

درخواست ارائه پیشنهاد ‌(RFP)

توسعه بستر ‌MLOps

فهرست

[1 مقدمه 3](#_Toc137115679)

[2 معماری کلان 6](#_Toc137115680)

[1-2 زیرساخت رایانش ابری گرانیت 7](#_Toc137115681)

[2-2 دیتا پلتفرم 7](#_Toc137115682)

[3-2 معماری محصول MLOps 7](#_Toc137115683)

[3 نیازمندی 12](#_Toc137115684)

[1-3 نیازمندی کارکردی 12](#_Toc137115685)

[1-1-3 بستر توسعه مدل 12](#_Toc137115686)

[2-1-3 مدیریت داده‌ 12](#_Toc137115687)

[3-1-3 مدیریت مدل 13](#_Toc137115688)

[2-3 نیازمندی غیر کارکردی 13](#_Toc137115689)

[4 فرایند توسعه محصول 15](#_Toc137115690)

[1-1-4 آزمایشگاه 17](#_Toc137115691)

[2-1-4 طرح آزمون پروژه 17](#_Toc137115692)

[5 مدیریت پروژه 19](#_Toc137115693)

[1-5 برآورد زمانی 19](#_Toc137115694)

[2-5 فازبندی 19](#_Toc137115695)

[پیوست۱: نیازمندی‌ها فنی در توسعه و استقرار محصول 20](#_Toc137115696)

[پیوست ۲: جداول امکانات و قابلیت‌های پلتفرم 23](#_Toc137115697)

[پیوست۳: الزامات پیشنهادهای فنی 25](#_Toc137115698)

1. مقدمه

در سال‌های اخیر نیازمندی‌های تحلیل داده، هوش مصنوعی و یادگیری ماشین بسیار گسترده شده است. سازمان‌های مختلف فعالیت‌هایی را در این زمینه شروع کرده‌اند و با استقرار تیم‌های تحلیل داده و هوش مصنوعی اقدام به تولید موتور‌های تحلیلی می‌کنند. این سازمان‌های در تجربه خود مکرراً به مشکلاتی برخورد می‌کنند که ناشی از عدم بلوغ فناوری و فرایندهای توسعه محصولات تحلیل داده است. این مشکلات عموماً شامل موارد زیر می‌باشد:

* عدم وجود ابزار و تکنولوژی استاندارد و یکسان در بین تیم‌ها و افراد یک سازمان
* عدم امکان مدیریت و کنترل دسترسی به دادگان سازمان بین تیم‌ها و متخصصین تحلیل داده
* عدم امکان استفاده بهینه از زیرساخت‌های پردازشی و ذخیره‌سازی سازمان در راستای تولید موتور‌های تحلیلی
* عدم امکان تجمیع زیرساخت‌های پردازشی و ذخیره‌سازی موجود سازمان برای این موضوع
* عدم وجود جریان کاری استاندارد و منسجم در مسیر تولید موتورهای تحلیلی (طراحی، توسعه مدل، عملیات)
* عدم امکان خودکارسازی فرایندهای CI/CD
* عدم امکان خودکارسازی فرایندهای آموزش مجدد مدل
* عدم وجود بستری جهت نگهداری نسخه‌های مختلف مدل و داده
* عدم امکان تولید نسخه‌های صنعتی و قابل‌استقرار از مدل به‌صورت خودکار
* عدم وجود بستر اجرای یکپارچه مدل‌های صنعتی شده
* سخت بودن همکاری بین تیم‌های مختلف درگیر در تحلیل داده به دلیل نبود استاندارد فرایند و فناوری مشخص
* دوباره و چندباره کاری‌های مکرر در زمان آماده‌سازی داده به دلیل نبود زیرساخت مشترک نگهداری داده قابل‌استفاده در آموزش مدل‌ها
* عدم وجود زیرساخت مانیتورینگ مدل‌های در حال اجرا
* ‌عدم جمع‌آوری لاگ مدل‌های در حال اجرا و استفاده از آن‌ها در آموزش مجدد مدل‌ها
* هزینه بسیار بالای تبدیل مدل‌های آموزش‌دیده به کدهای قابل‌اجرا درون‌برنامه‌های اصلی

مجموعه این مشکلات از نبود یک پلتفرم یکسان برای مدیریت فرایند طراحی، توسعه، استقرار مدل و عملیات نگهداری موتورهای تحلیلی ناشی می‌شود. در چند سال اخیر باتوجه‌به اینکه این موضوع یک مسئله جهانی است ابزارهایی برای حل آن توسعه‌یافته است که تمامی این ابزارها باتوجه‌به نوپا بودن مشکلات خاص خود را دارند. بااین‌وجود راهی به جز حل این موضوع با استفاده از ترکیبی مناسب از ابزارهای MLOps نیست. این پلتفرم‌ها صرفاً به جهت استفاده در تولید موتورهای یادگیری ماشین نیستند؛ بلکه بقیه موتورهای تحلیلی که فناوری‌های دیگری هم دارند می‌توانند از این پلتفرم استفاده کنند و استفاده از عنوان ML صرفاً به دلیل رواج این عنوان در صنعت است.

در نهایت MLOps ابزاری است که می‌تواند در جمع‌آوری داده، ذخیره‌سازی داده، ویژگی (Feature)و مدل، بهبود و پاک‌سازی داده، مهندسی نیازمندی (Requirement Engineering)، مهندسی ویژگی (Feature Engineering)، مهندسی داده (Data Engineering)، مهندسی مدل (Model Engineering)، تست و اعتبارسنجی مدل، نصب و راه‌اندازی مدل، جریان CI/CD/CT مدل، مدیریت داده و کد، نظارت و هشداردهی (Monitoring & Triggering) و ارائه بستر پردازشی و ذخیره‌سازی متمرکز ایفای نقش کند و بستری برای خودکارسازی فرایند توسعه موتورهای تحلیلی را فراهم نماید.

