الزامات فنی

نسخه 1.0

تاریخچه تغییرات

نسخه	تاریخ	شرح
1.0	۱۴۰۱/۱۲/۲۱	انتشار لیست الزامات فنی در سه دسته معماری، کیفیت و یک‌پارچه‌سازی و با دو ستون الزام و شرح الزام

فهرست مطالب

مقدمه

الزامات فنی در سه دسته «الزامات مرتبط با معماری نرم‌افزار»، «الزامات مرتبط با فرایند توسعه و کیفیت نرم‌افزار» و «الزامات مرتبط با یک‌پارچه‌سازی» ارائه شده‌اند.

- در ستون «الزام»: عنوان مختصری از الزام مورد نظر آورده شده است.
- در ستون «شرح الزام»: الزام مورد نظر شرح داده و سعی شده تا جای ممکن منظور و هدف مورد انتظار از آن الزام شفاف شود.

هم‌چنین، الزامات هر دسته در دو بخش «الزامات ضروری» و «الزامات پیشنهادی» ذکر شده است. الزامات ضروری به این معنی است در حالت کلی و برای اکثر پروژه‌ها و سامانه‌های نرم‌افزاری این الزام باید رعایت شود. الزامات پیشنهادی موارد غیرضروری اما مفیدی هستند اگر انجام شوند خوب و ارزشمند است و اگر انجام نشوند، ضربه‌ای به کیفیت پروژه نمی‌زنند.

الزامات معماری نرم‌افزار

الزامات ضروری

شفافیت معماری		
	الزام	**شرح الزام**
1	**شفافیت نیازمندی‌های کلان کارکردی**	نیازمندی‌های کارکردی که بر معماری مؤثر هستند، مشخص، شفاف و مستند شده باشند. لازم نیست تک‌تک موارد کاربرد ذکر شوند، بلکه نمونه‌ها یا دسته‌های اصلی موارد کاربرد که بر شناخت نیازمندی‌های کارکردی موثر هستند بیان شوند.
2	**شفافیت محدودیت‌های معماری**	محدودیت‌های کلان (constraints) به خوبی شناخته شده باشند و مستند باشند و مورد اتفاق نظر کارفرما باشند.
3	**شفافیت و کمّی‌سازی ویژگی‌های کیفی**	نیازمندی‌های غیرکارکردی و ویژگی‌های کیفی به خوبی شفاف و مستند شده باشد. معیارها و محدوده‌های مطلوب ویژگی‌های کیفی مشخص و مستند باشد. اعضای ارشد تیم به خوبی نسبت به محدوده مطلوب معیارهای ویژگی‌های کیفی آگاه باشند. ترجیحا موارد اصلی ویژگی‌های کیفی به صورت SLA شفاف شده باشد.

