبینی کننده: | مدیر پروژه |
| تاریخ آخرین بازبینی: | ۲۴/۰۱/۱۳۹۷ |
| نام ‌تائیدکننده: |  |
| تاریخ تائید: |  |
| وضعيت: | نسخه اولیه |
| نوع طبقه‌بندي سند: | محرمانه |

چکیده:

این سند سنگ بنای طراحی پروژه پلتفرم اینترنت اشیا می­باشد. بخش اصلی آن، معماری **کلان** پلتفرم اینترنت اشیا را پوشش می­دهد. معماری ارائه شده براساس موارد استخراج شده در فاز تحلیل نیازمندی­ها طراحی گردیده است. در طراحی این پلتفرم از معماری میکروسرویس استفاده شده است که طبق بررسی­های انجام شده، یکی از معماری­های مطلوب جهت توسعه پلتفرم­های اینترنت اشیا می­باشد. علاوه بر معماری کلان پلتفرم، طراحی بخش­های مختلف پلتفرم در این سند بررسی می‌گردد و ارتباطات بین اجزای پلتفرم مشخص می‌گردد.

دو مورد در خصوص این سند لازم به ذکر است:

* این سند، مستندسازی سامانه توسعه داده شده **نیست**. بنابراین به هیچ وجه به جزییات پیاده‌سازی پلتفرم نپرداخته است. بلکه هدف اصلی این مستند،‌ تشریح اجزای اصلی پلتفرم و ارتباطات آنها با یکدیگر است.
* این نسخه از سند، طراحی نهایی پلتفرم **نیست**. با توجه به نظرات کارفرمای محترم، همچنین تست‌هایی عملیاتی سامانه، امکان تغییرات جزیی در این معماری وجود دارد.

**واژه‌های کلیدی:**

پلتفرم اینترنت اشیاء، طراحی معماری، میکروسرویس

فهرست مطالب

[چکیده 3](#_Toc514829339)

[1- مقدمه 9](#_Toc514829340)

[2- معماري كلان پلتفرم 10](#_Toc514829341)

[2-1- مقدمه 10](#_Toc514829342)

[2-2- معماري ميكروسرويس 10](#_Toc514829343)

[2-3- معماري پلتفرم مبتني بر ميكروسرويس 11](#_Toc514829344)

[3- مولفه­ها و فرآیندهای پلتفرم اشیا 14](#_Toc514829345)

[3-1- مقدمه 14](#_Toc514829346)

[3-2- مولفه API-Server 14](#_Toc514829347)

[3-3- مولفه UI Server و Web UI 15](#_Toc514829348)

[3-4- مولفه Project Manager 17](#_Toc514829349)

[3-5- مولفه Up-link 18](#_Toc514829350)

[3-6- مولفه Down-link 19](#_Toc514829351)

[3-7- مولفه Data Manager 20](#_Toc514829352)

[3-8- مولفه Orchestrator 21](#_Toc514829353)

[4- پروتكل‌هاي ارتباطي با اشياء 23](#_Toc514829354)

[4-1- مقدمه 23](#_Toc514829355)

[4-2- پروتكل LoRaWAN 23](#_Toc514829356)

[4-3- پروتكل LAN 26](#_Toc514829357)

[4-4- واحد‌های مرتبط به انتخاب پروتکل ارتباطی با اشیاء 27](#_Toc514829358)

[5- پروتكل ارتباطي با برنامه‌هاي كاربردي 28](#_Toc514829359)

[5-1- مقدمه 28](#_Toc514829360)

[5-2- كليات پروتكل 28](#_Toc514829361)

[5-3- جزييات API مربوط به برنامه­های کاربردی 28](#_Toc514829362)

[6- پروتكل ارتباطي با ابزارهاي تحليل داده 30](#_Toc514829363)

[6-1- مقدمه 30](#_Toc514829364)

[6-2- اتصال به ابزارهاي داده 30](#_Toc514829365)

[7- حسابرسي 31](#_Toc514829366)

[7-1- مقدمه 31](#_Toc514829367)

[7-2- سرویس­های ارائه شده در این بخش 31](#_Toc514829368)

[7-3- فرايند خريد و پرداخت 32](#_Toc514829369)

[8- پايگاه‌داده 33](#_Toc514829370)

[8-1- مقدمه 33](#_Toc514829371)

[8-2- خوشه بندی در پایگاه داده Mongo 33](#_Toc514829372)

[8-3- تکثیر پایگاه­داده 34](#_Toc514829373)

[8-4- معماری پیشنهادی 36](#_Toc514829374)

[8-5- انواع پایگاه داده­های پلتفرم 37](#_Toc514829375)

[8-5-1- پايگاه داده‌هاي خام 37](#_Toc514829376)

[8-5-2- پايگاه داده‌هاي پارس شده 37](#_Toc514829377)

[8-5-3- پايگاه داده‌هاي پلتفرم 37](#_Toc514829378)

[8-6- سایر موارد مرتبط به پایگاه داده­ها 38](#_Toc514829379)

[9- طراحی فرآیند مدیریت اشیا 39](#_Toc514829380)

[10- طراحی و mockupپنل کاربر 40](#_Toc514829381)

[11- طراحی و mockupپنل مدیریت 51](#_Toc514829382)

**فهرست اشکال**

[شکل ‏2‑1 معماری میکروسرویس 11](#_Toc514829533)

[شکل ‏2‑2. معماری سیستم پیشنهادی 12](#_Toc514829534)

[شکل ‏3‑1. فرآیند ثبت نام کاربر 16](#_Toc514829535)

[شکل ‏3‑2. فرآیند تعریف شی 16](#_Toc514829536)

[شکل ‏3‑3. سناریوی تعریف پروژه 18](#_Toc514829537)

[شکل ‏3‑4. سناریوی دریافت داده 19](#_Toc514829538)

[شکل ‏3‑5. سناریوی ارسال داده 20](#_Toc514829539)

[شکل ‏3‑6. سناریوی نمایش داده 21](#_Toc514829540)

[شکل ‏4‑1. معماری مدل ارتباطی مبتنی بر LoRaServer 24](#_Toc514829541)

[شکل ‏4‑2. ارتباط اشیا با پلتفرم از طریق Loraserver.io 25](#_Toc514829542)

[شکل ‏4‑3. معماری گذرگاه LAN جهت ارتباط با اشیا 26](#_Toc514829543)

[شکل ‏5‑1. سناریوی ثبت نام کاربر از طریق اپلیکیشن اندروید 29](#_Toc514829544)

[شکل ‏8‑1. معمای خوشه­ای 34](#_Toc514829545)

[شکل ‏8‑2. معماری تکثیر پایگاه داده 35](#_Toc514829546)

[شکل ‏8‑3. معماری تکثیر با استفاده از خوشه بندی 36](#_Toc514829547)

[شکل ‏8‑4. معماری خوشه بندی با استفاده از تکثیر 37](#_Toc514829548)

[شکل ‏10‑1. صفحه ورود کاربران 40](#_Toc514829549)

[شکل ‏10‑2. صفحه ثبتنام کاربران 40](#_Toc514829550)

[شکل ‏10‑3. صفحه ویرایش اطلاعات حساب کاربری 41](#_Toc514829551)

[شکل ‏10‑4. داشبورد سامانه 41](#_Toc514829552)

[شکل ‏10‑5. صفحه افزودن افزونه جدید به داشبورد 42](#_Toc514829553)

[شکل ‏10‑6. نمایش لیست پروژه ها 42](#_Toc514829554)

[شکل ‏10‑7. صفحه تعریف پروژه جدید 43](#_Toc514829555)

[شکل ‏10‑8. صفحه مدیریت پروژه (1) 43](#_Toc514829556)

[شکل ‏10‑9. صفحه مدیریت پروژه (2) 43](#_Toc514829557)

[شکل ‏10‑10. صفحه مدیریت پروژه (3) 44](#_Toc514829558)

[شکل ‏10‑11. صفحه مدیریت پروژه (4) 44](#_Toc514829559)

[شکل ‏10‑12. صفحه افزودن شی جدید 45](#_Toc514829560)

[شکل ‏10‑13. صفحه افزودن دسته­ا­ی اشیا 45](#_Toc514829561)

[شکل ‏10‑14. صفحه فعال سازی اشیا 46](#_Toc514829562)

[شکل ‏10‑15. صفحه ارسال کدک 46](#_Toc514829563)

[شکل ‏10‑16. صفحه ارسال داده به شی 47](#_Toc514829564)

[شکل ‏10‑17. صفحه نمایش پروژه (1) 47](#_Toc514829565)

[شکل ‏10‑18. صفحه نمایش پروژه (2) 48](#_Toc514829566)

[شکل ‏10‑19. صفحه لیست اشیا 48](#_Toc514829567)

[شکل ‏10‑20. صفحه لیست گذرگاه­ها 48](#_Toc514829568)

[شکل ‏10‑21. صفحه افزودن گذرگاه 49](#_Toc514829569)

[شکل ‏10‑22. صفحه لیست پروفایل اشیا 49](#_Toc514829570)

[شکل ‏10‑23. صفحه ساخت پروفایل اشیا 50](#_Toc514829571)

[شکل ‏11‑1. صفحه لیست کاربران سامانه 51](#_Toc514829572)

[شکل ‏11‑2. صفحه اطلاعات کاربر (1) 52](#_Toc514829573)

[شکل ‏11‑3. صفحه اطلاعات کاربر (2) 52](#_Toc514829574)

[شکل ‏11‑4. صفحه لیست بسته­های موجود در سامانه 52](#_Toc514829575)

[شکل ‏11‑5. صفحه افزودن بسته جدید 53](#_Toc514829576)

[شکل ‏11‑6. صفحه مدیریت درگاه­های بانکی 53](#_Toc514829577)

**فهرست جداول:**

[جدول ‏2‑1. واحد‌ها به همراه شرح وظایف 13](#_Toc511313579)

# مقدمه

فاز طراحي يكي از فازهاي اصلي در توسعه نرم‌افزار است. اسناد طراحی در حقیقت سنگ­بنای مراحل توسعه، تست و ارزیابی سیستم می­باشند. در فصل دوم این مستند معماری کلان پلتفرم به همراه کلیت مولفه‌های آن مورد بررسی قرار می‌گیرد. فصل سوم اجزای داخلی پلتفرم به همراه فرآیندها یا سناریوهای آن را ارائه می­دهد. در ادامه در فصل چهارم پروتکل­های ارتباطی با اشیا و نحوه ارتباط با Gatewayها تشریح می‌گردد. فصل پنجم نحوه ارتباط برنامه­های کاربردی با پلتفرم را شرح می­دهد. سپس در فصل ششم پروتکل­ها و APIهای طراحی شده جهت ارتباط با ابزارهای تحلیل داده مورد بررسی قرار می‌گیرد. فصل هفتم جزییات بیشتر از نحوه حسابرسی و شارژینگ را مطرح می‌کند. در فصل هشتم، طراحی پایگاه داده پلتفرم و انواع معماری­های موجود را بررسی و روش پیشنهادی ارائه می‌گردد. فصل نهم به مدیریت اشیا اشاره­ای دارد. در نهایت فصل­های آخر به طراحی صفحات کاربری و مدیریت پلتفرم اختصاص پیدا کرده است.

# معماري كلان پلتفرم

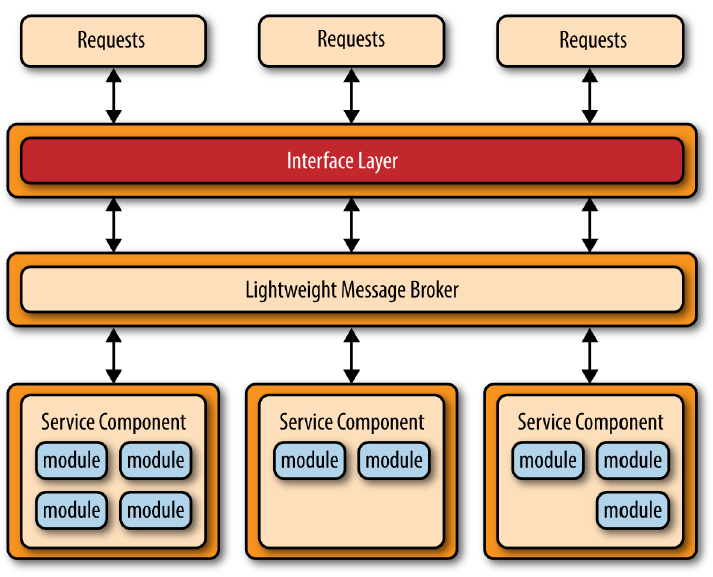
## مقدمه

يكي از فاكتورهايي كه نقش کلیدی در موفقيت يك محصول نرم‌افزاري دارد، معماري آن است. طبیعتا طراحی پلتفرم هم از این قضیه مستثنی نیست و تامین همه نیازمندی‌ها بخصوص نیازمندی‌های غیرکارکردی شامل دسترس­پذیری و مقیاس­پذیری در گرو استفاده از یک معماری مناسب برای سامانه است. معماری سامانه، اجزای داخلی سامانه و نحوه تعاملات آن­ها را در راستای برآورد انتظارات کارفرما مشخص می‌کند. همان­طور که در پروپوزال پروژه نیز شده است معماری ارائه شده بر اساس میکروسرویس می­باشد. در ابتدا معماری میکروسرویس و مزایای آن تشریح شده است. سپس معماری پلتفرم پیشنهادی با جزئیات کامل ارائه و مورد بررسی قرار می‌گیرد.

## معماري ميكروسرويس

در حوزه معماری سيستم‌هاي نرم‌افزاري، الگو‌های معماری متعدد و دسته‌بندی­های متفاوتی وجود دارد. مهمترین معماری­هایی که در حال حاضر استفاده می‌شوند شامل معماری­های لایه­ای[[1]](#footnote-1) ( مانند MVC[[2]](#footnote-2)) Layered، مبتنی بر رویداد[[3]](#footnote-3)، میکرو کرنل، میکروسرویس و Space-Based می­باشد.