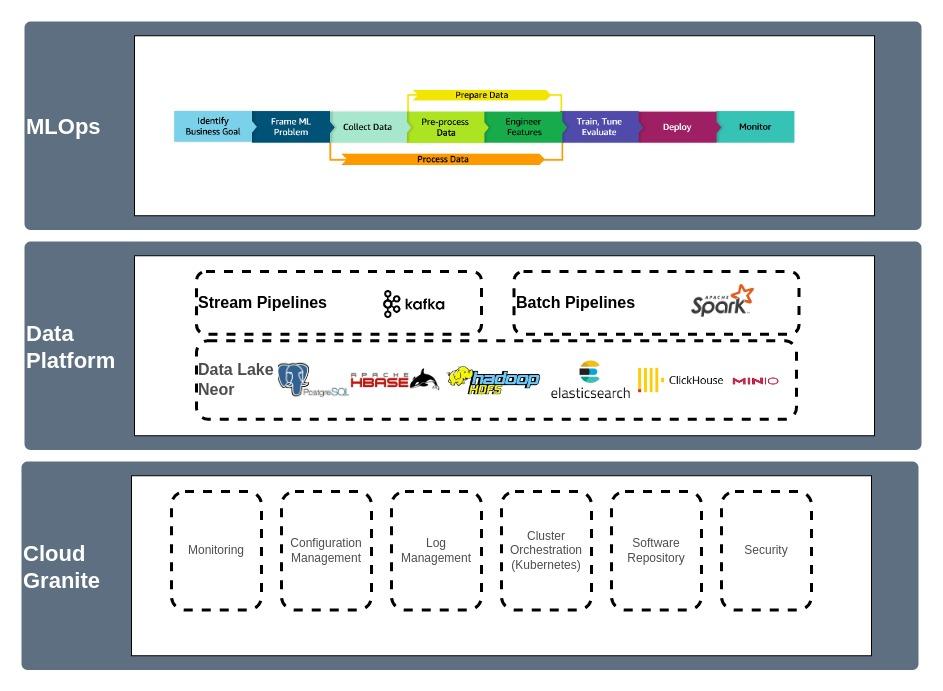
پیچیدگی‌های توسعه مدل‌های هوش مصنوعی بدون MLOps

* مدل‌های یادگیری ماشین عموماً به حجم‌ بالایی از داده وابسته‌اند.
* موضوع کنترل و مدیریت دسترسی به دادگان و مدل‌ها، مسئله‌ای غیرقابل‌اجتناب است.
* نسخه‌گذاری داده‌ها و مدل‌های مختلف یادگیری ماشین و نتایج ارزیابی آن‌ها باید قابل‌پیگیری و بررسی باشد. زمانی که داده‌ها متنوع و تعداد مدل‌ها زیاد می‌شود این مسئله اهمیت بیشتری پیدا می‌کند.
* سرویس‌های ارائه شده بر روی مدل‌های یادگیری ماشین باید متناسب با افزایش بار ورودی مقیاس‌پذیر باشند.
* آموزش مدل‌های یادگیری ماشین به‌صورت دوره‌ای و با دادگان جدید باید رخ دهد.
* پارامترهایی که در مدل‌های یادگیری ماشین تغییر داده‌ شده است و می‌تواند تأثیر فراوانی بر خروجی داشته باشد باید قابل‌پیگیری و نسخه‌گذاری باشد.
* مهندسی ویژگی‌ها به طور محسوسی بر صحت مدل تأثیر می‌گذارد بنابراین باید دائماً ویژگی‌هایی را که مدل با آن‌ها در تعامل است دنبال کنیم.
* مد‌ل‌ها در محیط عملیاتی باید دارای مانیتورینگ متناسب باشند و هرگونه تغییر در عملکرد آن‌ها باید منجر به واکنش متناسب گردد.
* رفع باگ‌ مدل‌های یادگیری ماشین نیازمند بستر مناسب توسعه است.

باتوجه‌به استراتژی شرکت سحاب پرداز به‌منظور تکمیل سبد محصولات و زیرساخت‌های خود دستیابی به این زیرساخت را هدف قرار داده است. البته باتوجه‌به تجربیات و زیرساخت‌های توسعه داده شده در سحاب نکات مختلفی در توسعه و تولید بستر ‌MLOps به‌منظور مطابقت و یکپارچگی آن با ابزارها و استانداردهای کنونی سحاب موردنیاز است که این موارد در این سند بیان خواهد شد.

1. معماری کلان

در حال حاضر برخی از محصولات از جمله دو محصول کلود (گرانیت) و دیتا پلتفرم توسط تیم فنی شرکت سحاب توسعه داده‌ شده و در چندین پروژه مورد بهره‌برداری قرار گرفته شده است؛ بنابراین این انتظار وجود دارد که محصول MLOps از طرفی سازگاری کافی با این پلتفرم‌ها را داشته باشد و از سرویس و خدمات آن‌ها به‌خوبی بهره بگیرد و همچنین فرایند توسعه و استقرار آن با استانداردهای و ملزومات این زیرساخت‌ها سازگار باشد. برای روشن‌شدن موضوع به‌اختصار به بیان ویژگی‌ها و دامنه این محصولات خواهیم پرداخت. در زیر شمای کلی از محصولات نمایش‌داده‌شده است.



معماری کلان نگهداری و تحلیل داده

* 1. زیرساخت رایانش ابری گرانیت

این پلتفرم وظیفه ایجاد بستر ابری را بر عهده دارد و برای این موضوع سرویس‌های مختلفی مانند مانیتورینگ، مدیریت لاگ و زیرساخت رایانش ابری (Kubernetes) را ارائه می‌کند؛ بنابراین محصول MLOps باید با این زیرساخت سازگار بوده و از سرویس‌های ارائه شده توسط آن استفاده نماید.

* 1. دیتا پلتفرم

این بستر از سه بخش اصلی زیر تشکیل شده است:

* دریاچه داده: این بخش وظیفه ارائه بستری برای ذخیره‌سازی داده با ویژگی‌های مختلف و برای استفاده‌های متفاوت را بر عهده دارد.
* پردازش جریانی: این بستر امکان پردازش جریانی بر روی داده با امکان تعامل با دریاچه داده را در اختیار کاربر قرار می‌دهد.
* پردازش دسته‌‌ای: این بستر امکان پردازش دسته‌ای بر روی داده ذخیره شده در دریاچه داده را بر عهده دارد.

سازگاری و تجمیع محصول زیرساخت MLOps با این پلتفرم اهمیت بسیار بالایی دارد و در توسعه آن دقت به این موضوع نقش کلیدی برای کارفرما دارد.