4	**وجود سند معماری بروز**	وجود یک سند معماری نرم‌افزار که روزآمد باشد و به تازگی ویرایش و بروزرسانی شده و قدیمی نباشد و اطلاعات آن مربوط به گذشته نباشد. بروزرسانی سند معماری همواره و به صورت مستمر در طول فازهای توسعه، بهره‌برداری و نگهداری نرم‌افزار ادامه یابد.
5	**گردش مناسب سند معماری و استفاده جاری در تیم**	سند معماری در میان توسعه‌دهندگان و اعضای تیم به خوبی به گردش درآمده باشد و اعضای تیم از وجود آن باخبر باشند و از آن استفاده کنند و در اصلاح و بروزرسانی آن مشارکت نمایند. شواهد کافی در این زمینه در اختیار کارفرما قرار گیرد (مثلا مشارکت افراد مختلف در تهیه و تغییر و خوانش سند معماری از طریق سامانه مدیریت دانش گزارش شود).
6	**قالب مناسب و کارآمد سند معماری**	• استفاده از یک قالب روزآمد، مناسب و کافی برای پوشش اهداف سند معماری. قالب‌های قدیمی و عدم استفاده از یک قالب مشخص، ضعف محسوب می‌شود. • در این زمینه، قالب مشخصی از طرف کارفرما تدوین شده است و تبعیت از این قالب الزامی است.
7	**پوشش نماهای اصلی در سند معماری**	نماهای اصلی معماری (مثل نماهای C4) و معماری استقرار به خوبی مستند شده باشند. در این نماها، مؤلفه‌های اصلی، معماری استقرار و نحوه تقسیم سامانه از دیدهای مختلف توصیف می‌شوند.
8	**پوشش نماهای فرعی در سند معماری**	نماهای فرعی مثل نمای داده و نمای واسط کاربری به خوبی مستند شده باشد.
9	**پرهیز از گنجاندن محتوای غیرضروری در سند معماری**	هدف از این الزام، حفظ ایجاز سند معماری و پرهیز از مستندسازی حجیم و غیرضروری است. هرچند سند معماری معمولا سندی نسبتا طولانی است، اما باید از طولانی‌تر شدن این سند جلوگیری کرد و این سند باید به موجزترین شکل ممکن نگارش و نگهداری شود. برخی از اشتباه‌های رایج در این زمینه عبارتند از: کپی کردن تصاویر معماری‌های رایج (مثل معماری لایه‌ای یا معماری میکروسرویس) از اینترنت در سند معماری، توضیح معماری زیرساخت‌ها و مؤلفه‌های آماده یا متن‌باز (که معماری آن‌ها در اینترنت موجود است و لازم نیست در سند معماری گنجانده شود)، توضیح و تشریح اصول و قواعد معماری (مثل معنای ویژگی‌های کیفی، مفهوم معماری لایه‌ای یا ...) و غیره.
10	**ارائه نمای معماری امنیت**	معماری امنیتی محصول نرم‌افزاری به خوبی مستند شده باشد. این معماری شامل نواحی مختلف (عمومی و داخلی)، نحوه کنترل دسترسی‌ها و نحوه استقرار مؤلفه‌های مختلف نرم‌افزاری در سرورهای مختلف است.

11	**شفافیت تصمیمات معماری**	تصمیمات کلان معماری به همراه دلیل تصمیمات (rational) مستند و موجود باشد. تاریخچه تصمیمات و دلیل تصمیمات کلان معماری مشخص و مستند باشد. معمولا در یک سامانه نرم‌افزاری، به سه تا بیست مورد تصمیم کلان معماری می‌توان اشاره کرد.
12	**وجود فهرست مدون ریسک‌های معماری**	ریسک‌های فنی اصلی از منظر معماری، وام‌های فنی و نقاط بهبود اصلی معماری، به خوبی شناسایی شده باشد، توسط اعضای تیم شناخته شده باشد و به خوبی مستند شده باشد. برنامه مناسب و معقولی برای مقابله با ریسک‌های فنی وجود داشته باشد.
سبک و فناوری‌های اصلی معماری		
	الزام	**شرح الزام**
13	**انتخاب زبان مناسب متناسب با معماری سامانه**	زبان مناسب انتخاب شده باشد که با اهداف معماری متناسب باشد. انتخاب زبان برنامه‌نویسی، با شرایط اکوسیستم نرم‌افزاری کشور و همچنین با ترجیحات فنی سازمان سازگار باشد. نکته: این نکته در معیارهای کیفیت نرم‌افزار هم مطرح شده بود، ولی در اینجا تناسب انتخاب زبان با معماری مد نظر است.
14	**انتخاب پلتفرم و فریم‌ورک‌های مناسب متناسب با معماری سامانه**	پلتفرم‌ها و فریم‌ورک‌های مناسبی انتخاب شده باشد که با شرایط پروژه و معماری کلان سازگار و متناسب باشد. نکته: این نکته در معیارهای کیفیت نرم‌افزار هم مطرح شده بود، ولی در اینجا تناسب انتخاب **پلتفرم** با معماری مد نظر است.
15	**ماژولار بودن معماری**	تقسیم مناسب معماری به زیرسیستم‌های منطقی و مناسب. امکان جداسازی و جایگزینی بعضی از ماژول‌ها که از نظر کسب‌وکار، فعالیت‌های نسبتا مستقلی انجام می‌دهند و از نظر منطقی امکان جایگزینی دارند.
16	**معماری لایه‌ای**	معماری نرم‌افزار به لایه‌های مستقل تقسیم شده باشد. به ویژه لایه واسط کاربری و لایه داده مجزا باشند. در موارد لازم لایه‌های دیگر مثل لایه سرویس هم مستقل شده باشند. امکان دسترسی هر لایه فقط به لایه زیرین میسر باشد.
17	**فناوری مناسب پایگاه‌داده**	مثل Oracle, SQL Server و سایر پایگاه‌داده‌های مورد ترجیح سازمان. در صورت استفاده از پایگاه‌های NoSQL دلایل و شواهد کافی درباره انتخاب پایگاه‌داده موجود باشد و به تایید کارفرما برسد.