معماری میکروسرویس برخلاف روشهای سنتی سامانه‌های یکپارچه، یک سیستم توزیع شده متشکل از چندین سرویس است که هر سرویس به صورت مجزا پیاده‌سازی شده و سرویس مربوطه را از طریق API در اختیار سایر سرویس‌ها و مشتریان قرار می‌دهد. این معماری در حقیقت سرعت و کارایی بالایی را در توسعه و تغییر سیستم فراهم می‌کند و معماری کلاسیک مبتنی بر سرویس را بهبود می­دهد. شمای کلی این معماری در شکل ‏2‑1 نشان داده شده است.



شکل ‏2‑1 معماری میکروسرویس­

در معماری میکروسرویس، کارکردهای سامانه در قالب سرویس‌هایی پیاده‌سازی می‌شوند که هر سرویس یک واحد مستقل است و همه موارد مورد نیاز جهت اجرای خود را دارد. علاوه بر آن می­تواند در هر کجای شبکه استقرار پیدا کند. برای توزیع درخواست‌های داده شده به سامانه و همچنین ارتباطات بین سرویس‌ها عموما از یک Message Broker در این معماری استفاده می‌شود. استفاده از این معماری برای پیاده‌سازی پلتفرم بومی‌ عام منظوره پیشنهادی دارای چندین مزیت است. اول اینکه اکثر نیازمندی‌های کارکردی سامانه می‌توانند به صورت سرویس‌های مجزا در قالب این معماری پیاده‌سازی شوند که فرایند نگه­داری و قابلیت اطمینان آن را راحت­تر می‌کند چرا که به راحتی می­توان تشخیص داد کدام واحد دچار خطا شده است. این معماری به دلیل توزیع‌شدگی به صورت ذاتی مقیاس‌پذیر بوده و امکان اضافه کردن قابلیت‌های غیرکارکردی به این معماری وجود دارد. همچنین در طراحی­هایی که کامپوننت­های مختلف وجود دارند و یا توسعه آن­ها برون­سپاری می‌گردد امکان توسعه آن­ها به زبان­های مختلف و در تیم­های جداگانه به راحتی از طریق APIهای استاندارد فراهم می‌گردد.

## معماري پلتفرم مبتني بر ميكروسرويس

شمای کلی معماری ارائه شده برای پلتفرم اینترنت اشیا در شکل ‏2‑2 نشان داده شده است. در این شکل اجزای مختلف پلتفرم، ارتباطات داخلی و ارتباطات خارجی آن مشخص شده است.



شکل ‏2‑2. معماری سیستم پیشنهادی

در جدول ‏2‑1 واحدهای متفاوت این معماری به شکل خلاصه معرفی شده‌اند و مشخص شده است که هر واحد از طریق کدام زبان برنامه­نویسی پیاده‌سازی شده و یا برای راه‌اندازی آن از چه تکنولوژی و ابزاری استفاده شده است.

جدول ‏2‑1. واحد‌ها به همراه شرح وظایف

| **نام واحد** | **وظیفه** | **زبان / تکنولوژی** |
| --- | --- | --- |
| Data Analysis | این واحد برای اتصال به ابزارهای تحلیل داده مورد استفاده قرار می‌گیرد. | Java/Apache Spark |
| Web UI | این واحد واسط‌های کاربری گرافیکی[[4]](#footnote-4) وب سایت پلتفرم را پیاده می‌کند(front-end) . | HTML/CSS/JS |
| UI server | این واحد وظیفه پیاده‌سازی توابع واسط­های کاربری و ارتباط با API Server جهت فراخوانی واسط­های سمت سرور را بر عهده دارد (front-end) . | ReactJS |
| API-Server | واسط­ها و توابع back-end سمت سرور توسط این واحد پیاده‌سازی می‌گردد. واسط­های فراهم شده هم توسط وب و هم توسط موبایل قابل استفاده و فراخوانی است. | Laravel (PHP Framework) |
| Project Manager | وظیفه مدیریت پروژه‌های ایجاد شده توسط کاربر را به عهده دارد و به هر کاربر یک container اختصاص می­دهد. | Go |
| Data Manager | این واحد وظیفه پیاده‌سازی توابع مربوط به پرس وجو­های بر روی داده‌ها را بر عهده دارد. | Go |
| Containers | مجموعه­ای از Container‌ها را شامل می‌گردد که هر Container وظیفه مدیریت سناریوها و کدک‌های آن کاربر را بر عهده دارد. | Python3/Go |
| Down-Link | این واحد وظیفه انتقال اطلاعات از پلتفرم به اشیاء را بر عهده دارد. | Go |
| Up-Link | این واحد وظیفه انتقال اطلاعات از اشیاء به پلتفرم را به عهده دارد. | Go |
| Connectivity | این واحد وظیفه مدیریت اتصال به gateway‌های پروتکلهای ارتباطی با اشیا را به عهده دارد. | Go |
| LoRaWAN Gateway | Gateway ارتباط با اشیا را از طریق LoRa Server فراهم می‌کند. | Go |
| LAN Gateway | Gateway ارتباط با اشیا را از طریق LAN Server فراهم می‌کند. | Python3 |
| Orchestrator | این واحد وظیفه مدیریت تمامی‌واحد‌ها را بر عهده دارد. | Kubernetes/ Rancher |
| Database | این واحد وظیفه مدیریت داده‌های پلتفرم را به عهده دارد. | MongoDB |
| Redis | به عنوان یک محل ذخیره سازی داده به صورت in memory، نقش یک واسط را بین پایگاه داده و واحد Up-Link ایفا می‌کند. | Redis |
| Android App | یک برنامه کاربردی موبایل است که امکان اتصال مستقیم و استفاده از توابع ارائه شده توسطServer API را دارد. | -- |

در ادامه این مستند عملکرد واحد­ها به تفصیل مورد بحث و بررسی قرار خواهد گرفت.

# مولفه­ها و فرآیندهای پلتفرم اشیا

## مقدمه

در این بخش ابتدا مولفه­های درون پلتفرم اینترنت اشیا تشریح می‌گردد. پس از معرفی سرویس­ها، فرآیندهای پایه­ای که در راستای مدیریت اشیا و پلتفرم تعریف شده است مورد بررسی قرار می‌گیرد.

## مولفه API-Server

این مولفه به نوعی در قلب پلتفرم قرار دارد و وظیفه پیاده­سازی توابع Backend وب را برعهده دارد. تمام توابع مرتبط به سمت کاربر و مدیریت پلتفرم در این واحد پیاده­سازی شده‌اند. این مولفه از یک سمت با واحدهای Project Manager، Data Manager، Gateway Manager و در برخی موارد با Gatewayهای ارتباطی با اشیا مانند LoRaWAN در ارتباط است. در سمت دیگر نیز با واسط‌های گرافیکی کاربران نهایی و مدیر پلتفرم ( Front-End) و برنامه­های کاربردی (مانند کاربرد سمت اندروید) در ارتباط است. این مولفه در این ارتباط، سرویس­های لازم را فراهم می‌کند و در صورت لزوم برخی از سرویس­ها را دریافت می‌کند. توابع ارائه شده توسط این بخش به دو دسته اصلی زیر تقسیم شده‌اند:

1. توابع کاربران نهایی که برخی از مهمترین آنها عبارتند از

* احراز هویت شامل ورود به سیستم، ثبت نام و ویرایش کاربران
* مدیریت پروژه‌ها شامل ساخت پروژه، و به روزرسانی اطلاعات پروژه
* مدیریت اشیاء شامل ساخت شی، حذف شی، دریافت و ارسال داده، نوع اینترفیس ارتباطی (LAN و LoRaWAN)
* مدیریت پروفایل اشیاء شامل ساخت پروفایل، و بازیابی پروفایل
* مدیریت کدک شامل ساخت کدک، و بازیابی کدک
* مدیریت سناریو‌ها شامل ساخت سناریو، به روزرسانی سناریو، بازیابی سناریو، و صحت سنجی سناریو
* مدیریت Gatewayها
* حسابرسی و پرداخت کاربران
* داشبورد کاربر
* ...

1. توابع مدیریت پلتفرم که برخی از مهمترین آنها عبارتند از

* مدیریت کاربران و اطلاعات آن­ها
* مدیریت اجازه دسترسی‌ها و نقش ها
* مدیریت درگاه­های ارتباطی
* مدیریت کانتینرهای (Containers) پروژه
* مدیریت درگاه­های ارتباطی و وضعیت آن­ها
* گزارش­گیری
* مدیریت پرداخت
* و...

همه توابع ذکر شده، قابلیت ارائه سرویس به واسط­های گرافیکی تحت وب (Web UI , UI Server) و کاربردهای سمت اندروید (android App) را دارند.

## مولفه UI Server و Web UI

این مولفه­ها همان­طور که قبلا گفته شد بخش Front-end پلتفرم را تشکیل می­دهند. که با توجه به جدا بودن Back-end، در نتیجه APIهای فراهم شده برای این بخش توسط سایر برنامه­های کاربردی نیز قابل استفاده است. واحد Web UI ظاهر گرافیکی سایت و UI Server نیز توابع مربوط به واسط گرافیکی سایت و فراخوانی APIهای مربوط به Back-end را برعهده دارد.

برای نمونه دو فرآیند اصلی ثبت نام کاربر و تعریف شی در شکل ‏3‑1 و شکل ‏3‑2 نمایش داده شده است. در شکل­ها واحد‌های درگیر در فرآیند نیز با رنگ آبی نشان داده شده­اند. در سناریوی ثبت نام کاربر (شکل ‏3‑1) ابتدا کاربر از واسط کاربری خود گزینه ثبت نام را انتخاب می‌کند. پس از انتخاب کاربر توابع مورد نیاز برای ثبت اطلاعات از API-Server فراخوانی می‌شود و اطلاعات کاربر در پایگاه داده مربوط به کاربران ثبت می‌گردد. در سناریوی تعریف شی یا Activation از نوع اشیا Lora (شکل ‏3‑2)، کاربر از طریق واسط کاربری گزینه مرتبط به فعال­سازی شی را انتخاب می‌کند. پس از انتخاب کاربر، توابع مرتبط از بخش API-Server فراخوانی می‌شود و اطلاعات شی مورد نظر پس از طی روال تایید آن در سمت LoRaWAN Gateway، در پایگاه داده ثبت می‌گردد. نیازمندی­ها و پروسه ثبت نام اشیا به دو روش OTAA و ABP قبلا در سند تحلیل نیازمندی­های کارکردی اینترنت اشیا -سرویس­های پایه (IoT-RA-BS-v1.16) شرح داده شده است.



شکل ‏3‑1. فرآیند ثبت نام کاربر



شکل ‏3‑2. فرآیند تعریف شی

## مولفه Project Manager

این واحد وظیفه مدیریت پروژه­های تعریف شده کاربران را دارد که شامل ایجاد پروژه، فعال یا غیر فعال‌سازی پروژه می‌باشد. در

شکل ‏3‑3 سناریوی تعریف پروژه نشان داده شده است و واحد‌های مرتبط با رنگ آبی مشخص شده‌اند. در این سناریو کاربر از طریق منوی تعریف پروژه که در واسط کاربری فراهم شده است یک پروژه ایجاد می‌کند. پس از دریافت درخواست کاربر از طریق واسط کاربری، توابع مرتبط در سمت سرور فراخوانی می‌شود که این امر باعث فعال شدن واحد‌های Project Manager، Runner و Database می‌شود و در نهایت پروژه تعریف می‌گردد. واحد Runners مجموعه­ای از کانتیرنها می­باشد و در حقیقت سناریو و کدک‌های کاربران را مدیریت می‌کند و به ازای هر کاربر یک کانتینتر تعریف می‌گردد که کدهای کاربر در آن اجرا می‌گردد. در این صورت خرابی کدک یا سناریو یک کاربر تاثیری در کدک­ها و سناریوهای سایر کاربران ندارد. با توجه به اینکه هر کاربر یک Container برای خود دارد در نتیجه حجم مورد استفاده آن به ازای هر کاربر، بر اساس تعداد کاربران و اندازه دیتاسنتر در مراحل پایانی پروژه بررسی شده و ارائه می‌گردد. البته وضعیت این Runnerها نیز در پلتفرم مدیریتی در اختیار مدیر پلتفرم قرار خواهد گرفت. اطلاعات ایجاد شده نیز در پایگاه داده Platform، جهت استفاده­های بعدی ذخیره می‌گردد.



شکل ‏3‑3. سناریوی تعریف پروژه

## مولفه Up-link

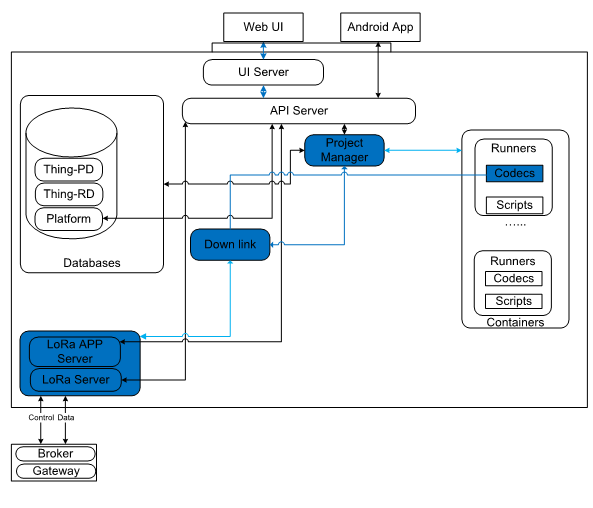
این مولفه وظیفه انتقال اطلاعات از شی به پلتفرم را به عهده دارد. در این بخش داده­ها به دو صورت پارس شده و خام در پایگاه داده قرار می‌گیرند. در شکل ‏3‑4 سناریوی دریافت داده نشان داده شده است و واحد‌های مرتبط با رنگ آبی نشان داده شده‌اند. اطلاعات خام دریافتی از سمت LoRaGateway از طریق uplink به سمت پایگاه داده ارسال می‌گردد. برای افزایش سرعت نیز از Redis به عنوان یک منبع ذخیره سازی ساختار داده in-memory استفاده شده است که توانایی پاسخ گویی به تعدادی زیادی از داده­های دریافت شده در کسری از میلی ثانیه را داراست در نتیجه امکان Lost داده­های دریافتی از بین می­رود. برای داده­های پارس شده نیز، از کدکی که برای اشیا از سمت کاربران تعریف شده است استفاده می‌گردد و اطلاعات اشیا در ابتدا پردازش و سپس جهت ذخیره­سازی به پایگاه داده ارسال می‌گردد. در این حالت، واحد Up-Link اطلاعات مربوط به کدک کاربران را از Runner یا کانتینر مربوطه دریافت می‌کند. اینکه کدام Runner، کدک مربوطه را داراست از طریق Project Manager مشخص می‌شود. سپس اطلاعات درون شی را استخراج می‌کند و در پایگاه داده ذخیره می‌کند.