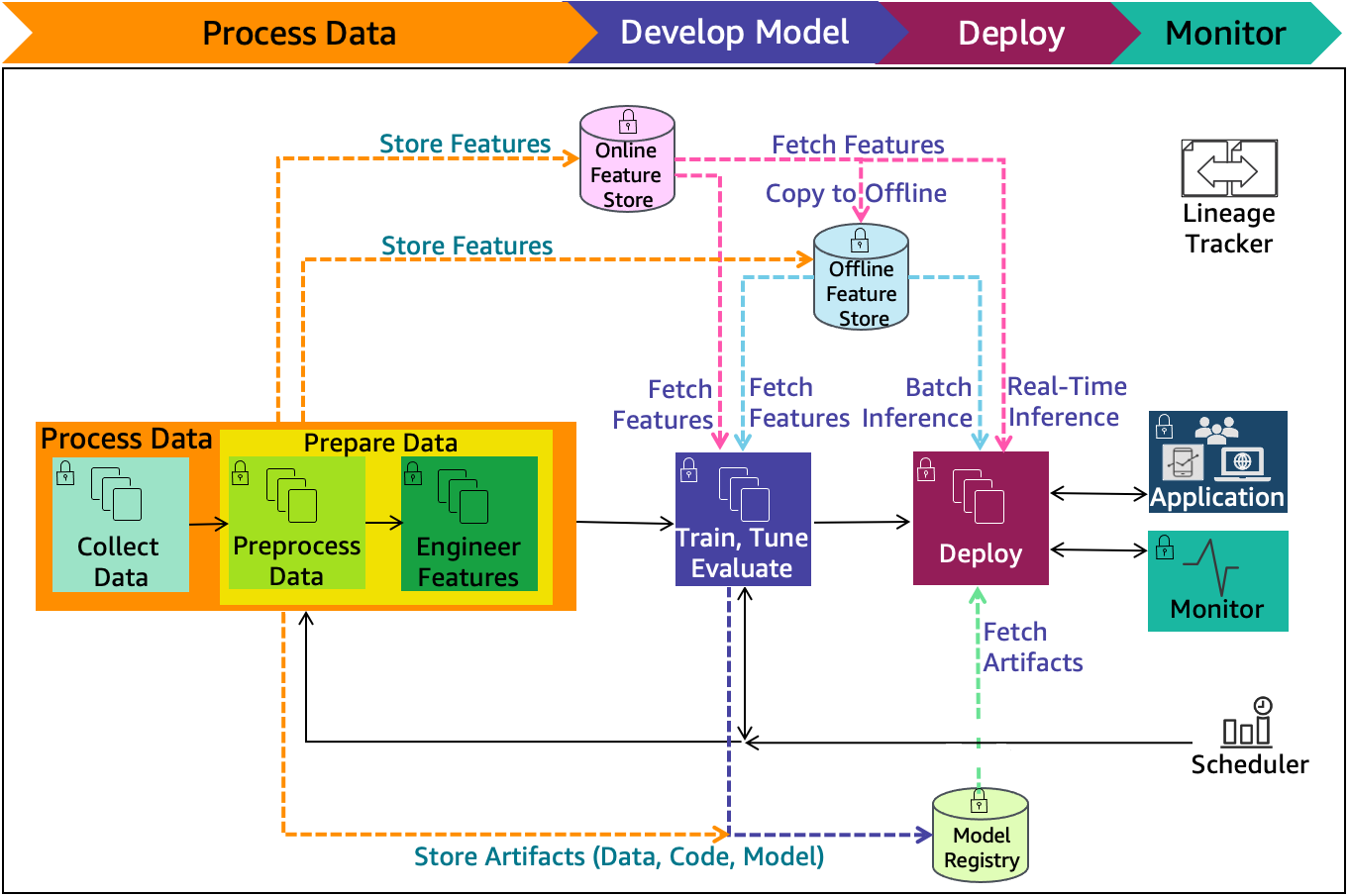
* 1. معماری محصول MLOps

محصولات مبتنی بر یادگیری ماشین چرخه متفاوتی را نسبت به دیگر محصولات نرم‌افزاری طی می‌کنند و یکی از دلایل استفاده از MLOps برای این محصولات (به‌جای DevOps معمول) همین تفاوت‌ها هستند. در ادامه فرایند تولید پروژه‌های مبتنی بر یادگیری ماشین را مرور می‌کنیم. مرحله‌های مختلف این فرایند در شکل زیر نشان‌داده‌شده است.

مرحله‌های مختلف چرخه حیات پروژه‌های یادگیری ماشین

در ابتدا، اهداف کسب‌وکاری پروژه مشخص ‌شده و سپس چارچوب مسئله یادگیری ماشین مشخص می‌شود. در ادامه فرایند مراحل نرم‌افزاری پروژه آغاز می‌شود، مرحله اول جمع‌آوری داده‌ها است. موفقیت پروژه‌های یادگیری ماشین به جمع‌آوری داده‌های مناسب وابسته است، مهم‌تر از تعدد این داده‌ها، کیفیت آن‌ها است که می‌تواند دقت خروجی مدل هوش مصنوعی را بالا ببرد. همچنین این داده‌ها باید به‌روز باشند و برای همین جمع‌آوری داده در بازه‌های زمانی تکرار می‌شود، سپس پردازش اولیه روی این داده‌ها انجام می‌شود. در ادامه به یکی از مهم‌ترین مراحل یعنی فرایند یادگیری مدل می‌رسیم که زمان‌برترین مرحله این چرخه است. باتوجه‌به این‌که ممکن است برخی فرا پارامترها نیز نیاز به تنظیم داشته باشند، فرایند یادگیری مدل می‌تواند چندین بار و برای فرا پارامترهای مختلف انجام شود. همچنین تست مدل پس از یادگیری نیز در این مرحله انجام می‌شود. پس از این، مدل آماده خدمت‌رسانی است.

باتوجه‌به بار استفاده از مدل، ممکن است لازم باشد مدل در چندین سرور استقرار پیدا کند و در طول زمان تعداد این سرورها تغییر یابد. مرحله نهایی این فرایند دیده‌بانی است که با بقیه پروژه‌های نرم‌افزاری از لحاظ ماهیت تفاوت دارد، در پروژه‌های معمول نرم‌افزاری پس از تولید، محصول محدودیت زمانی برای استفاده ندارد. در مقابل، در پروژه‌های یادگیری ماشین، با گذر زمان کیفیت محصول می‌تواند با تولید داده‌های جدید کاهش پیدا کند و توزیع داده‌های تولید شده و یا حتی تابع یادگیری تغییر کند؛ بنابراین کیفیت محصول باید به‌صورت مداوم دیده‌بانی شود و در زمان‌های مشخص یادگیری مدل با داده‌های جدید تکرار شود. این موضوع که continuous training نامیده می‌شود، در کنار CI/CD قرار می‌گیرد و بنابراین در پروژه‌های یادگیری ماشین فرایند CI/CD/CT موردنیاز است. پایپ لاین‌های آماده‌سازی داده، ایجاد و ذخیره فیچرهای آنلاین و آفلاین، استنتاج، تجمیع، تحویل و یادگیری مستمر (CI/CD/CT)، و یادگیری مجدد مدل بر روی معماری که در ادامه شرح داده شده، مستقر شده است.