18	**سبک معماری مناسب**	سبک کلان معماری (Architecture Style) مناسب و متناسب با شرایط پروژه باشد. سبک‌های نوین مثل میکروسرویس مورد ترجیح است.
19	**استفاده از موتورهای مناسب نرم‌افزاری**	در موارد لازم و مفید از موتور گردش کار، موتور قوانین و موتور گزارش‌ساز استفاده شود. در این زمینه‌ها، فناوری مناسب و متناسب با نیازمندی‌های پروژه انتخاب شده باشد.
20	**پرهیز از تولید مولفه‌های نرم‌افزاری که نمونه ساده یا متن‌باز مناسب دارند**	پرهیز از اختراع دوباره چرخ و استفاده از مولفه‌های باکیفیت و آماده (مثل کتابخانه‌ها و فریمورک‌های نرم‌افزاری) باعث بهبود معماری نرم‌افزار است. همچنین این رویکرد به بهره‌گیری از پیشرفت‌های محصولات آماده کمک می‌کند و بر نگهداشت‌پذیری سامانه هم موثر است.
21	**عدم نیاز به لایسنس برای مولفه‌های مورد استفاده**	اجزای آماده مورد استفاده در سامانه (اجزای Third Party از قبیل مولفه‌ها، کتابخانه‌ها، فریمورک‌ها، پکیج‌ها، پلاگین‌ها و غیره) نیاز به تهیه لایسنس نداشته باشند. تیم توسعه باید از رایگان بودن و متن‌باز بودن مولفه‌های مورد استفاده اطمینان حاصل کند و کارفرما مسئولیتی از این حیث نخواهد داشت. در موارد اضطراری و حیاتی، ممکن است کارفرما با متن‌باز نبودن و یا خریداری لایسنس توسط پیمانکار موافقت کند که این موارد، باید به تایید مکتوب کارفرما برسد.
دسترس‌پذیری		
	الزام	**شرح الزام**
22	**سازوکار مناسب پشتیبان‌گیری و بازیابی (Backup/Restore)**	• برای همه منابع سامانه (مثل پایگاه‌داده) پشتیبان‌گیری مناسب (backup) وجود داشته باشد. سازوکار پشتیبان‌گیری و بازیابی اطلاعات متناسب با نیازمندی‌های پروژه باشد. • مثلا نرخ پشتیبان‌گیری، مکانیزم پشتیبان‌گیری، میزان بروز بودن نسخه پشتیبان و همچنین سرعت بازیابی اطلاعات (restore) متناسب با شرایط و نیازمندی‌های پروژه باشد. • اگر هنوز به زمان عملیات نرسیده‌ایم و از سیستم بهره‌برداری نشده است، طرح پشتیبان‌گیری و بازیابی مستند شده باشد و به تایید کارفرما برسد . در این طرح به ابزارها و سازوکار و نقش‌ها (افراد) و روندها به خوبی اشاره شده باشد و موارد خودکار و دستی به تفکیک توصیف شده باشد. • برای هر پروژه، این نیازمندی‌ها به صورت عددی و کمّی ذکر شود: نرخ پشتیبان‌گیری، سرعت بازیابی. • در طرح پشتیبان‌گیری به مکانیزم انتقال بکاپ به محل ذخیره‌سازی غیر متصل به سایت مثل tapeهای آفلاین اشاره شود.