شکل ‏3‑4. سناریوی دریافت داده

## مولفه Down-link

این بخش وظیفه انتقال داده از پلتفرم به اشیاء را به عهده دارد. روال آن در شکل ‏3‑5 آمده است. واحد Down-link پس از دریافت اطلاعات از سمت کاربر، از طریق Project Manager، کدک­های مرتبط شامل encoding مربوطه به آن شی را پیدا کرده، داده مد نظر را با استفاده از encoder مربوطه کد کرد و تبدیل به یک bit stream می‌کند که می‌تواند به شی مربوطه ارسال گردد. با توجه به شناسه شی، داده شده، اطلاعات از طریق Downlink به سمت LoRa Gateway ارسال می‌گردد.

****

شکل ‏3‑5. سناریوی ارسال داده

## مولفه Data Manager

این واحد وظیفه پیاده‌سازی توابع مرتبط به عملیات پس و جوی (query) کاربران بر روی داده‌های پایگاه داده پارس شده را بر عهده دارد. با توجه به اینکه درخواست­های متعددی از واسط­های گرافیکی کاربر برای نمایش داده به صورت نمودار و جدولی و جزئیات نمایش آن­ها شامل Aliasing ارسال می‌گردد مدیریت و نحوه پاسخگویی به این درخواست­ها در این مولفه قرار داده شده است.

در شکل ‏3‑6 فرآیند نمایش داده ترسیم شده است که واحد‌های مرتبط با رنگ آبی نشان داده شده‌اند. در این سناریو ابتدا کاربر از طریق واسط کاربری گزینه نمایش داده‌های شی را انتخاب می‌کند. انتخاب کاربر در سمت سرور باعث فراخوانی توابع مرتبط می‌شود. در ادامه توابع بازیابی از واحد Data Manager صدا زده می‌شود و در نهایت اطلاعات مورد نظر بازیابی می‌گردد و به کاربر نمایش داده می‌شود.



شکل ‏3‑6. سناریوی نمایش داده

## مولفه Orchestrator

همان­طور که در شکل شکل ‏2‑2 نیز نشان داده شد، این واحد با تمام واحدهای دیگر در ارتباط است و به نوع مدیریت این واحد­ها را برعهده دارد. در اینجا از kubernetes که یک بستر برای مدیریت و هماهنگ‌سازی کانتینرها می‌باشد استفاده می‌گردد. این بستر اپلیکشن‌های شما را که ممکن است از یک یا تعدادی کانتینر تشکیل شده باشد اجرا و مدیریت می‌کند و نیازمندی‌های ارتباطی آن‌ها را فراهم می‌آورد. در این بین کاربر می‌تواند هر یک از کانتینرهایی دلخواه خود را گسترش داده و تعداد نمونه‌های آن را زیاد یا کم کند. کانتینرها ممکن است نیاز به حافظه‌هایی جهت ذخیره‌ی دائمی‌داده‌ها داشته باشند که به آن‌ها storage گفته می‌شود، کوبرنتیس storage هر یک از کانتینر‌ها را نیز مدیریت می‌کند.

کوبرنتیس سرویس‌های Discovery نیز برای اپلیکشن‌ها فراهم می‌آورد که از طریق آن می‌توانند با یکدیگر ارتباط برقرار کنند. در صورت خرابی یا fail شدن یک کانتینر، کوربرنتیس این موضوع را تشخیص داده و می‌تواند آن کانتینر را باز اجرا کند.

کوبرنتیس برای پشتیبانی از کانتینرها از موتورهای Docker و RKT پشتیبانی می‌کند که پشتیبانی از موتور RKT هنوز در فاز اولیه قرار دارد.

# پروتكل‌هاي ارتباطي با اشياء

## مقدمه

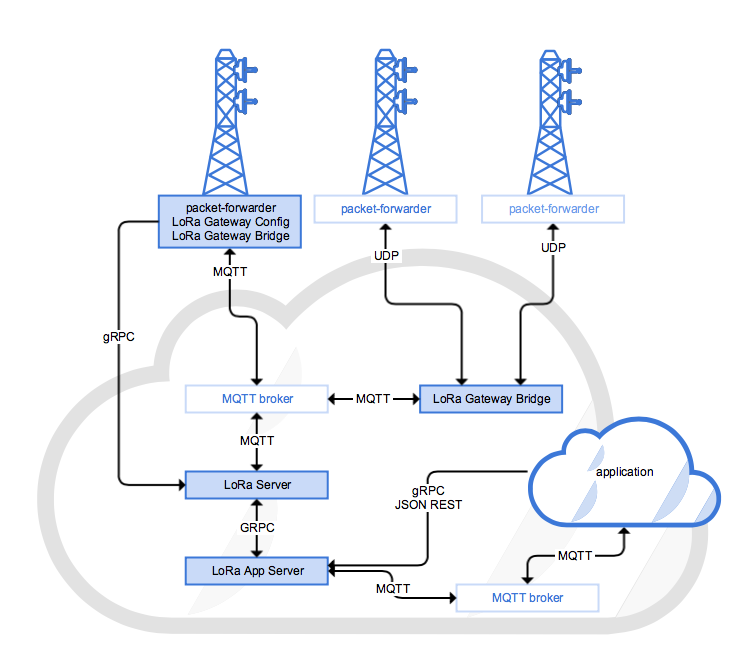
ارتباط با اشیاء یکی از نیازمندی‌های اصلی پلتفرم پیشنهادی است. پروتکل‌های متفاوتی برای این امر وجود دارد. در معماری ارائه شده چهار پروتکل زیر قرار دارند :

1. پروتكل LoRaWAN
2. پروتكل ارتباط IP از طريق LAN
3. پروتكل خاص منظوره كارفرما
4. پروتكل NB-IoT

با توجه به اولویت بندی انجام شده و نیازمندی­های کارفرما در حال حاضر دو مدل ارتباطی یکی بر مبنای LoRaWAN و دیگری LAN در مرحله طراحی و پیاده­سازی قرار گرفته­اند. تشریح این پروتکل­ها در ادامه آمده است.

## پروتكل LoRaWAN

در این بخش برای ارتباط با اشیا LoRa از پروژه متن باز LoraServer.io استفاده شد. این پروژه مجموعه ای از برنامه‌های کاربردی متن باز است که امکان ارتباط بین درگاه‌های دریافت کننده پیام از اشیا LoRa و ابزار‌ها یا پلتفرم­های مدیریتی را در اینترنت اشیاء فراهم می‌کند. LoRaWAN یک پروتکل در سطح لایه Mac است که ارتباطات بیسیم را فراهم می‌کند. این پروتکل ارتباط بین اجزا را در توان پایین ولی با برد بالا فراهم می‌کند. در شکل ‏4‑1 معماری پروژه متن باز LoRaServer نمایش داده شده است.



شکل ‏4‑1. معماری مدل ارتباطی مبتنی بر LoRaServer

معماری ارائه شده توسط LoRaServer از مولفه‌های زیر تشکیل شده است.

* LoRa nodes: اشیایی که داده‌ها را از طریق درگاه ارتباطی به شبکه ارسال میکنند. این دستگاه‌ها می‌توانند برای مثال حسگرهای اندازه گیری کیفیت هوا، دما و رطوبت باشند.
* LoRa Gateway: این مولفه وظیفه دریافت داده از اشیا را بر عهده دارد. درگاه‌ها معمولا از نرم‌افزار‌های forwarder بسته‌ها استفاده میکنند. در برخی موارد همان­طور که در شکل نیز نشان داده شده است امکان دارد این مولفه LoRa Gateway Bridge را نیز در خود داشته باشد که ارتباطات را از طریق MQTT به LoRa Server ارسال می‌کند.
* LoRa Gateway Bridge: این مولفه مسئول برقراری ارتباط با درگاه ارتباطی است. این مولفه پکت­های UDP دریافتی از Gateway را به فرمت JSON بر روی MQTT می­برد. این مولفه مزایای مانند تامین امنیت، آسان سازی اشکال یابی و ... را داراست.
* LoRa Server: مسئول هماهنگی تمام مولفه‌ها در شبکه است. این واحد از اشیا فعال در شبکه مطلع است. هنگامی‌که یک شی جدید به شبکه متصل می‌شود این واحد از App server در مورد وضعیت شی اطلاعات می‌گیرد که در صورت لزوم آن دستگاه را در سیستم ثبت کند. موارد دیگر از جمله مدیریت عمل de-duplicat (حذف داده­های تکراری) داده­های دریافتی از سمت چند Gateway، اعتبار سنجی این داده­ها و ارسال آن به سمت Application-Server و در صورت لزوم دریافت پیام برگشتی از آن را برعهده دارد.
* LoRa App Server: این واحد مسئول پیاده‌سازی کاربرد هاست به شکلی که با LoRa Server هماهنگ باشند. قابلیت­های مدیریت اشیا به ازای هر کاربرد و سازمان و مدیریت گذرگاه به ازای هر سازمان را در اختیار قرار میدهد. سایر قابلیت­های مربوط به مدیریت کاربران و ارتباط آن با کاربردها را نیز فراهم می‌کند. ارتباط با کاربرد‌ها از طریق JSON بر روی MQTT فراهم خواهد شد. API مناسبی جهت توسعه کاربرد‌ها در پروژه LoRa تدارک دیده شده است.
* Application: جایگاه کاربردها را نمایش می­دهد. هر کاربرد از طریق یک عنوان و با استفاده از MQTT داده‌ها را از دستگاه‌ها جمع آوری می‌کند. برنامه کاربردی حتی قادر به ارسال داده از طریق MQTT نیز هست. در صورت لزوم نیز توانایی ارتباط API از gRPC یا JSON Rest را دارد.

بخش‌های ذکر شده از پروژه LoRa شامل Lora Gateway Bridge، MQTT Brocker، LoRa Server و LoRa APP Server در پلتفرم اینترنت اشیا نیز مورد استفاده قرار گرفته است. در حقیقت در حال حاضر بخشی از پلتفرم اینترنت اشیا می­باشند. در ادامه جایگاه پلتفرم ارائه شده در تعامل با مولفه‌های ذکر شده مورد بررسی قرار گرفته است.



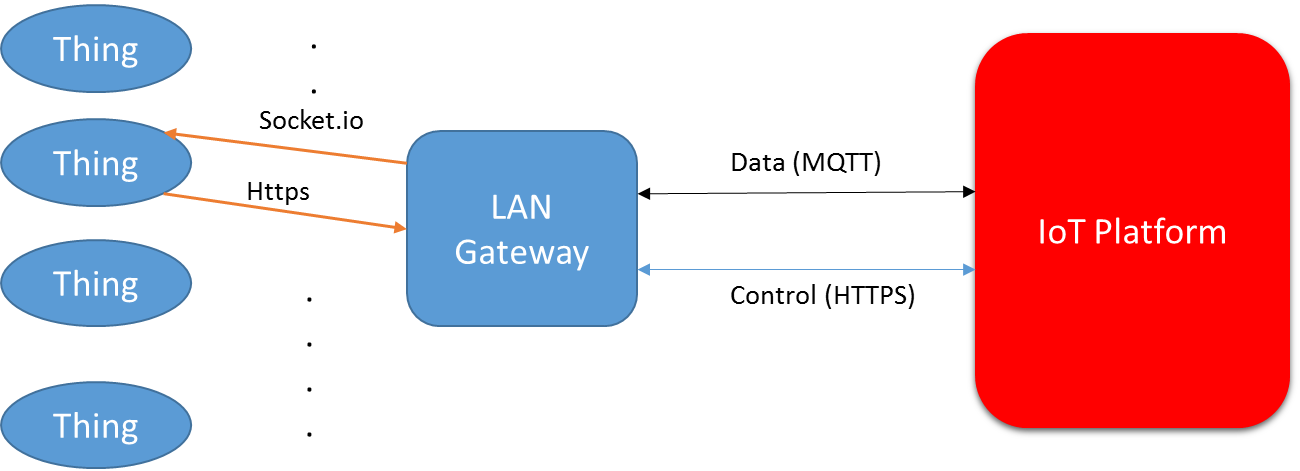
شکل ‏4‑2. ارتباط اشیا با پلتفرم از طریق Loraserver.io

همان­طور که در شکل ‏4‑2 نشان داده شده است بخش LoRa App Server مسئول تعامل با پلتفرم است. در حال حاضر در معماری ارائه شده MQTT برای انتقال داده و ارتباط مبتنی بر HTTP به عنوان کنترلی استفاده می‌گردد.

اطلاعات مربوط به تنظیمات کلی پروتکل، تنظیمات شبکه، پروتکلی کنترلی و داده مابین پلتفرم و سرور LoRa و سایر موارد در مستند (IoT-RA-BS-v1.16) ذكر شده است.