معماری بستر MLOps

مؤلفه‌های این معماری در ادامه شرح داده شده است:

* پردازش داده(process data): در این مؤلفه کتابخانه‌هایی در جهت استخراج داده از منابع مختلف، استخراج مشخصات آماری داده، تمیزسازی داده و ایجاد فیچرها ارائه می‌شود.
* پردازش داده (process data): در این مؤلفه کتابخانه‌هایی در جهت استخراج داده از منابع مختلف، استخراج مشخصات آماری داده، تمیزسازی داده و ایجاد فیچرها ارائه می‌شود. برای این منظور ابزارهایی مانند Spark و کتابخانه‌های مرسوم از جمله TensorFlow SciKit, pandas, PyTorch, NumPy, Keras, در این پلتفرم موردنیاز است.
* انباره داده آنلاین/آفلاین فیچرها (online/offline store): با ذخیره‌کردن فیچرها، محاسبات تکراری آن‌ها در بخش‌های مختلف سازمان حذف می‌شود. انباره داده آنلاین در جهت دریافت سریع فیچرها برای استفاده در مرحله استنتاج به کار می‌رود. انباره داده آفلاین، تاریخچه مقادیر فیچرها را نگهداری می‌کند و در مرحله یادگیری مدل مورداستفاده قرار می‌گیرد. باتوجه‌به حجم و ابعاد داده امکان استفاده از انباره داده‌های توزیع شده نیز در این معماری وجود دارد. مورد انتظار است که برای این مؤلفه از ابزارهای موجود در دریاچه داده؛ مانند ‌ Minio , ClickHouse, Redis, HDFS استفاده شود و در صورت نیاز به ابزاری که توسط این زیرساخت ارائه نمی‌شود برای انتخاب آن نیاز به هماهنگی با تیم کارفرما وجود دارد.
* توسعه مدل (train/tune/evaluate): این مؤلفه بستری برای توسعه مدل از طریق آزمایش‌ها مختلف را فراهم می‌کند. این مؤلفه شامل نوت‌بوک‌ها و کتابخانه‌های مرسوم برای توسعه و ارزیابی مدل و کتابخانه‌هایی برای تنظیم پارامترها می‌باشد. همچنین بستری برای مدیریت فرایند توسعه مدل که شامل گام‌های train/tune/evaluate است، را فراهم می‌کند.
* رجیستری مدل (model registry): رجیستری مدل یک مخزن برای ذخیره مدل‌های یادگیری ماشین و فراداده‌های مرتبط است. در این رجیستری، نسخه‌های مختلف داده و مدل قرار می‌گیرد.
* مانیتورینگ (monitoring): این مؤلفه وظیفه نظارت بر مدل و تشخیص مشکلات را بر عهده دارد. مشکلات می‌تواند مرتبط باکیفیت داده، کیفیت مدل و دریفت باشد.
* استقرار (deploy): این مؤلفه وظیفه استقرار مدل در محیط عملیاتی و سرو آن را بر عهده دارد. این مؤلفه عملیاتی مانند مقیاس‌پذیری متناسب با بار ورودی و استقرار قناری را پشتیبانی می‌کند.
* زمان‌بند (scheduler): این مؤلفه می‌تواند فرایند یادگیری مجدد را در بازه‌های زمانی مشخص اجرا کند.
* ردیاب جامع (lineage tracker): این مؤلفه بستری برای ثبت و ضبط اطلاعات تمام منابع در هر نقطه از زمان را ارائه می‌کند. این اطلاعات می‌تواند شامل نسخه کد، داده، فیچرها، مدل و نتایج مربوطه باشد.

لازم به ذکر است که باتوجه‌به این‌که داده‌های استفاده شده در بسیاری از پروژه‌ها حساس و یا دارای ارزش تجاری بالایی هستند، لازم است امنیت این داده‌ها تأمین شود.

1. نیازمندی

در این بخش به بیان نیازمندی‌های عملکردی و غیرعملکردی که از بستر MLOps موردنیاز است، خواهیم پرداخت.

* 1. نیازمندی کارکردی

پلتفرم موردنظر امکان ارائه قابلیت‌های کلیدی زیر را داشته باشد.

* + 1. بستر توسعه مدل
* ارائه ابزار، محیط و استانداردهای مناسب برای توسعه مدل
* سیستم نسخه‌گذاری برای ذخیره‌سازی، دنبال‌کردن و نسخه‌گذاری کدهای یادگیری ماشین
* ارائه نوت‌بوک‌های تعاملی برای توسعه مدل
* امکان یکپارچگی با مخازن کنترل کد‌ (Git): امکان نگهداری کد توسعه داده شده مدل‌های یادگیری ماشین در سیستم‌های کنترل نسخه
  + 1. مدیریت داده‌
* Data gathering: امکان اتصال و جمع‌آوری داده از منابع داده‌ایی مختلف
* Data transformation/preparation: امکان پیش‌پردازش، تغییر و پاک‌سازی داده
* قابلیت نسخه‌گذاری داده
* Exploratory data analysis (EDA)
* امکان ایجاد پایپ لاین و استخراج featureهای موردنیاز از داده ورودی
* قابلیت Feature Engineering
* قابلیت ‌ Online Feature Store
* قابلیت ‌ Online Feature Store
* قابلیت کنترل دسترسی به دادگان
  + 1. مدیریت مدل
* Model training and tuning: ایجاد قابلیت یارگیری و بهینه‌سازی مدل‌های یادگیر ماشین به امکان پشتیبانی از کتابخانه‌های معمول یادگیری ماشین از جمله TensorFlow، PyTorch، SciKit، pandas، PyTorch، NumPy، Keras
* نسخه‌گذاری مدل: ارائه یک مکانیزم و راهکار برای مدیریت و نسخه‌گذاری مدل‌های یادگیری ماشین
* استقرا مدل: قابلیت ایجاد و مدیریت پایپ لاین ‌ CI/CD/CT به‌منظور استقرار خودکار مدل
* مانیتورینگ مدل: پایش مستمر کارایی مدل مستقر شده در محیط عملیاتی
* قابلیت خودکارسازی فرایند یادگیری و استقرار مدل: این قابلیت امکان اجرای مکرر و خودکار چرخه یادگیری و استقرار مدل را فراهم می‌کند.
* قابلیت تنظیم هایپرپارامترها: ایجاد مجموعه‌ای از ابزارها برای خودکارسازی فرایند تنظیم پارامترها به‌منظور افزایش کیفیت و سرعت در بهینه‌سازی مدل
* Automated model retraining: ارائه قابلیت یادگیری خودکار مدل مبتنی بر هشدارهای سیستم مانیتورینگ
  1. نیازمندی غیرکارکردی

در زیر قابلیت‌های غیرعملکردی مورد انتظار از این پلتفرم بیان شده است.