23	**امکان افزونگی سرورها در محیط عملیات**	• در معماری نرم‌افزار به خوبی از افزونگی (Redundancy) سرورها حمایت و پشتیبانی شود تا در موارد بروز خطای نرم‌افزاری و سخت‌افزاری، سرورهای افزونه بتوانند جایگزین سرورهای معیوب شوند. • شواهد، آزمایش‌ها و گزارش‌های کافی درباره حمایت مناسب از افزونگی در اختیار کارفرما قرار گیرد. • این نیازمندی، هم ناظر بر شرایط عملیات است (در محیط عملیات سرورهای افزونه موجود باشد) و هم ناظر بر معماری نرم‌افزار است (در معماری نرم‌افزار، افزونگی سرورها ممکن باشد و در طراحی‌ها تمهیدات لازم دیده شده باشد)
24	**جایگزینی خودکار سرور افزونه در موارد بروز خطا**	جایگزینی سرور افزونه به جای سرور معیوب، به صورت خودکار باشد. در این زمینه سازوکار مناسبی تمهید شده باشد. سازوکار انتخاب‌شده برای این موضوع و ابزارهای مورد استفاده (مثلا کلاسترینگ اپلیکیشن‌سرورها و ابزار مورد استفاده برای کلاسترینگ) به کارفرما اعلام شود و تایید کارفرما دریافت شود.
25	**مکانیزم مناسب پشتیبان‌گیری و جایگزینی پایگاه‌داده**	به طور ویژه برای پایگاه‌داده، سازوکار مناسبی از منظر افزونگی و جایگزینی سرور تمهید شده باشد. سازوکار انتخاب‌شده برای این موضوع و ابزارهای مورد استفاده (مثلا SQL Server Clustering) به کارفرما اعلام شود و تایید کارفرما دریافت شود.
26	**محدودسازی نرخ فراخوانی**	به منظور افزایش دسترس‌پذیری، از تکنیک محدودسازی نرخ فراخوانی استفاده شود. نرخ فراخوانی (Rate limit) در بخش‌های مختلف (به ویژه نرخ فراخوانی سرویس‌ها و نرخ دریافت درخواست کاربران) محدود شود. مقدار محدودیت به اطلاع و تایید کارفرما برسد.
27	**محاسبه خودکار شاخص‌های دسترس‌پذیری**	امکان محاسبه خودکار شاخص‌های دسترس‌پذیری فراهم شود. امکان تفکیک دلایل قطعی (قطعی زیرساختی، قطعی شبکه، اختلال سخت‌افزاری، اختلال در سامانه‌های وابسته و در نهایت اختلال نرم‌افزاری سامانه) در گزارش‌ها فراهم شود. دسترسی به گزارش‌ها و داشبوردهای مربوطه به کارفرما داده شود.
28	**مجهز بودن همه پردازه‌ها به سرویس کنترل سلامت (Health Check)**	همه پردازه‌ها به ویژه سرویس‌های نرم‌افزاری مجهز به سرویسی باشند که با دسترسی به آن‌ها بتوان از سلامت خود آن‌ها و سرویس‌های مورد نیازشان مطلع شد.
29	**هشدار نقض دسترس‌پذیری به صورت خودکار**	هشدارهای لازم در موارد نقض دسترس‌پذیری به صورت خودکار تولید و ارسال شوند. بسترهای ارسال هشدار (ایمیل، پیامک و غیره) به تایید کارفرما برسد. دلیل و منشا نقض دسترس‌پذیری در هشدار ذکر شود.