## پروتكل LAN

دیگر پروتکل ارتباطی با اشیا پروتکل‌ LAN (خانواده IEEE 802 و مشخصاIEEE 802.3، IEEE 802.11 و ...) می­باشد. شمای کلی ارتباطات شکل ‏4‑3 نمایش داده شده است.



شکل ‏4‑3. معماری گذرگاه LAN جهت ارتباط با اشیا

لازم به ذکر است که این پروتکل علاوه بر پلتفرم، بر روی اشیاء نیز باید پیاده‌سازی گردد، بنابراین استفاده از پروتکل‌های استاندارد موجود برای این منظور (در عمل) الزامی ‌است. ساختار کلی طراحی در سه بخش زیر آمده است:

1. اشیاء برای اتصال به Lan Gateway در آدرس ‘/login’ با استفاده از token وارد می‌شوند. برای ارسال داده از اشیاء به پلتفرم، یک درخواست POST با استفاده از HTTPS به Lan Gateway ارسال می‌شود. سپس این داده‌ها برای ارسال به پلتفرم روی کانال MQTT گذاشته می‌شوند. درخواست POST ارسال شده از شیء به Lan Gateway به آدرس ‘/data’ و با فرمت JSON است که شامل token و data است. این مرحله قابلیت اتصال اشیا به صورت امن را فراهم می‌کند.
2. برای ارسال داده از پلتفرم به اشیاء، داده‌ها از پلتفرم با استفاده از کانال MQTT به Lan Gateway ارسال می‌شوند. Lan Gateway که در حال گوش دادن به این کانال است با دریافت داده با استفاده از Socket.io داده‌ها را برای شیء ارسال می‌کند. Lan Gateway عنوان و داده‌ها را به صورت JSON دریافت می‌کند و به اشیاء ارسال می‌کند. این قابلیت نیز برآورده کردن امکان ارسال داده از اشیا به پلتفرم را بر اساس نیازمندی­ها تامین ­کند.
3. بر روی Lan Gateway یک API با استفاده از HTTP وجود دارد تا دستورات کنترلی به آن ارسال گردد. یکی از دستورات کنترلی، دستور اضافه کردن اطلاعات یک شیء به پایگاه داده‌ی بر روی Lan Gateway است. این API دارای آدرس ‘/control’ و نوع درخواست POST است که حاوی type (نوع درخواست کنترلی) و data (داده‌های مربوط به درخواست) است. با ارسال این درخواست توکن مربوط به شی جدید با استفاده از JWT تولید و برگردانده می‌شود. این قابلیت نیز امکان ارسال داده کنترلی به اشیا از طریق پلتفرم را فراهم می‌کند.
   1. واحد‌های مرتبط به انتخاب پروتکل ارتباطی با اشیاء

همانگونه که در شکل ‏2‑2 نشان داده شده است برای ارتباط با درگاه‌ها واحد connectivity در پلتفرم طراحی شده است که وظیفه مدیریت اتصال به درگاه‌ها را دارد. به عنوان مثال این واحد با استفاده از اطلاعاتی که از درگاه‌ها دارد و پروتکل ارتباطی مورد استفاده، تبادلات اشیاء مربوط به هر درگاه را از دیگر واحد‌ها به آن درگاه هدایت می‌کند.

# پروتكل ارتباطي با برنامه‌هاي كاربردي

## مقدمه

يكي از قابليت‌هايي كه پلتفرم در اختيار قرار مي‌دهد، امكان توسعه برنامه‌هاي كاربردي است كه مي‌توانند از امكانات پلتفرم براي دسترسي به اشياء و کار با پلتفرم استفاده كنند. همان­طور که قبلا هم توضیح داده شد، این APIهای از طریق واحد API Server فراهم می‌گردد. هر كاربر با توجه به نياز خود مي‌تواند برنامه خاص خود را توسعه دهد. پلتفرم پیشنهادی واسط‌هاي برنامه‌نويسي كه در قالب يك سري پروتكل پياده‌سازي مي‌شوند را در اختيار اين برنامه‌هاي كاربردي قرار می­دهد. به عنوان مثال امكان توسعه يك برنامه كاربردي خارج از پلتفرم با قابليت اتصال پلتفرم براي مديريت وجود خواهد داشت.

## كليات پروتكل

کلیات پروتکل پیشنهادی مبتنی بر معماری REST است كه در واسط کاربری نیز استفاده شده است. REST مخفف عبارت Representational State Transfer است و متکی بر یک پروتکل ارتباطی بدون حالت[[5]](#footnote-5)، کلاینت/سرور[[6]](#footnote-6) و با قابلیت cache کردن می‌باشد. در این معماری از HTTP برای تعامل بین واحد‌ها استفاده می‌شود. ایده اصلی معماری REST این است که به جای استفاده از مکانیزم‌های پیچیده ای مانند CORBA، RPC یا SOAP برای اتصال ماشین‌ها ازHTTP ساده برای برقراری ارتباط بین ماشین‌ها استفاده شود.

از لحاظ رویکرد برنامه نویسی REST جایگزینی ساده برای سرویس‌های وب است. توسعه‌پذیری در تعاملات میان اجزا، عمومیت واسط ها، توسعه مستقل اجزا و استفاده از واسطه‌ها از کلیدی ترین اهداف معماری REST می‌باشد. لازم به ذکر است که استفاده از معماری REST در برنامه‌نویسی کارایی، سادگی، انعطاف‌پذیری و قابلیت اطمینان را افزایش می‌دهد.

## جزييات API مربوط به برنامه­های کاربردی

یکی از ویژگی‌های API ارائه شده این است که وابستگی به انواع ابزار هایی که از پلتفرم استفاده میکنند ندارد. توابع API به زبان PHP نوشته شده‌اند و امکان دسترسی به تمامی ‌توابع ارائه شده توسط API-Server که در بخش3-2 ارائه شده است توسط برنامه‌های کاربردی به صورت مستقیم وجود دارد. به عنوان مثال سناریوی ثبت نام کاربر از طریق اپلیکیشن اندروید در شکل ‏5‑1 نشان داده شده است. در این سناریو کاربر از طریق اپلیکیشن موجود در موبایل خود گزینه ثبت نام را انتخاب می‌کند. در ادامه توابع مربوط به ثبت نام از بخش API-Server فعال می‌گردند و اطلاعات کاربر در سیستم ثبت می‌گردد.

--

شکل ‏5‑1. سناریوی ثبت نام کاربر از طریق اپلیکیشن اندروید

# پروتكل ارتباطي با ابزارهاي تحليل داده

## مقدمه

با توجه به بررسی­های انجام شده در سند تحلیل نیازمندی­ها (IoT-RA-DP-v1.0)، سامانه باید قابلیت اتصال به ابزارهای تحلیل داده را فراهم کند. در حقیقت در پلتفرم­های اینترنت اشیا با توجه به حجم داده زیادی که از اشیا دریافت می­کنند امکان اتصال به ابزار تحلیل داده بسیار حائز اهمیت می­باشد. این سرویس در واقع محل اتصال پلتفرم اینترنت اشیاء به ابزارهای تحلیل داده است که داده­های جمع­آوری شده را در اختیار ابزار تحلیل داده قرار می­دهد.

## اتصال به ابزارهاي داده

در معماری ارائه شده برای اینترنت اشیاء که در شکل ‏2‑2 نشان داده شده است واحد Data Analyses مسئول پیاده‌سازی سرویس اتصال به ابزار تحلیل داده است. با توجه به ارزیابی صورت گرفته در تحلیل نیازمندی‌ها، ابزار Spark برای تحلیل داده در نظر گرفته شد که از زبان Java استفاده می‌کند. ابزار انتخاب شده برای پردازش داده­های با مقیاس بزرگ مناسب می­باشد. این واحد از سه API به نام­های SparkTextRead، SparSQLDataRead و SparkMongoDataRead تشکیل شده است.

SparkTextRead جهت تعامل با منابع داده­ای متنی در نظر گرفته شده است. در این بخش، برنامه­های طراحی شده با استفاده از کتابخانه­های مختلفی مانند map و reduce که توسط ابزار spark فراهم می‌گردد، می­تواند پردازش­ها و عملیات مختلفی را بر روی داده­های منابع متنی انجام دهند. بر حسب نوع داده مورد نیاز، APIهای مختلفی را می­توان در این قسمت جهت ارتباط با پلتفرم قرار داد.

SparSQLDataRead نیز جهت تعامل با پایگاه­ داده­های ساخت یافته پیاده­سازی شده است. در صورتی که پلتفرم در آینده نیاز به پایگاه داده­های از نوع SQL داشته باشد از APIهای توسعه یافته در این بخش می­توان استفاده کرد. جهت اتصال به پایگاه داده SQL نیز از توابع load/save یا توابع jdbc استفاده می‌شود و سپس می­توان از توابع مختلف ابزار spark استفاده کرد. در حال حاضر نیز اتصالات مربوط به پایگاه داده PostgreSQL فراهم شده است.

SparkMongoDataRead نیز برای اتصال به پایگاه‌داده‌ی Mongo از رابط پیشنهادی آن برای زبان جاوا استفاده می‌کند. بدین ترتیب می‌توان داده‌های جمع‌آوری شده در پایگاه‌داده‌ی پلتفرم را بازیابی کرد. برای اتصال به‌‌ پایگاه‌داده از کلاس MongoClient استفاده می‌گردد. پس از بازیابی داده‌ها، مانند خواندن از پایگاه‌داده‌های ساخت‌یافته عمل می‌کنیم و پس از آن می‌توان از توابع مختلف ابزار spark استفاده کرد.

1. حسابرسي

## مقدمه

پلتفرم ارائه شده قابلیت ارائه سرویس جهت انجام امور تراکنش­ها و مدیریت پرداخت کاربران را دارد. برای استفاده رایگان نیز، اکانت یک یا دو ماهه برای یک یا دو سنسور ایجاد گردیده است. با توجه به اینکه پکیج انتخابی به صورت تعداد سنسور در ماه می­باشد نوع پرداخت به شکل prepaid می­باشد. در ادامه نوع سرویس­های فراهم شده در این بخش مورد بررسی قرار خواهد گرفت

## سرویس­های ارائه شده در این بخش

سرویس­های ارائه شده در این بخش دو کاربر نهایی و مدیر پلتفرم را پوشش می­دهد.

* سمت کاربر
  + خرید بسته توسط کاربر با امکان انتخاب یکی از درگاه­های بانکی
  + امکان مشاهده و پرداخت فاکتورهای تخفیفی تعریف شده از سمت ادمین در بخش فاکتورهای پرداخت نشده
  + مشاهده وضعیت بسته‌هاي خریداری شده
  + مشاهده تراکنش­های انجام شده
* مدیریت بسته‌ها در سمت مدیر پلتفرم(ادمین)
  + نمایش کل تراکنش‌های موفق و ناموفق به ازای کل پلتفرم و به ازای هر کاربر
  + تعریف بسته‌های جدید
  + تعریف تخفیف برای کاربران (صدور فاکتور)
  + فعال و غیر فعال کردن درگاه‌های پرداخت تعریف شده
  + محاسبه‌ی قیمت پیشنهادی برای بسته‌ها بر اساس فرمول از پيش تعيين شده
  + مشاهده بسته­های تعریف شده در سیستم

## فرايند خريد و پرداخت

نحوه ارتباط با سیستم­های بانکی و پرداخت از طریق درگاه بانکی است. با توجه به زمانبر بودن دریافت نماد الکترونیکی، در حال حاضر عمل پرداخت به صورت ساختگی از طریق درگاه زرین پال فراهم شده و تست گردیده است. به هنگام تحویل به کارفرما و فراهم شدن نماد الکترونیکی با توجه به درگاه­های انتخابی از سمت کاربر این بخش تکمیل خواهد شد و جزئیات آن در اختیار کارفرما قرار خواهد گرفت

# پايگاه‌داده

## مقدمه

پایگاه داده MongoDB در پلتفرم استفاده شده است. پایگاه داده MongoDB را می‌توان یکی از پرمخاطب‌ترین پایگاه‌های داده‌ موجود در جمع اعضای خانواده NoSQL‌ دانست. این پایگاه داده‌ یک مدل منعطف، پویا و سندگرا را ارائه می‌کند که ساختاری با خروجی بسیار بالا و قابلیت مقیاس‌پذیری آسان را دارا است. این نوع پایگاه داده در پروژه‌ها و سیستم‌هایی که با حجم بزرگی از داده­ها مواجه هستند مورد نیاز است.

بدلیل متغیر بودن ساختار داده‌ها و حجم زیاد داده‌ها در پلتفرم اینترنت اشیا، یک پایگاه داده‌ی NoSQL که در آن از جدول‌بندی داده‌ها استفاده نمی‌شود، باید استفاده گردد. به همین دلیل یکی از مناسب‌ترین گزینه‌ها استفاده از MongoDB می‌باشد. در این بخش مفاهیم مطرح، طراحی پایگاه داده و همچنین پایگاه داده­های استفاده شده در سیستم تعریف می‌گردند.