* مقیاس‌پذیری: مورد انتظار است که معماری این پلتفرم مقیاس‌پذیر طراحی شود که در صورت تأمین منابع ‌سخت‌افزاری، قابلیت مقیاس‌پذیری در پشتیانی و مدیریت جریان‌های کاری مربوط به مدل‌های‌ یادگیری ماشین و همچنین مقیاس‌پذیری در زمینه مدیریت حجم داده کلان و پردازش آن را فراهم نماید.
* امنیت: در این پلتفرم امنیت موارد مختلف مانند موارد زیر موردنیاز است.
  + احراز هویت و کنترل دسترسی به جریان کاری یادگیری ماشین
  + احراز هویت و کنترل دسترسی به دادگان
  + امکان رمزگذاری داده‌ها در ارتباطات
* Multi-Tenancy: امکان به‌کارگیری این پلتفرم توسط کاربرها و سازمان‌های متفاوت به‌صورت هم‌زمان و در نتیجه ارائه قابلیت Isolation برای کاربران مختلف در این پلتفرم موردنیاز است.
* قابلیت اطمینان و دسترس‌پذیری: در طراحی و توسعه این پلتفرم ارائه قابلیت تحمل خطا و امکان خودترمیمی در تمامی مولفه‌های پلتفرم برای دستیابی به قابلیت اطمینان و دسترس‌پذیری در نظر گرفته شود.
* یکپارچگی با بستر ابری سحاب: یکپارچگی این پلتفرم با بستر ابری سحاب و ابزارهای موجود در آن مانند ‌کوبرنیتیز، مخازن‌داده و ابزارهای مانیتورینگ و مدیریت لاگ
* مانیتورینگ و پایش: جمع‌آوری متریک و پایش وضعیت سیستم و عملکرد مولفه‌های مختلف و همچنین عملکرد جریان‌های کاری توسعه و استقرار مدل‌های یادگیری ماشین و اعلام هشدار در مواقع لازم
* مدیریت لاگ: جمع‌آوری، تبدیل و ارائه لاگ‌های عملکردی مولفه‌های سامانه به‌صورت متمرکز

1. فرایند توسعه محصول

موردنیاز است که تست سیستم به‌صورت تست‌های داخلی اتوماتیک و تست آلفا - بتا در بستر تست انجام شود و پس از موفقیت در بستر اصلی انتشار یابد. در زمینه ارزیابی ویژگی‌های کارکردی، فرایند تست مقبولیت نیز مورداستفاده قرار می‌گیرد. پیمانکار لیست موارد تست‌ را در قالب یک سند به کارفرما ارائه ‌می‌دهد، تا پس از توافق به‌عنوان مبنای کار در زمینه سنجش ویژگی‌های کارکردی قرار گیرد. هر مورد تست، مجموعه‌ای از ویژگی‌های کارکردی و غیرکارکردی را هدف قرار داده و بیانگر یک یا چند سناریو بوده که بر روی سامانه نهایی اجرا می‌شود. تأیید هر مورد تست به منزله تأیید ویژگی‌های کارکردی مربوطه است. در ادامه در ابتدا فرایند توسعه محصول موردنیاز برای تولید این محصول و کاربرد آزمایشگاه بیان خواهد شد.

فرایند توسعه محصول

پس از نصب یک نسخه خاص از محصول در محیط عملیاتی، فرایند به‌روزرسانی (شامل رفع باگ، ایجاد تغییرات زیرساختی و ارائه ویژگی جدید و...) از ایجاد یک تغییر CL[[1]](#footnote-2) شروع شده و در نهایت این CL در محیط عملیاتی نصب می‌گردد (البته قبل از نصب کلاستر عملیاتی نیز این فرایند وجود دارد و تنها مرحله آخر آن حذف می‌گردد). فرایند زیر برای اعمال یک CL صورت می‌گیرد:

* CL پس از ایجاد در کدبیس توسط یک یا چند توسعه‌دهنده در طی یک فرایند iterative مرور[[2]](#footnote-3) شده و در فرایند بازبینی از ابزارهای بررسی کیفیت کد مانند sonar نیز برای خودکارسازی بخشی از فرایند بازبینی استفاده می‌شود. در نهایت پس از تأیید CL در کدبیس ادغام می‌گردد. یک CL می‌تواند علاوه بر کد، شامل تغییرات لازم در کلاستر عملیاتی به‌صورت اسکریپت‌های مهاجرت داده، تغییر یا نصب مؤلفه در زیرساخت و... باشد.
* پس از ادغام CL، فرایند استقرار با build محصول و ساخت artifactهای پروژه آغاز می‌شود. این artifactهای ساخته شده بر روی Repository ذخیره می‌شوند.
* محصول در یک محیط کوچک (عموماً به‌صورت ماشین مجازی) نصب شده و آزمون‌های پذیرش بر روی آن اجرا می‌شود. درصورتی‌که هر یک از آزمون‌ها با خطا مواجه شوند، CL مربوطه در پایپلاین نصب به مرحلهٔ بعدی نمی‌رود.
* درصورتی‌که مرحلۀ قبل با موفقیت صورت گیرد، CL در محیط Staging نصب می‌شود و در صورت موفقیت­آمیز بودن این فرایند، CL در لیست CLهای موفق برای نصب در محیط عملیاتی قرار می‌گیرد. محیط Staging یک محیط شبیه به محیط عملیاتی (ترجیحاً با نوع سخت­افزار یکسان) می‌باشد که اندازه آن کوچک­تر از محیط عملیاتی بوده و غالباً این اندازه می‌بایست نسبت معقولی با محیط عملیاتی داشته باشد. هدف از وجود این محیط، تست CLها بر روی محیط مشابه، قبل از اجرا بر روی محیط عملیاتی به هدف کاهش خطا و ریسک تغییر است.
* CLهایی که پایپ لاین ۱ تا ۴ را با موفقیت گذرانده‌اند، در محیط عملیاتی اعمال می‌شوند.
* اطلاعات و آمار در مورد کیفیت توسعه محصول که در این فرایند ایجاد می‌شود برای شناخت بهتر از بلوغ محصول جمع‌آوری می‌شود. این اطلاعات برای شناخت مواردی مانند کیفیت توسعه، نتایج تست‌های مختلف انجام­شده، نتایج فرایندهای خودکار تولید و نصب سیستم و... جمع‌آوری می‌شوند که شامل شاخص‌های مختلفی مانند میزان پوشش تست‌ها، میزان موفقیت و شکست تست‌های مختلف، میزان موفقیت و شکست استقرار خودکار محصول و... هستند.
  + 1. آزمایشگاه