30	**امکان ادامه فعالیت سیستم در صورت بروز مشکل فقط در بخشی از سیستم**	در موارد خرابی یا اختلالِ بخش‌هایی از سامانه و یا سامانه‌های وابسته، بخش‌هایی از سیستم که به موضوع اختلال وابسته نیستند بتوانند به فعالیت و سرویس‌دهی ادامه دهند.
31	**اطمینان از دسترس‌پذیری با آزمون سناریوهای لازم**	اطمینان از صحت تکنیک‌های حفظ دسترس‌پذیری با تست سناریوهای مختلف قطعی و اختلال انجام شده باشد. طرح آزمون و نتایج آزمون در اختیار کارفرما قرار گیرد. آزمون به وجود محیط عملیاتی موکول نشود و قبل از عملیات آزمون‌های مناسب طراحی و در حد بضاعت سخت‌افزاری و نرم‌افزاری موجود، اجرا شده و نتایج آن گزارش شود.
32	**امکان بازگشت سریع به نسخه قبلی در صورت نصب ناموفق**	در صورت نصب یا بروزرسانی ناموفق، بازیابی نسخه قبلی (rollback) به سرعت و با کمک ابزار و رویکرد مناسب ممکن باشد.
33	**بروزرسانی نرم‌افزار بدون توقف سرویس**	بروزرسانی نرم‌افزار و نصب نسخه‌های جدید، بدون نیاز به توقف سرویس‌دهی ممکن باشد. از تکنیک‌های لازم (مثلا blue/green deployment) استفاده شود. تکنیک‌های مورد استفاده به اطلاع کارفرما رسانده شود.
34	**ساز و کار Disaster Recovery**	در طراحی سامانه ساز و کارهای لازم برای نصب همزمان در چند مرکز داده به صورت Active/Passive یا Active/Active وجود داشته باشد.

کارایی

	الزام	شرح الزام
35	**استفاده موثر از Cache در لایه داده**	• استفاده مناسب از فناوری‌های cache همانند Redis • استفاده از سازوکار مناسب برای همگام‌سازی و بروزرسانی cache
36	**استفاده از ابزار و رویکرد مناسب برای تولید گزارش از داده‌ها**	در مواردی که گزارش (report) تولید می‌شود، از ابزار مناسب و طراحی مناسب لایه داده استفاده شود تا سرعت تولید گزارش بالا رود و سرعت سایر موارد کاربرد کاهش نیابد.
37	**وجود آزمون‌های اطمینان‌بخش کارایی سطوح و لایه‌های مختلف**	• برنامه آزمون به خوبی مدون شود و به تایید کارفرما برسد .آزمون‌های متنوع و کافی برای تست کارایی طراحی و آماده شده باشد. به ویژه به آزمون بار، آزمون استرس و آزمون استقامت توجه شود. • شرایط مناسب (داده‌های مناسب و روال مناسب) برای تست فراهم شود. • گزارش تست‌های کارایی به صورت دوره‌ای در اختیار کارفرما قرار گیرد. • انجام تست‌های کارایی به فازهای انتهایی و یا حصول شرایط ایده‌آل سخت‌افزاری موکول نشود و در همه فازهای توسعه، با سخت‌افزار و نرم‌افزار موجود آزمون‌های کارایی انجام شود.

38	**مصرف بهینه منابع**	میزان مصرف منابع به ویژه پردازنده، حافظه اصلی و پهنای باند شبکه بهینه باشد. نشت حافظه (Memory Leak) وجود نداشته باشد. دسترسی کارفرما به گزارش‌های اطمینان‌بخش خودکار در این زمینه فراهم شود.
39	**وجود گزارش‌های خودکار و آنلاین از شاخص‌های کارایی در محیط عملیات**	امکان پایش خودکار معیارهای کارایی در محیط عملیات وجود داشته باشد. گزارش‌های خودکار و آنلاین برای شاخص‌هایی مثل نرخ پاسخ موفق، نرخ خطا در پاسخ، سرعت پاسخ (میانگین و بیشینه)، میزان مصرف منابع و غیره وجود داشته باشد .امکان دسترسی کارفرما به داشبوردهای کارایی فراهم شود.
40	**بهینه‌سازی تنظیمات زیرساخت‌ها مثل وب‌سرور و پایگاه‌داده**	بهینه‌سازی مناسب در سطح Application Server و DBMS و سایر موارد زیرساختی انجام شود .اقدامات انجام‌شده و نتایج آن در اختیار کارفرما قرار گیرد و به صورت دوره‌ای تکرار شود.
41	**استفاده موثر از امکانات پایگاه‌داده برای ارتقای کارایی**	استفاده موثر از تکنیک‌هایی مثل Secondary Indexes یا Denormalization یا Materialized Views
42	**توزیع بار با هدف افزایش سرعت**	استفاده از Load Balancer با هدف ارتقای کارایی انجام شود. بهینه‌سازی Load Balancer برای بیشینه‌سازی کارایی انجام شود.
43	**تنظیم و بهینه‌سازی صف‌ها**	هر جا از صف (Queue) استفاده می‌شود، بهینه‌سازی‌ها و تنظیمات لازم انجام شود. به ویژه تعیین محدودیت در اندازه صف و تنظیم Time Out برای حضور در صف انجام شود.
44	**ایجاد محدودیت در نرخ ورود درخواست‌ها**	نرخ ورود درخواست‌ها در همه موارد محدودیت داشته باشد، به ویژه نرخ ورود درخواست کاربران، فراخوانی وب‌سرویس‌ها و سایر ورودی‌های سامانه. اندازه و مقدار محدودیت با هماهنگی و تایید کارفرما تعیین و اعمال شود.
45	**امکان دسته‌بندی درخواست‌ها و اولویت‌بندی درخواست‌های ورودی**	در اختصاص منابع و پردازش کارها، به کاربردهای مهم‌تر اولویت بالاتری داده شود. به خاطر موارد کاربرد فرعی‌تر، موارد کاربرد مهم‌تر کند نشوند. مثلا سرعت کار کاربران در زمان ارائه گزارش مدیریتی به شدت کاسته نشود و یا از کار نیافتد.
46	**تنظیم مهلت (Timeout) در همه موارد لازم**	برای پاسخ درخواست کاربر، فراخوانی وب‌سرویس‌ها، کوئری‌های پایگاه‌داده و سایر فراخوانی‌ها، تایم‌اوت تنظیم شود.
47	**استفاده مناسب از Resource Pooling در همه موارد لازم**	برای Connection و Thread و همه مواردی که Pooling مفید است، از این تکنیک استفاده شود.