* 1. خوشه بندی در پایگاه داده Mongo

در پایگاه داده MongoDB می‌توان داده‌ها را خوشه‌بندی کرد، به این معنا که همزمان چندین پایگاه داده را راه‌اندازی کرده و سپس با سیستم Clustering آن‌ها را مدیریت کرد. این قابلیت مزایای زیر را دارد:

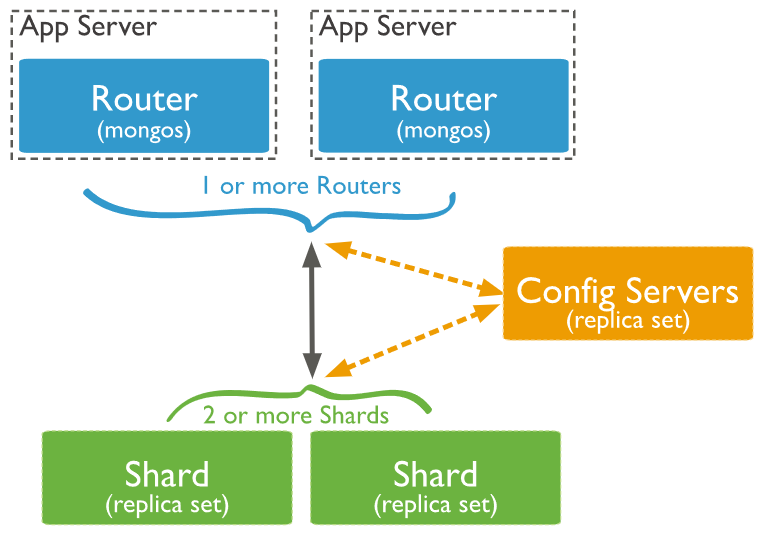
* سرعت خواندن و نوشتن بیشتر
* استفاده بهینه‌تر از فضا
* خواندن و نوشتن به صورت Partial

در این پایگاه داده دو نوع خوشه‌بندی وجود دارد:

* Hashed Sharding: در این نوع خوشه بندی،MongoDB داده‌ها را با روش Hashکردن، میان چندین پایگاه داده تقسیم می‌کند. در این حالت اگر یکی از پایگاه داده‌ها از سرویس خارج گردید و داده­های آن از بین رفت، بقیه داده­ها قابل دسترس هستند.
* Ranged Sharding: در خوشه‌بندی مرتب‌سازی‌شده، یک داده به قسمت‌های مختلفی تقسیم شده و در چندین پایگاه داده ذخیره می‌گردد.

با توجه به طراحی فعلی و نوع استفاده از داده­های پلتفرم در طراحی از نوع Hashed Sharding استفاده شده است.

معماری ارائه شده در شکل ‏8‑1 ساختار خوشه بندی ذکر شده را نمایش می­دهد.



شکل ‏8‑1. معمای خوشه­ای

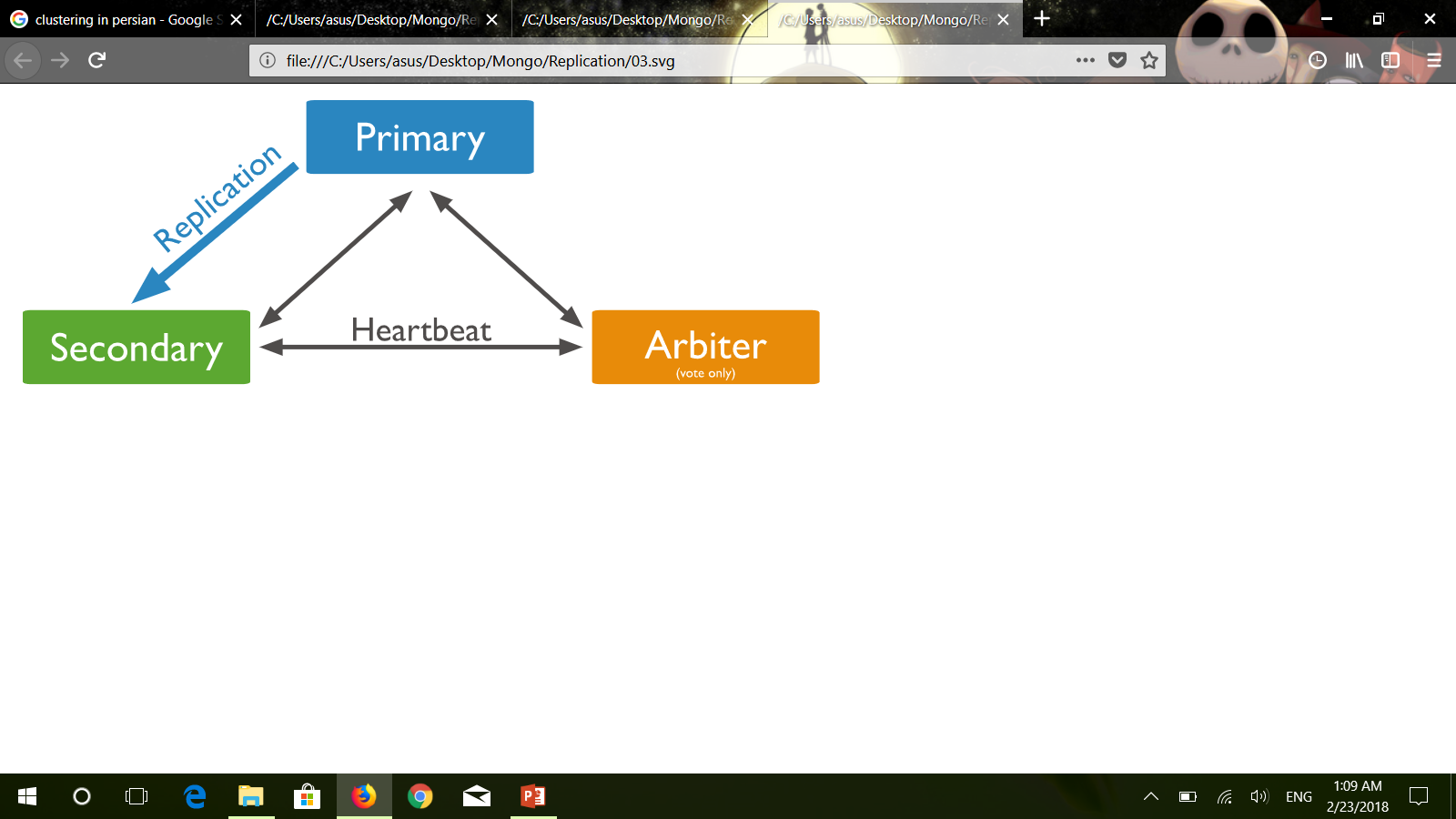
* config: پایگاه داده مبدا را که بقیه MongoDBها به آن متصل می‌شوند را Config یا همان Configuration می­نامند که تنظیمات مربوط به Clustering را در خود ذخیره می‌کند.
* Shard: پایگاه داده‌هایی را که به Config متصل می‌شوند و داده‌ها را به صورت خوشه‌بندی شده در خود ذخیره می‌کنند.
* Mongos: این قسمت مثل یک روتر عمل می‌کند و رابطی بین برنامه ی کاربر و شاردهای کلاستر است.
  1. تکثیر پایگاه­داده

امکان دارد تحت شرایط خاصی، ماشینی که پایگاه داده روی آن اجرا می‌شود از سرویس خارج گردد و داده‌های آن از بین برود. پایگاه داده Mongo برای این موضوع راه‌حل تکثیر (Replication) را پیشنهاد می‌کند، به صورتی که چندین پایگاه داده Mongo راه‌اندازی کرده و یک کپی از هر داده در آن­ها نگهداری می‌شود.. از مزایای تکثیر می‌توان به موارد زیر اشاره کرد:

* دسترسی پایدار به داده‌ها
* از بین نرفتن داده‌ها
* کاربر می‌تواند درخواست خواندن داده را، همزمان به چندین سرور ارسال کند
* استفاده از نسخه‌های اضافی برای اهداف اختصاصی مانند: بازیابی فوری، تهیه نسخه پشتیبان(Backup) یا گزارش‌دهی

فرآیندها یا پایگاه­داده­های Mongo که یک مجموعه داده تکراری را نگه داری می­کنند با عنوان Replica set معرفی می‌گردد. در هر Replica Set تنها یک پایگاه داده اصلی (primary) وجود دارد. عملیات خواندن در این نوع معماری را می­توان طوری تنظیم کرد که هم توسط پایگاه داده اصلی و هم توسط پایگاه داده ثانویه (secondary) قابل انجام باشد. در نتیجه می­توان از طریق افزودن پایگاه داده secondary به Replica set، کارایی در عملیات خواندن از پایگاه داده را افزایش داد. اما در عملیات نوشتن در پایگاه داده، تنها در پایگاه داده اصلی نوشته می‌شود و سپس به سایر پایگاه­ داده­های موجود در Replica-set پخش می‌شود. در نتیجه افزودن پایگاه داده­های ثانویه به مجموعه تاثیر در سرعت عملیات نوشتن ندارد.

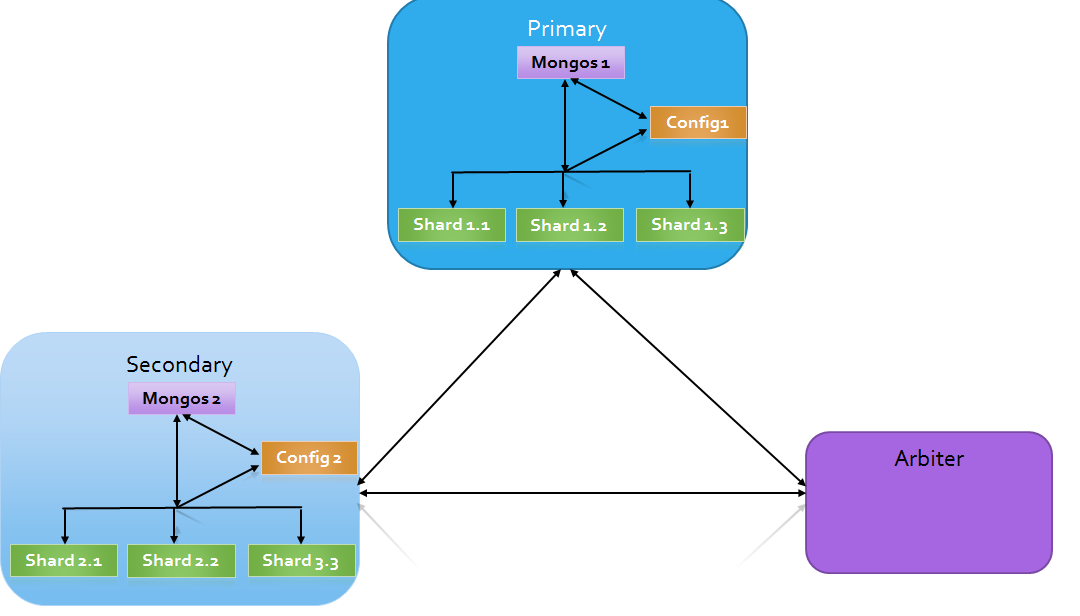
در این نوع معماری در حالتی که چندین پایگاه Secondary وجود داشته باشد، در صورت قطع ارتباط و از دست رفتن پایگاه داده اصلی، سایر اعضا یا Secondary‌ها در رابطه با انتخاب پایگاه داده اصلی تصمیم گیری می­کنند. اما در صورتی که در Replica Set تنها پایگاه داده اصلی و ثانویه وجود داشته باشد، از مولفه­ای به نام داور (Arbiter) استفاده می‌گردد. معماری آن درشکل ‏8‑2 نشان داده شده است. Arbiter مانند بقیه اجزا داده­ای را نگه­داری نمی‌کند. در این صورت در حالتی که پایگاه داده اصلی از دسترس خارج گردد، Arbiter در انتخاب پایگاه داده ثانویه به عنوان پایگاه داده اصلی نقشی اساسی بازی می‌کند. . باید در نظر داشت که نباید arbiter در سیستم­هایی که میزبان Replication‌های اصلی و ثانویه است راه­اندازی گردد.



شکل ‏8‑2. معماری تکثیر پایگاه داده

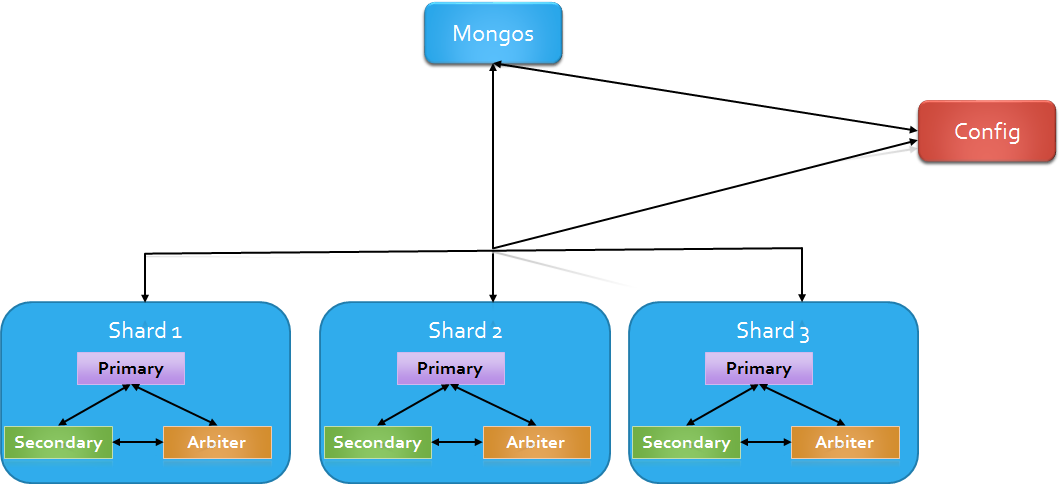
* 1. معماری پیشنهادی

معماری­های ارائه شده عموما در دو مدل ترکیب می‌گردند. در شکل ‏8‑3، پایه و اساس بر اساس معماری تکثیر است ولی در داخل primary و secondary از تکنولوژی sharding استفاده شده است. در این حالت ۶ پایگاه داده جهت خوشه‌بندی و دو Mongo برای Config و پایگاه داده‌ای برای داور بودن راه‌اندازی می‌گردد. و در آخر نیز دو mongos راه اندازی کرده که replSet آن­ها یکسان قرار می‌گیرد. در این مدل هم سرعت خواندن و هم سرعت نوشتن به صورت جداگانه در primary و Secondary افزایش پیدا کرده است.