محیط Staging به‌منظور تست برخی ویژگی‌های غیرکارکردی همچون مقیاس‌پذیری، قابلیت اطمینان و کارایی است که امکان انجام خودکار آن‌ها وجود دارد. پس از توافق پیمانکار و کارفرما، در مورد ویژگی‌های قابل تست خودکار، روال‌های مناسب توسط پیمانکار توسعه داده شده و در اختیار کارفرما قرار می‌گیرد.

نکته­ای مهم این است که این محیط، تا زمانی که پروژه زنده است و تغییرات، رفع باگ و... در آن انجام می‌شود، ‌ باید حفظ و نگهداری شود و در حقیقت بخشی از تجهیزات اصلی پروژه بوده و نمی‌توان به آن نگاه گذرا و کوتاه­مدت داشت. تأمین سخت­افزار آزمایشگاه برعهده کارفرما است و پیمانکار وظیفهٔ راه­اندازی سامانه آزمایشگاهی را برعهده دارد.

* + 1. طرح آزمون پروژه

طرح آزمون پروژه، به‌منظور ایجاد سنجهٔ مورد توافق بین پیمانکار و کارفرما در خصوص انطباق محصول نهایی با تعهدات اولیه تدوین می‌گردد. ازآنجاکه این طرح با نیازمندی‌های مندرج در اسناد مرجع مطابقت داده شده است، در آن از ورود به جزئیات پیاده‌سازی خودداری شده و تمرکز بر روی نیازمندی‌های کارفرما است. این آزمون‌ها، هم نیازمندی‌های عملکردی و هم نیازمندی‌های غیرعملکردی (کیفی) را شامل می‌شوند.

تست نهایی که توسط پیمانکار به کارفرما ارائه خواهد شد، جدا از تست‌های Unit Test، Regression Test و Integration Test خواهد بود. در واقع در این نوع تست که به‌عنوان Acceptance Test شناخته می­شوند، ویژگی‌های نهاییFunctional و Non-Functional که مدنظر کارفرما است، تست خواهد شد. هر آزمون در قالب جدول زیر ارائه شود.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **کد آزمون** | |  | **عنوان آزمون** | |  |
| **نوع آزمون** | |  | **بند متناظر در سند مرجع (RFP/SoC/Requirements)** | |  |
| **شرح نیازمندی در سند مرجع (RFP/SoC/Requirements)** | | |  | | |
| **اهمیت نیازمندی** |  | | **تأثیر نیازمندی** |  | |
| **پارامترهای مورد سنجش در آزمون** | | |  | | |
| **نقاطی که در آزمون باید سنجیده شوند (Watchpoints)** | | |  | | |
| **سناریوی آزمون** | | |  | | |
| **ابزارهای آزمون** | | |  | | |
| **پیش‌شرط‌های آزمون** | | |  | | |
| **پس‌شرط‌های آزمون** | | |  | | |
| **شرط موفقیت کامل** | | |  | | |
| **شرط موفقیت نسبی** | | |  | | |
| **شرط شکست** | | |  | | |

قالب طرح آزمون پروژه

1. مدیریت پروژه

پیشنهاددهندگان موظف هستند در هنگام ارائه پیشنهاد، پروپوزال پروژه را طبق موارد درخواستی در این سند ارائه دهند.

* 1. برآورد زمانی

مدت‌زمان پیشنهادی این پروژه با احتساب پیاده‌سازی امکانات تکمیلی پلتفرم، 12 ماه از زمان عقد قرارداد است. پیمانکار موظف است در طول انجام پروژه گزارش پیشرفت پروژه را در قالب درخواستی کارفرما در بازه‌های زمانی دو هفته یکبار بر اساس برنامه زمان‌بندی ارائه کند

* 1. فازبندی

نیاز است که اجرای پروژه فازبندی شده و مدت هر فاز بیش از سه ماه نباشد. برای شفاف‌شدن هر فاز پیشنهادی مواردی موردنیاز است که در زیر مشخص شده است.

* فعالیت‌های موجود در هر فاز
* شرح هر فعالیت
* پیش‌نیاز هر فعالیت در صورت نیاز
* موارد تحویل‌دادنی هر فاز
* بستر تحویل مؤلفه یا قابلیت سامانه

پیمانکار موظف است ساختار شکست (WBS) و ریزفعالیتهای مطرح در هر فاز (گانت چارت) را به طور دقیق در پیشنهاد خود ارائه نماید. همچنین موارد قابل‌تحویل و محیط مورداستفاده برای تحویل مؤلفه و قابلیت‌های توسعه داده شده به تفکیک هر فاز در پیشنهاد خود اعلام نماید.