48	**پایش و تحلیل کوئری‌های پایگاه‌داده**	تحلیل کوئری‌های پایگاه‌داده، بهینه‌سازی آن‌ها و ارائه دسترسی کارفرما به گزارش‌های مربوطه و اقدامات انجام‌شده در این زمینه.
49	**بهره‌گیری از Non-Blocking IO**	در مواردی که مستلزم تعامل پر فرکانس با شبکه، فایل و پایگاه‌داده است، جهت افزایش سرعت و کارایی از کتابخانه‌ها و تکنیک‌های Non-Blocking IO استفاده شود.
50	**پارتیشن‌بندی افقی یا عمودی داده‌ها در موارد مفید**	در مواردی که به کارایی کمک می‌کند، به ویژه برای جداول پایگاه‌داده که داده‌های بسیار زیادی را نگهداری می‌کنند، داده‌های پایگاه‌داده به پارتیشن‌های عمودی یا افقی یا هر دو تقسیم شوند.
51	**ایجاد پردازش موازی، هم‌روند یا توزیع‌شده در موارد لازم**	در مواردی که به کارایی کمک می‌کند، از پردازش موازی یا هم‌روند یا توزیع‌شده یا تکنیک‌هایی مثل Map-Reduce استفاده شود.
52	**زمان‌بندی مناسب پردازش‌ها**	در مواردی که به افزایش کارایی کمک می‌کند، زمان‌بندی (Scheduling) مناسب کارها و منابع انجام شود. مثلا موکول کردن کارهای پرحجم و قابل تعویق به نیمه‌شب‌ها و یا تعویق عملیات زمان‌بر در ساعات اوج ترافیک به زمانی دیگر.
53	**بهینه‌سازی برنامه‌ها در لایه کلاینت**	نمایش داده در سمت کلاینت سرعت و کارایی مناسب داشته باشد. استفاده از تکنیک‌های لازم برای افزایش کارایی لایه کلاینت، همانند: حذف کدهای مرده به منظور ارتقای سرعت کلاینت‌ساید، بهینه‌سازی و کاهش حجم محتوای کلاینت (HTML, JS)، کمینه کردن کتابخانه‌های کلاینت و حجم آن‌ها.