شکل ‏8‑3. معماری تکثیر با استفاده از خوشه بندی

نوع دیگری از ترکیب معماری در شکل ‏8‑4 است. این نوع معماری متداول تر بوده است. در این حالت mongos درخواست­های دریافتی ما بین shardهای مختلف تقسیم می‌کند. که در داخل هر shard نیز یک Replica-set تعریف شده است که پشتیبانی داده­های موجود در آن shard را فراهم می‌کند. در این حالت با تقسیم داده­ها به صورت متناسب ما بین shardها و تقسیم درخواست بر اساس بار هر shard می­توان سرعت خواندن و نوشتن را بسیار بهبود بخشید. در نهایت availability داده­ها نیز افزایش پیدا می‌کند. در پلتفرم پیشنهادی این ترکیب از Replication و Sharding پیشنهاد و تست شده است که باعث می‌گرد قابلیت تحمل و ترمیم نسبت به خرابی بهبود یابد و هم افزایش سرعت و کاهش بار روی ماشین‌های مرتبط به پایگاه داده ایجاد گردد. در نهایت بر اساس میزان فضای در اختیار قرار داده شده و اهمیت داده­های مربوط به پایگاه داده­های سیستم میزان اعمال آن بر تمام یا بخشی از پایگاه داده­های سیستم تصمیم گیری خواهد شد. ولی این معماری برای پایگاه داده­های پارس شده ضروری می­باشد.



شکل ‏8‑4. معماری خوشه بندی با استفاده از تکثیر

## انواع پایگاه داده­های پلتفرم

### پايگاه داده‌هاي خام

در این بخش از پایگاه داده اطلاعات سنسور‌ها بدون پردازش خاص ذخیره سازی می‌گردد. در معماری پلتفرم اینترنت اشیاء که در شکل ‏2‑2 نشان داده شده است این بخش با نام Thing-RD مشخص شده است.

### پايگاه داده‌هاي پارس شده

در این بخش از پایگاه داده اطلاعات پردازش شده از سنسور‌ها ذخیره سازی می‌گردد. در معماری پلتفرم اینترنت اشیاء که در شکل ‏2‑2 نشان داده شده است این بخش با نام Thing-PD مشخص شده است.

### پايگاه داده‌هاي پلتفرم

در این بخش از پایگاه داده اطلاعات مربوط به دیگر بخش‌های پلتفرم اینترنت اشیاء مانند کاربران، پرداخت و پروژه‌ها ذخیره سازی می‌گردد. در معماری پلتفرم اینترنت اشیاء که در شکل ‏2‑2 نشان داده شده است این بخش با نام Platform به عنوان یکی از اجزای Databases مشخص شده است.

## سایر موارد مرتبط به پایگاه داده­ها

همانگونه که در شکل ‏2‑2 نشان داده شده است برای ارتباط با پایگاه داده در Up-Link از Redis نیز استفاده شده است. این واحد نقش یک واسط (حافظه میانجی) را بین پایگاه داده و واحد Up-Link ایفا می‌کند. این واحد اطلاعات مربوط به اشیاء را پیش از ثبت شدن در پایگاه داده نگهداری می‌کند. به هنگامی‌که حجم داده‌های جمع آوری شده از اشیاء افزایش میابد این واحد در کاهش بار کاری پایگاه داده موثر خواهد بود. سناریوی استفاده از این مولفه در بخش مولفه Up-link شرح داده شده است.

# طراحی فرآیند مدیریت اشیا

با توجه به نیازهای کارفرما و طراحی انجام شده در رابطه با مدیریت اشیا که تعریف کامل آن­ها در فصل 3 با عنوان مولفه­ها و فرآیند­های پلتفرم اشیا آورده شده است. اهم موارد مربوط به مدیریت اشیا به شرح زیر می­باشد:

* مراحل شناسایی و دریافت داده یک شی دارای پیش نیاز­هایی است
  + ابتدا باید یک پروژه برای آن شی ساخته شود.
  + گذرگاه مربوطه که به شی متصل است تعریف گردد.
  + پروفایل شی ساخته گردد.
* پس از انجام پیش­نیازهای مربوطه مراحل زیر جهت فعال سازی شی انجام می­گردد.
  + در مرحله تعریف شی، اطلاعات آن به همراه پروفایل آن شی معرفی می­گردد.
  + در مرحله آخر فعال سازی شی با استفاده از وارد کردن اطلاعاتی مانند Device Address، Application Session Key، Network Session Key و... انجام می­گردد.
* پس از انجام مراحل بالا شی آماده ارسال کدک به آن و نمایش داده می­باشد.

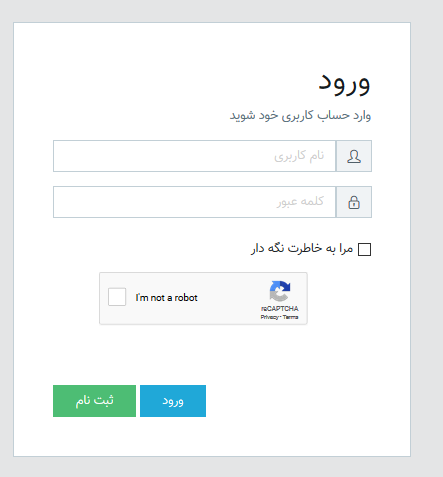
علاوه بر موارد ذکر شده امکانات زیر نیز در رابطه با اشیا در اختیار کاربر قرار داده شده است:

* افزودن، حذف و ویرایش تکی اطلاعات هر شی
* افزودن، حذف و ویرایش اطلاعات اشیا به صورت دستی
* امکان تعریف شی در چند پروژه کاربر
* امکان بررسی لاگ گذرگاه جهت عیب­یابی
* و ...

# طراحی و mockupپنل کاربر

در این بخش با توجه به تغییر مداوم در طراحی واسط گرافیکی کاربری سامانه برخی از صفحاتی که طراحی شده­اند و تا حدی مورد تایید کارفرما بوده است به صورت شکل نمایش داده شده است. شایان ذکر است که این صفحات تمام صفحات پنل کاربری نمی­باشند. لیست تقریبا کاملی از صفحات کاربر در بخش 3-2 مولفه API-Server با عنوان کاربر نهایی آورده شده است. اطلاعات کامل این واسط­های گرافیکی و توضیحات آن­ها در مستندات نهایی پلتفرم آورده خواهد شد.

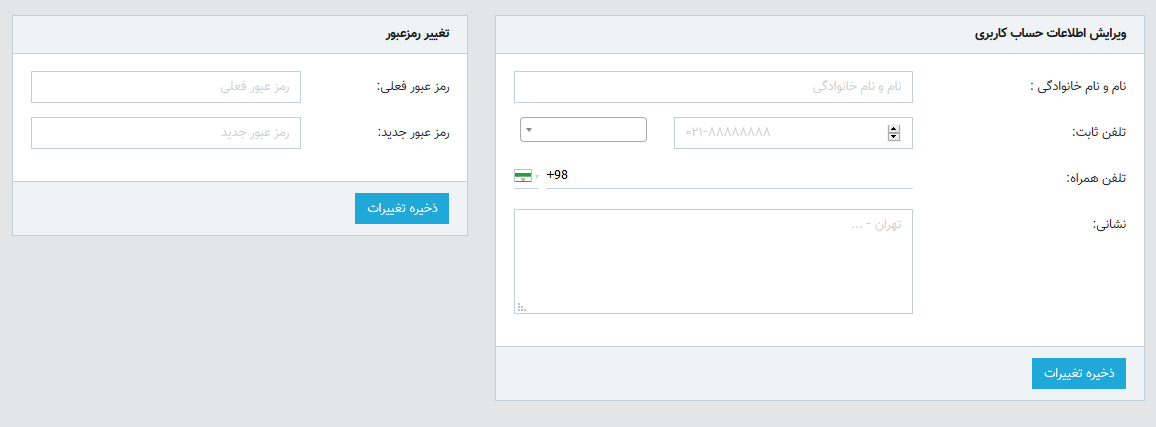
شکل ‏10‑1، شکل ‏10‑2 و شکل ‏10‑3 صفحات اولیه شامل ورود ، ثبت نام و ویرایش اطلاعات کاربران را نمایش می­دهد.



شکل ‏10‑1. صفحه ورود کاربران

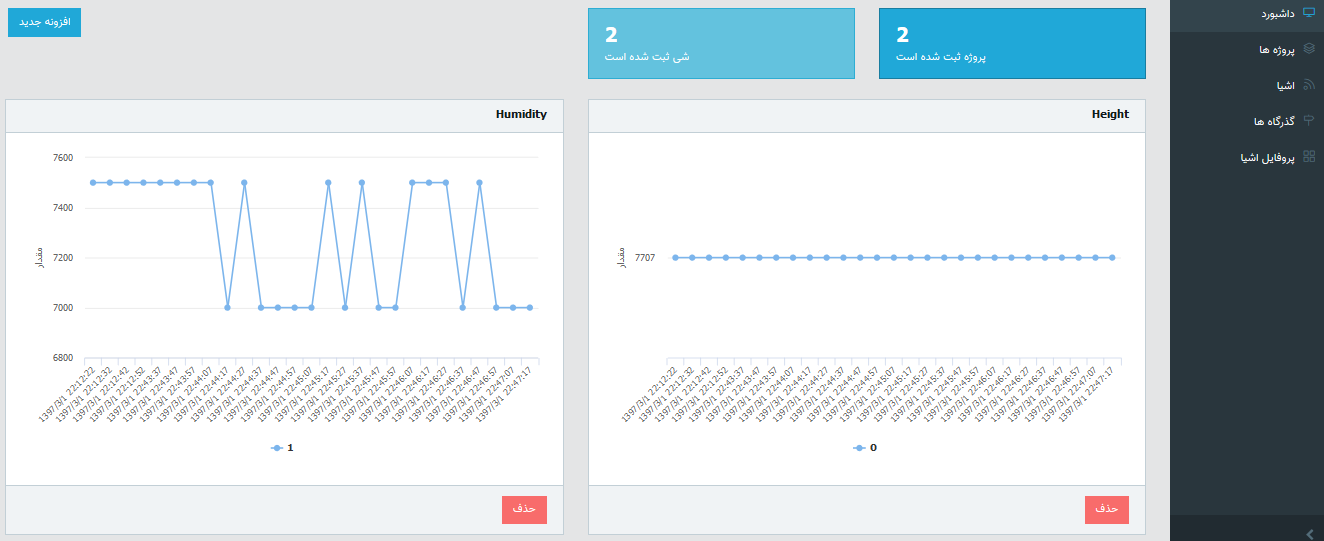


شکل ‏10‑2. صفحه ثبت­نام کاربران



شکل ‏10‑3. صفحه ویرایش اطلاعات حساب کاربری

شکل ‏10‑4 داشبورد کاربر سامانه را نمایش می­دهد که در حال حاضر امکان نمایش تعداد پروژه­ها، اشیا و همچنین امکان افزودن نمودار در آن وجود دارد. در شکل ‏10‑5 امکان افزودن افزونه نمایش داده شده است.

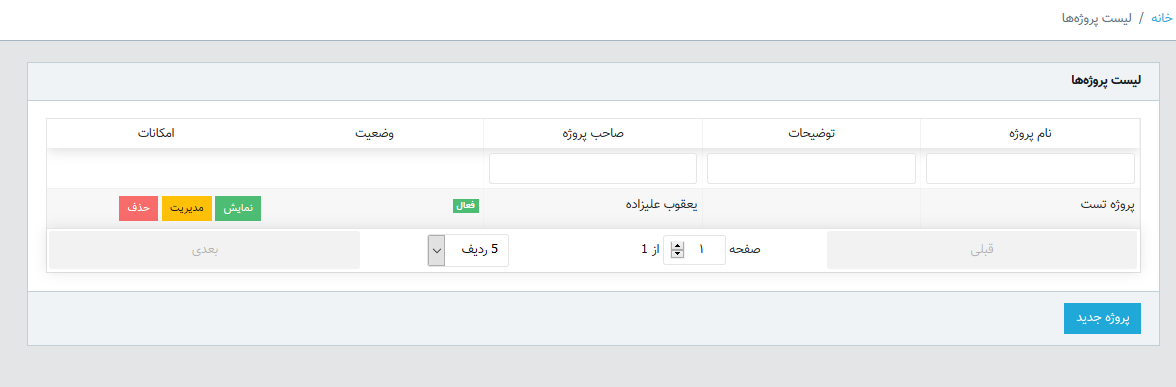


شکل ‏10‑4. داشبورد سامانه

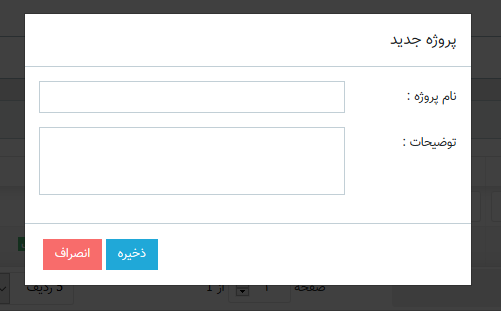


شکل ‏10‑5. صفحه افزودن افزونه جدید به داشبورد

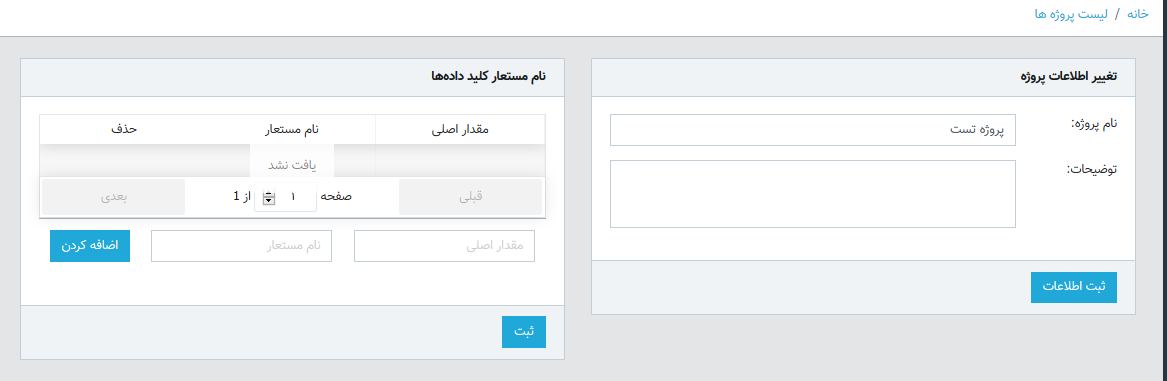
شکل ‏10‑6 تا شکل ‏10‑11 امکانات مربوط به مدیریت پروژه را نمایش می­دهد. که شامل بخش­های مربوط به لیست پروژه­ها وامکانات مدیریتی ، ، لیست اشیا و امکانات مرتبط، لیست کدک­ها و سناریوها و امکانات مدیریتی آن می­باشد.