پیوست ۱: نیازمندی‌ها فنی در توسعه و استقرار محصول

1. پروژه در دو محیط آزمایشگاه و عملیاتی مستقر خواهد شد. به‌منظور اینکه تیم پیمانکار بتواند استقلال موردنظر خود را داشته باشد، می‌تواند محیط آزمایشگاهی علاوه بر این دو محیط راه‌اندازی نماید که شامل مولفه‌های زیرساختی است و نصب و راه‌اندازی پروژه در آن محیط آزمایشگاه کاملاً بر عهده تیم پیمانکار است. در محیط عملیاتی و آزمایشگاه کارفرما، زیرساخت ابری پروژه شامل کوبرنیتیز، پایگاه‌داده‌ها و ابزارهای زیرساختی مانند ابزار مانیتورینگ و مدیریت لاگ توسط تیم کارفرما نصب و راه‌اندازی خواهد شد؛ بنابراین فرایند نصب و اعمال تمامی مولفه‌های محصول MLOps، پلاگین‌ها، تنظیمات باید قابل‌تکرار در محیط‌های مختلف بوده و بستری برای نصب مجدد پروژه در محیط دلخواه ارائه شود تا بتوان پروژه را روی محیط عملیاتی نیز راه‌اندازی نمود.
2. تمامی کدهای پروژه باید از ابتدا در یک codebase که توسط تیم کارفرما نیز قابل‌دسترسی است، قرار گیرند. تمامی مولفه‌های توسعه داده شده، تنظیمات صورت‌گرفته، پلاگین‌های توسعه داده شده، پایپ لاین‌های داده، و اسکریپت‌های نصب، راه‌اندازی، و نگهداری باید در codebase قرار گیرند.
3. تمامی مولفه‌های توسعه داده شده باید به‌صورت Docker قابل‌اجرا باشند. Dockerfileهای مربوطه و اسکریپت‌های Build باید در codebase قرار گیرد.
4. ابزارهای 3rd Party مورداستفاده باید با هماهنگی با تیم کارفرما انتخاب شوند. به‌عنوان‌مثال مدنظر است برای orchestration از ابزار Kubernetes و برای صف از ابزار Kafka استفاده شود.
5. مولفه‌های توسعه داده شده و تنظیمات صورت‌گرفته باید به‌نوعی باشند که بر روی زیرساخت ابری نیز قابل‌اعمال بوده و سازگار باشد.
6. لیست تغییرات بین هر دو نسخه اصلی ارائه شده و درصورتی‌که تغییر از یک نسخه به نسخه دیگر نیازمند migration است راهنما و اسکریپت لازم برای این تغییر ارائه شود.
7. تیم پیمانکار موظف به ارائه مستندات و آموزش کافی در مورد جزئیات مؤلفه‌ها و همچنین نصب، پیکربندی، راه‌اندازی، و نگهداری محصول می‌باشد.
8. در صورت نیاز، نماینده فنی کارفرما باید بتواند جلسات هفتگی با تیم پیمانکار داشته باشد تا در جریان نحوه و کیفیت پیشبرد اهداف قرار گیرد و از طرفی آموزش‌های موردنیاز را دریافت کند.
9. در صورت رخداد مشکلی در نصب و راه‌اندازی و یا Bug در زمان اجرا در محیط عملیاتی، تیم پیمانکار موظف به حل مشکل می‌باشد.
10. اصول کیفیت کد شامل Clean Code، Unit Testing و Documentation باید در codebase رعایت شود.
11. کدها، مؤلفه‌ها، آرتیفکت‌ها، تنظیمات و اسکریپت‌های نصب و راه‌اندازی باید طی جلساتی به نماینده کارفرما تحویل داده شده و انتقال دانش موردنظر نیز انجام پذیرد. موارد تحویلی باید از جهت دارابودن سطح کیفیت کافی ذکر شده در بندهای بالا، مورد تأیید کارفرما قرار گیرد.
12. تمامی مؤلفه‌ها باید به‌صورت کوبرنتیزی (از طریق هلم) انتشار یابند. سرویس‌های غیر کوبرنتیزی موردنیاز از طریق مذاکره و تخصیص زمان قابل فراهم‌سازی است.
13. پادها امکان دسترسی به دیسک‌های لوکال و یا اجرا با privilege بالا را ندارند و برای بحث ذخیره‌کردن state باید از api سرویس‌های گرانیتی مانند پایگاه‌های داده استفاده کنند.
14. تمامی objectهای کوبرنتیزی پروژه باید در namespace مشخص شده قرار گیرند. استفاده از taint و affinity با مذاکره قابل‌انجام است.
15. به‌منظور اتصال به سرویس‌های موجود در گرانیت، آدرس و credentials سرویس‌ها به‌صورت متنی در اختیار سرویس‌گیرنده قرار می‌گیرد.
16. برای مدیریت پیش‌نیازهای مؤلفه‌ها بر روی بستر گرانیت مانند ایجاد یا تغییر پایگاه‌های داده و جداول، می‌توان از CRDها و init-container های گرانیتی استفاده نمود.
17. مؤلفه‌ها باید از بستر مانیتورینگ (Grafana/Prometheus) و مدیریت لاگ (EFK)گرانیت استفاده کرده ،و Observability موردنیاز برای نگهداشت سامانه با SLA مشخص شامل آلرت‌های disaster و high و داشبوردهای مناسب را ایجاد کنند.
18. SPOF در سطح فرایند یا داده نباید وجود داشته باشد؛ لذا تمامی مؤلفه‌ها باید HA باشند و داده‌ها بر روی بسترهای با replication ذخیره‌سازی شوند. همچنین درصورتی‌که نیاز به فرایندهای مشخصی برای Disaster Recovery همچون پشتیبان‌گیری از داده‌های حیاتی وجود دارد، این فرایندها باید مستندسازی یا خودکار شوند.

پیوست ۲: جداول امکانات و قابلیت‌های پلتفرم

در این بخش امکانات پایه و تکمیلی پلتفرم به‌صورت دقیق مشخص شده است.

|  |  |
| --- | --- |
| شماره | قابلیت |
| ۱-۱ | ارائه ابزار، محیط و استاندادهای مناسب برای توسعه مدل |
| ۱-۲ | سیستم نسخه‌گذاری کد |
| ۱-۳ | ارائه نوت‌بوک‌های تعاملی به‌منظور توسعه مدل |
| ۱-۴ | امکان یکپارچگی با مخازن کنترل کد‌ (Git) |
| ۲-۱ | Data Gathering |
| ۲-۲ | Data transformation/preparation |
| ۲-۳ | قابلیت نسخه‌گذاری داده |
| ۲-۴ | Exploratory data analysis (EDA) |
| ۲-۵ | امکان ایجاد پایپ لاین و استخراج feature |
| ۲-۶ | Feature Engineering |
| ۲-۷ | Online Feature Store |
| ۲-۸ | Online Feature Store |
| ۲-۹ | کنترل دسترسی به دادگان |
| ۳-۱ | Model training and tuning |
| ۳-۲ | نسخه‌گذاری مدل |
| ۳-۳ | استقرا مدل |
| ۳-۴ | مانیتورینگ مدل |
| ۳-۵ | قابلیت خودکارسازی فرایند یادگیری و استقرار مدل |
| ۳-۶ | قابلیت تنظیم خودکار هایپرپارامترها |
| ۳-۷ | Automated model retraining |
| ۴-۱ | مقیاس‌پذیری |
| ۴-۲ | امنیت |
| ۴-۳ | Multi-Tenancy |
| ۴-۴ | قابلیت اطمینان و دسترس‌پذیری |
| ۴-۵ | یکپارچگی با بستر ابری سحاب |
| ۴-۶ | مانیتورینگ، پایش |
| ۴-۷ | مدیریت لاگ |

پیوست ۳: الزامات پیشنهادهای فنی

پیشنهاددهندگان موظف هستند در پیشنهاد خود معماری سامانه را به طور کامل ارائه دهند. همچنین ملزم به پاسخگویی به کلیه امکانات مطرح شده در این پیوست هستند. لازم است برای هر یک از امکانات مشخص کنند که چه سطحی از پشتیبانی را پوشش می‌دهند. در صورت نیاز می‌توانند توضیحاتی نیز ارائه کنند. پیشنهادها باید کامل باشند و کلیه موارد موردنیاز را مطابق با الزامات تعیین شده در این RFP و پیوست‌های آن شرح دهند.