شکل ‏10‑6. نمایش لیست پروژه ها



شکل ‏10‑7. صفحه تعریف پروژه جدید



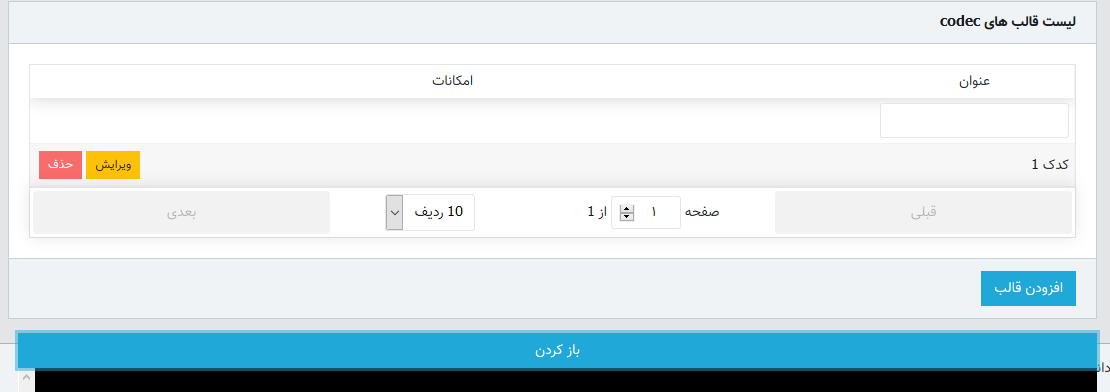
شکل ‏10‑8. صفحه مدیریت پروژه (1)



شکل ‏10‑9. صفحه مدیریت پروژه (2)

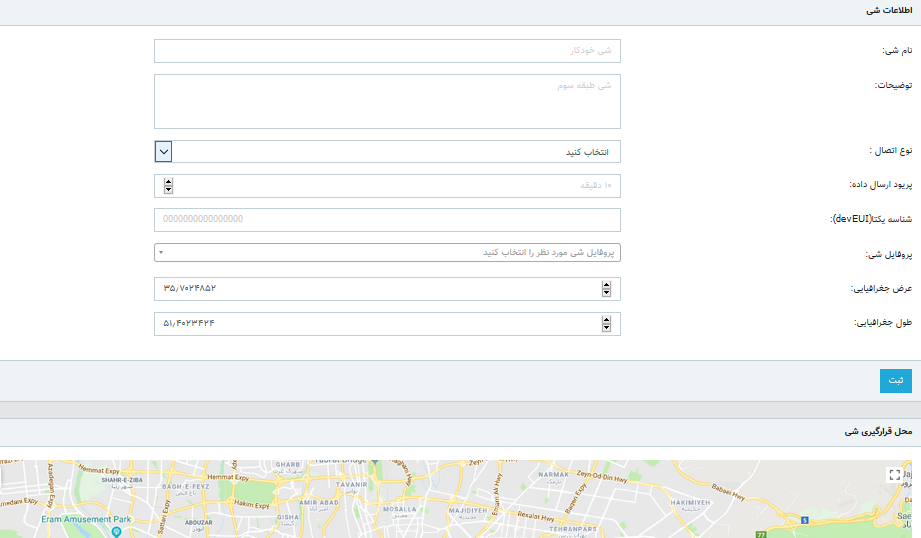


شکل ‏10‑10. صفحه مدیریت پروژه (3)



شکل ‏10‑11. صفحه مدیریت پروژه (4)

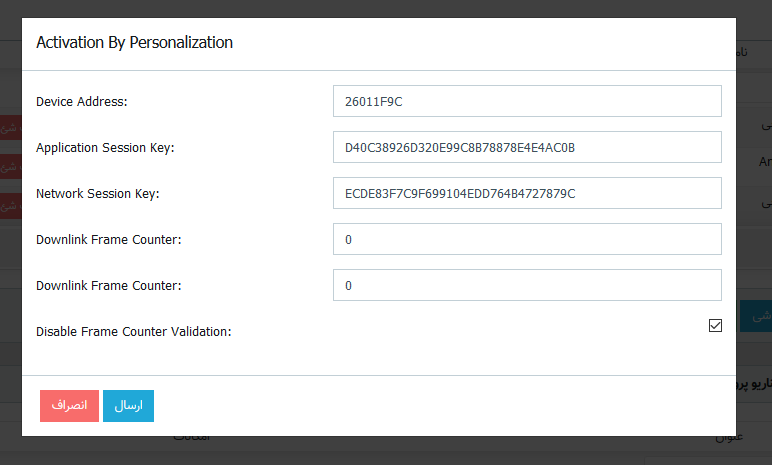
با توجه به امکاناتی که در صفحه مدیریت پروژه قرار داده شده است در شکل ‏10‑12 تا شکل ‏10‑16 صفحات امکانات مربوطه شامل افزودن شی، افزودن دسته­ای اشیا، فعال سازی شی، ارسال داده به شی و صفحه ارسال و بررسی کدک نمایش داده شده است.



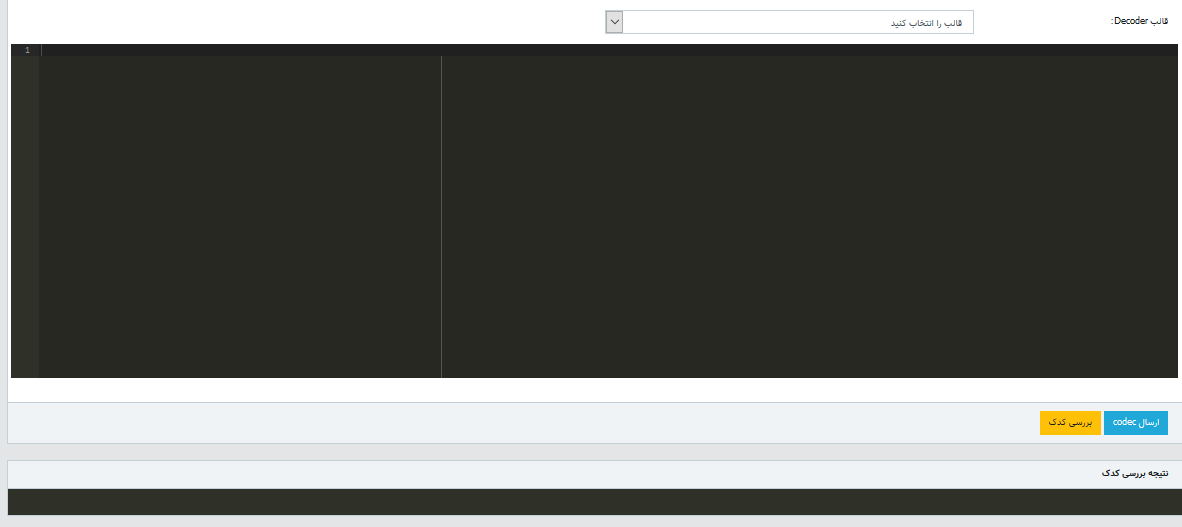
شکل ‏10‑12. صفحه افزودن شی جدید



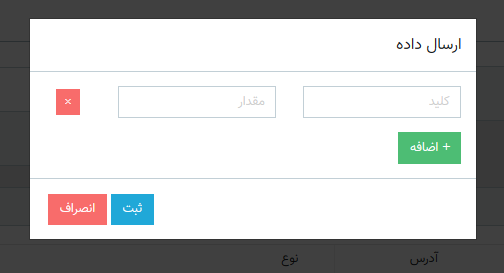
شکل ‏10‑13. صفحه افزودن دسته­ای اشیا



شکل ‏10‑14. صفحه فعال سازی اشیا

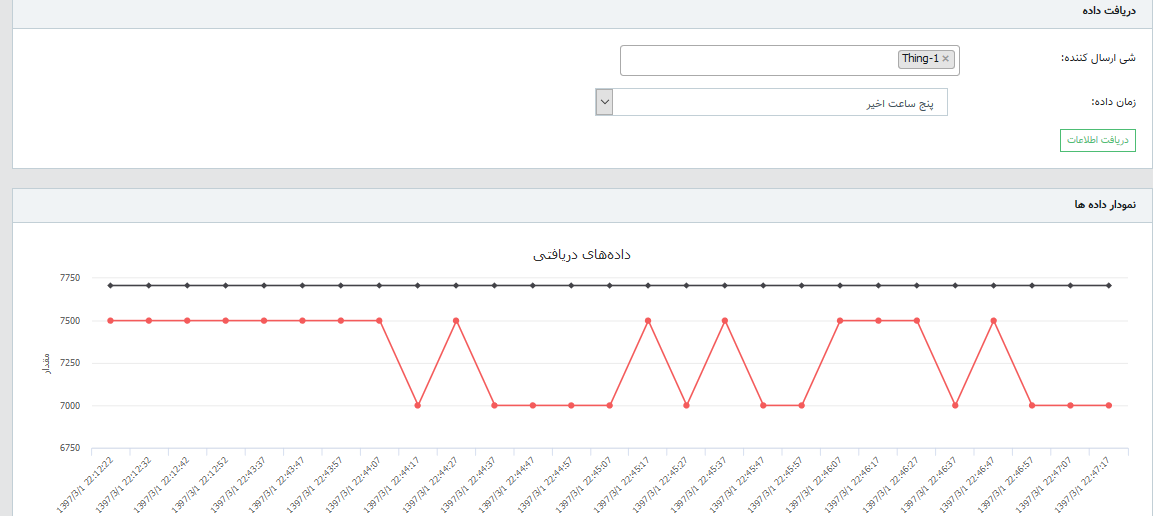


شکل ‏10‑15. صفحه ارسال کدک

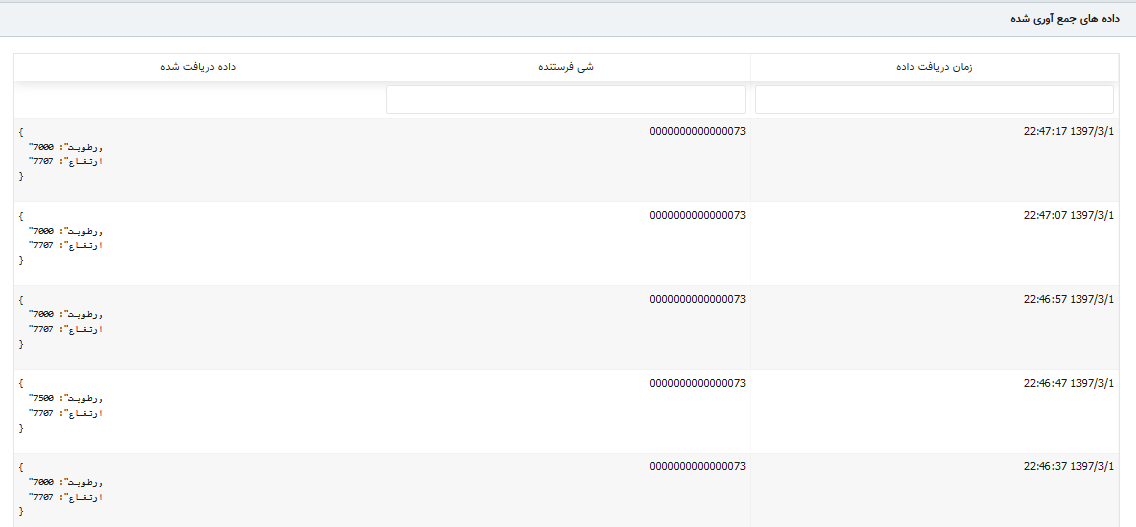


شکل ‏10‑16. صفحه ارسال داده به شی

در شکل ‏10‑17و شکل ‏10‑18 صفحه نمایش پروژه که دریافت اطلاعات از اشیا مربوط به پروژه را به صورت نموداری و جدولی بر عهده دارد را نشان داده شده است.



شکل ‏10‑17. صفحه نمایش پروژه (1)



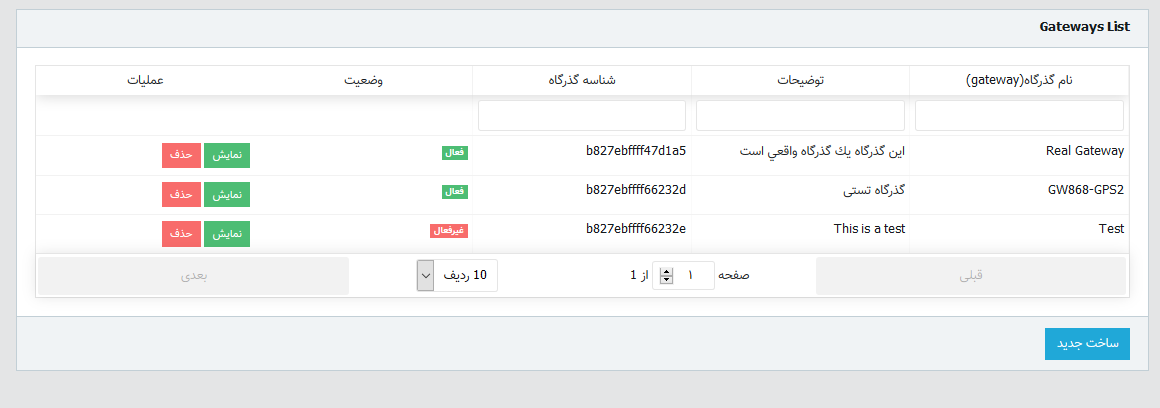
شکل ‏10‑18. صفحه نمایش پروژه (2)

در ادامه شکل ‏10‑19 از منوی اصلی لیست اشیا را نمایش می­دهد.