بستر توسعه مدل:

|  |  |
| --- | --- |
|  | امکان ۱-۱: ارائه ابزار، محیط و استانداردهای مناسب برای توسعه مدل |
| کامل  جزیی  عدم پشتیبانی | میزان پشتیبانی |
| توضیحات | |

|  |  |
| --- | --- |
|  | امکان ۲-۱: نسخه‌گذاری کد |
| کامل  جزیی  عدم پشتیبانی | میزان پشتیبانی |
| توضیحات | |

|  |  |
| --- | --- |
|  | امکان ۳-۱: ارائه نوت‌بوک‌های تعاملی به‌منظور توسعه مدل |
| کامل  جزیی  عدم پشتیبانی | میزان پشتیبانی |
| توضیحات | |

|  |  |
| --- | --- |
|  | امکان ۴-۱: امکان یکپارچگی با مخازن کنترل کد‌ |
| کامل  جزیی  عدم پشتیبانی | میزان پشتیبانی |
| توضیحات | |

مدیریت داده:

|  |  |
| --- | --- |
|  | امکان ۱-۲: Data Gathering |
| کامل  جزیی  عدم پشتیبانی | میزان پشتیبانی |
| توضیحات | |

|  |  |
| --- | --- |
|  | امکان ۲-۲: Data transformation/preparation |
| کامل  جزیی  عدم پشتیبانی | میزان پشتیبانی |
| توضیحات | |

|  |  |
| --- | --- |
|  | امکان ۳-۲: قابلیت نسخه‌گذاری داده |
| کامل  جزیی  عدم پشتیبانی | میزان پشتیبانی |
| توضیحات | |

|  |  |
| --- | --- |
|  | امکان ۴-۲: Exploratory data analysis (EDA) |
| کامل  جزیی  عدم پشتیبانی | میزان پشتیبانی |
| توضیحات | |

|  |  |
| --- | --- |
|  | امکان ۵-۲: امکان ایجاد پایپ لاین و استخراج Feature |
| کامل  جزیی  عدم پشتیبانی | میزان پشتیبانی |
| توضیحات | |

|  |  |
| --- | --- |
|  | امکان ۶-۲: Feature Engineering |
| کامل  جزیی  عدم پشتیبانی | میزان پشتیبانی |
| توضیحات | |

|  |  |
| --- | --- |
|  | امکان ۷-۲: Online Feature Store |
| کامل  جزیی  عدم پشتیبانی | میزان پشتیبانی |
| توضیحات | |

|  |  |
| --- | --- |
|  | امکان ۸-۲: Offline Feature Store |
| کامل  جزیی  عدم پشتیبانی | میزان پشتیبانی |
| توضیحات | |

|  |  |
| --- | --- |
|  | امکان ۹-۲: کنترل دسترسی به دادگان |
| کامل  جزیی  عدم پشتیبانی | میزان پشتیبانی |
| توضیحات | |

مدیریت مدل:

|  |  |
| --- | --- |
|  | امکان ۱-۳: Model training and tuning |
| کامل  جزیی  عدم پشتیبانی | میزان پشتیبانی |
| توضیحات | |

|  |  |
| --- | --- |
|  | امکان ۲-۳: نسخه‌گذاری مدل |
| کامل  جزیی  عدم پشتیبانی | میزان پشتیبانی |
| توضیحات | |

|  |  |
| --- | --- |
|  | امکان ۳-۳: استقرا مدل |
| کامل  جزیی  عدم پشتیبانی | میزان پشتیبانی |
| توضیحات | |

|  |  |
| --- | --- |
|  | امکان ۴-۳: مانیتورینگ مدل |
| کامل  جزیی  عدم پشتیبانی | میزان پشتیبانی |
| توضیحات | |

|  |  |
| --- | --- |
|  | امکان ۵-۳: قابلیت خودکارسازی فرایند یادگیری و استقرار مدل |
| کامل  جزیی  عدم پشتیبانی | میزان پشتیبانی |
| توضیحات | |

|  |  |
| --- | --- |
|  | امکان ۶-۳: قابلیت تنظیم خودکار هایپرپارامترها |
| کامل  جزیی  عدم پشتیبانی | میزان پشتیبانی |
| توضیحات | |

|  |  |
| --- | --- |
|  | امکان ۷-۳: Automated model retraining |
| کامل  جزیی  عدم پشتیبانی | میزان پشتیبانی |
| توضیحات | |

نیازمندی غیرکارکردی:

|  |  |
| --- | --- |
|  | امکان ۱-۴: مقیاس‌پذیری |
| کامل  جزیی  عدم پشتیبانی | میزان پشتیبانی |
| توضیحات | |

|  |  |
| --- | --- |
|  | امکان ۲-۴: امنیت |
| کامل  جزیی  عدم پشتیبانی | میزان پشتیبانی |
| توضیحات | |

|  |  |
| --- | --- |
|  | امکان ۳-۴: Multi-Tenancy |
| کامل  جزیی  عدم پشتیبانی | میزان پشتیبانی |
| توضیحات | |

|  |  |
| --- | --- |
|  | امکان ۴-۴: قابلیت اطمینان و دسترس‌پذیری |
| کامل  جزیی  عدم پشتیبانی | میزان پشتیبانی |
| توضیحات | |

|  |  |
| --- | --- |
|  | امکان ۵-۴: یکپارچگی با بستر ابری سحاب |
| کامل  جزیی  عدم پشتیبانی | میزان پشتیبانی |
| توضیحات | |

|  |  |
| --- | --- |
|  | امکان ۶-۴: مانیتورینگ، پایش |
| کامل  جزیی  عدم پشتیبانی | میزان پشتیبانی |
| توضیحات | |

|  |  |
| --- | --- |
|  | امکان ۷-۴: مدیریت لاگ |
| کامل  جزیی  عدم پشتیبانی | میزان پشتیبانی |
| توضیحات | |

1. Change List [↑](#footnote-ref-2)
2. Review [↑](#footnote-ref-3)