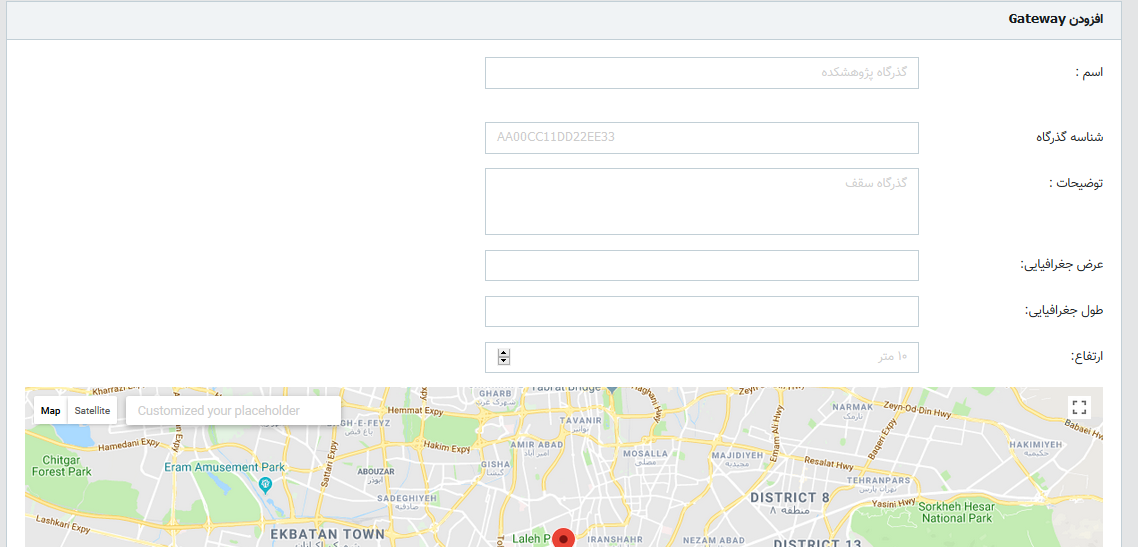


شکل ‏10‑19. صفحه لیست اشیا

شکل ‏10‑20 و شکل ‏10‑21 امکانات مربوط به گذرگاه را نمایش داده­اند که شامل نمایش لیست گذرگاه­ها، حذف و افزودن گذرگاه می­باشد.



شکل ‏10‑20. صفحه لیست گذرگاه­ها

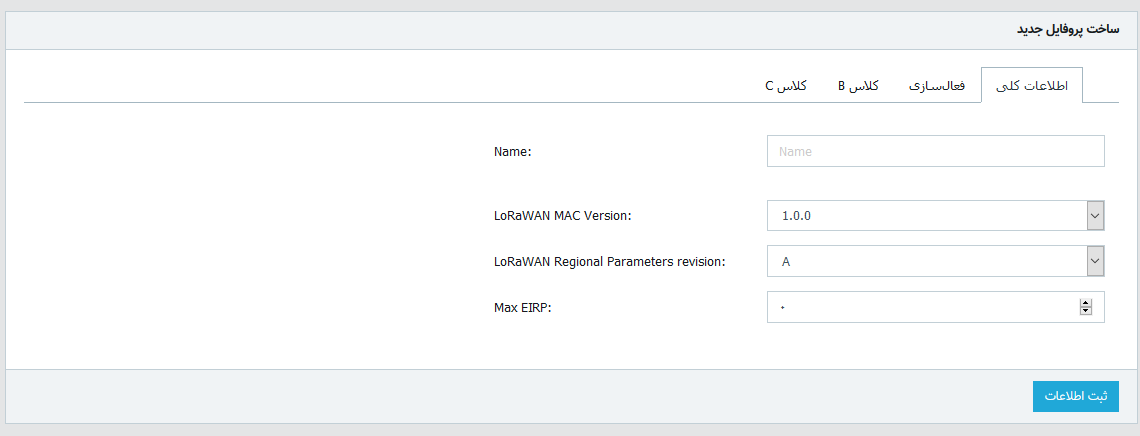


شکل ‏10‑21. صفحه افزودن گذرگاه

شکل ‏10‑22 و شکل ‏10‑23 نیز امکانات مرتبط به مدیریت پروفایل اشیا شامل لیست، مشاهده، حذف و ساخت را در اختیار کاربر قرار می­دهد.



شکل ‏10‑22. صفحه لیست پروفایل اشیا

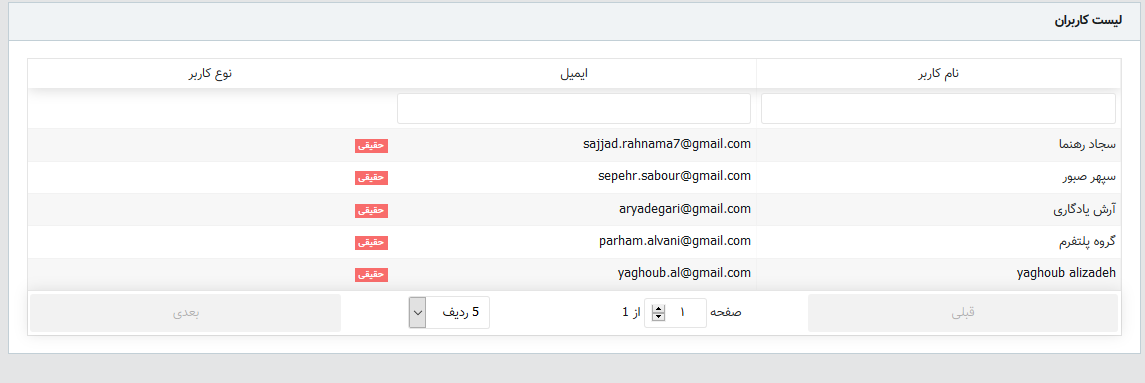


شکل ‏10‑23. صفحه ساخت پروفایل اشیا

# طراحی و mockupپنل مدیریت

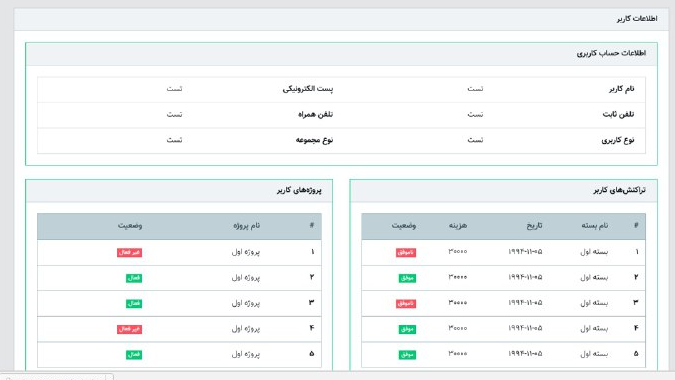
در پنل مدیریت سامانه نیز به مانند پنل کاربری، لیست تمام صفحات در این گزارش به دلیل تغییرات مداوم ذکر نشده است. لیست اولیه­ای از صفحات پنل مدیریت در بخش 3-2 مولفه API-Server با عنوان کاربر مدیر پلتفرم آورده شده است. در ادامه لیستی از طراحی­های انجام شده آورده شده است.

در شکل ‏11‑1 لیست کاربران سامانه به همراه قابلیت جستجو و مشاهده وضعیت آن­ها نمایش داده شده است.



شکل ‏11‑1. صفحه لیست کاربران سامانه

برای هر کاربر اطلاعات آن در چهار بخش اطلاعات عمومی، بسته­های خریداری شده، تراکنش­ها، پروژه ها و اشیا در شکل ‏11‑2 و شکل ‏11‑3 نمایش داده شده است.

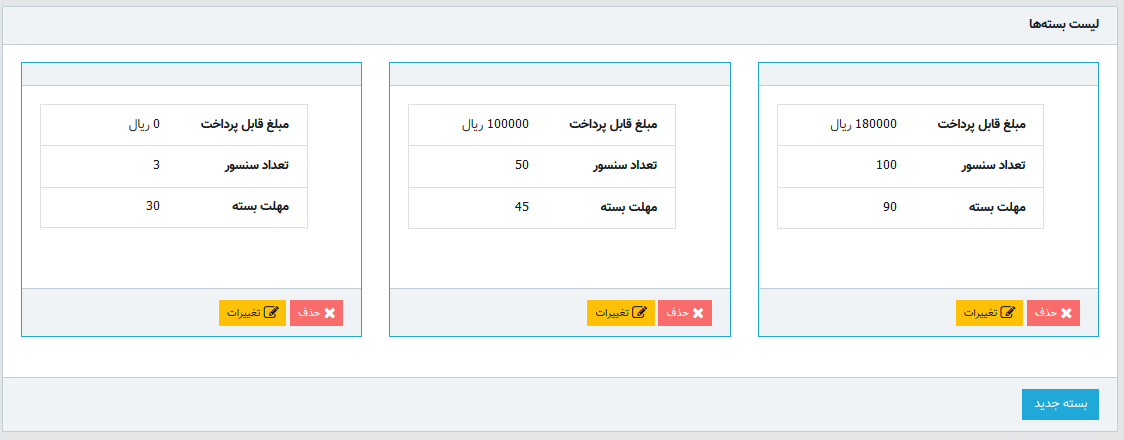


شکل ‏11‑2. صفحه اطلاعات کاربر (1)

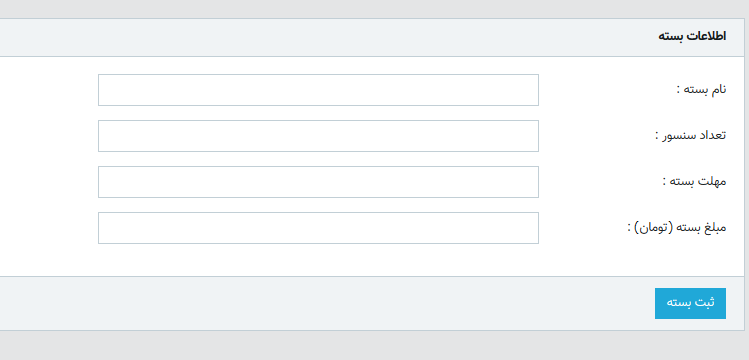


شکل ‏11‑3. صفحه اطلاعات کاربر (2)

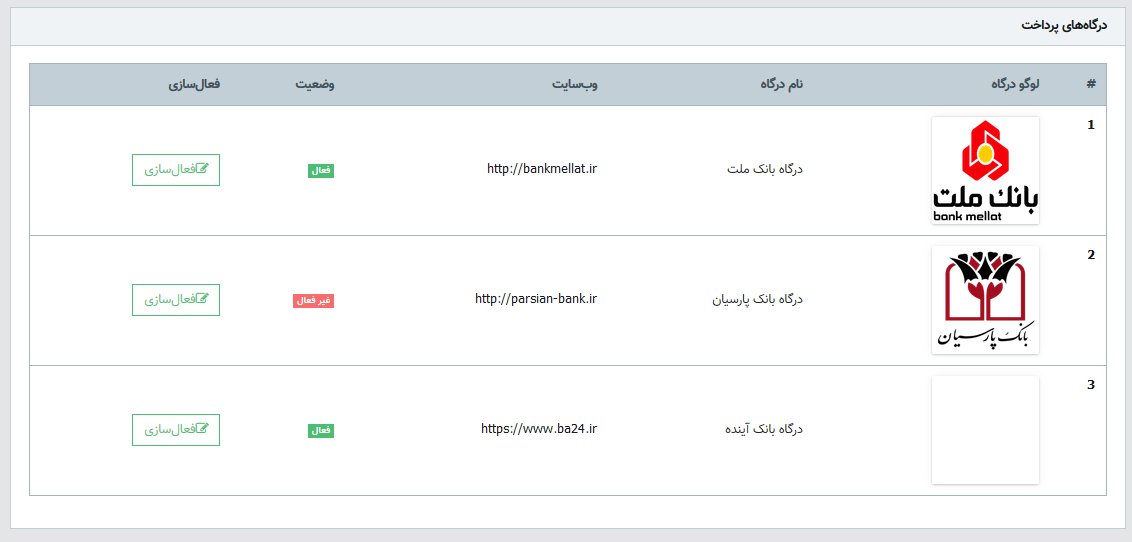
در نهایت شکل ‏11‑4 تا شکل ‏11‑6 صفحات مربوط به لیست بسته­ها (شامل نمایش، حذف وایجاد بسته جدید)، مدیریت درگاه­های پرداخت ولیست تراکنش­ها را نمایش می­دهد.



شکل ‏11‑4. صفحه لیست بسته­های موجود در سامانه



شکل ‏11‑5. صفحه افزودن بسته جدید



شکل ‏11‑6. صفحه مدیریت درگاه­های بانکی

1. Layered (n-tier) [↑](#footnote-ref-1)
2. Model View Controller [↑](#footnote-ref-2)
3. Event Driven [↑](#footnote-ref-3)
4. GUI [↑](#footnote-ref-4)
5. Stateless [↑](#footnote-ref-5)
6. Client/Server [↑](#footnote-ref-6)