مقیاس‌پذیری		
	الزام	**شرح الزام**
54	**توزیع بار با هدف افزایش مقیاس‌پذیری**	از Load Balancer با هدف مقیاس‌پذیری استفاده شود. بهینه‌سازی Load Balancer برای بیشینه‌سازی کارایی در دستور کار قرار گیرد. قبل از عملیاتی شدن سامانه، Load Balancer در حالت آزمایشی راه‌اندازی شود و راه‌اندازی آن به فازهای پایانی و مراحل عملیاتی سامانه موکول نشود. گزارش‌های اطمینان‌بخش از صحت و کارایی Load Balancer ارائه شود و در اختیار کارفرما قرار گیرد.
55	**سازوکار مقیاس‌پذیر مدیریت نشست کاربران**	سازوکار مدیریت نشست کاربران، مقیاس‌پذیر باشد. ابزار و سازوکار مدیریت نشست کاربران (User Sessions) متناسب با نیازمندی‌های سامانه باشد. به عنوان مثال در صورت شلوغی یک سرور یا مسیردهی به یک سرور دیگر، نشست کاربر در سرور دیگری قابل ادامه باشد.

56	**بهره‌گیری از معماری Stateless به منظور افزایش مقیاس‌پذیری**	در موارد لازم، از طراحی سرورهای Stateless به منظور حداکثرسازی مقیاس‌پذیری استفاده شود.
57	**وجود آزمون‌های اطمینان‌بخش مقیاس‌پذیری**	• آزمون‌های متنوع و کافی برای تست مقیاس‌پذیری طراحی و آماده شده باشد. • شرایط مناسب (داده‌های مناسب و روال مناسب) برای تست فراهم شود. • گزارش این تست‌ها به صورت دوره‌ای در اختیار کارفرما قرار گیرد.
58	**امکان Scale-Up در همه سطوح و لایه‌ها**	در هیچ بخشی (پایگاه‌داده، وب‌سرور و غیره) محدودیتی برای افزایش منابع مثل پردازنده و حافظه (Scale-Up) نداشته باشیم.
59	**امکان Scale-Out در لایه اپلیکیشن‌سرورها**	امکان افزایش تعداد سرورها در لایه Application Servers وجود داشته باشد. تعداد سرورها در این لایه محدود نباشد. تکنیک‌های مورداستفاده در این زمینه (مثل Load balancer یا Clustering یا تکنیک‌های مشابه) به اطلاع و تایید کارفرما برسد.
60	**امکان Scale-Out در لایه پایگاه‌داده**	امکان افزایش تعداد سرورهای پایگاه‌داده وجود داشته باشد و سرور پایگاه‌داده به یک سرور محدود نباشد. تکنیک‌های مورد استفاده (مثل Database Clustering یا Distributed Databases یا NoSQL) به تایید کارفرما برسد.
61	**استقرار بر روی کانتینر در محیط عملیات**	به منظور افزایش مقیاس‌پذیری، استقرار در محیط عملیاتی بر روی کانتینرها (مثل Docker) انجام شود.
62	**مدیریت و تنظیم خودکار کانتینرها در محیط عملیات**	به منظور افزایش مقیاس‌پذیری و البته کمک به استقرار و مانیتورینگ بهتر، مدیریت و تنظیم کانتینرها به صورت خودکار و با ابزار مناسب (مثل Kubernetes) انجام شود.
63	**امکان استقرار در سرورهای مجزا برای مولفه‌هایی که منطقا مجزا هستند**	مولفه‌هایی که وابستگی مستقیم به هم ندارند، امکان جداسازی و توزیع در سرورهای مختلف داشته باشند. مثلا امکان جداسازی اپلیکیشن‌سرور از پایگاه‌داده و استقرار آن‌ها در سرورهای مجزا وجود داشته باشد. همچنین ماژول‌های مستقل نرم‌افزار، امکان جداسازی در سرورهای مختلف (و تعامل از طریق وب‌سرویس) داشته باشند.
64	**استقرار مقیاس‌پذیر**	استفاده از رویکرد و ابزارهای مناسب به منظور خودکارسازی و تسهیل نصب، بروزرسانی، تنظیم و تخصیص منابع بر روی سرورهای مختلف
نگه‌داشت‌پذیری		
	الزام	**شرح الزام**

65	**ریزدانگی مناسب و کافی ماژول‌ها**	ماژول‌های توسعه‌یافته به اندازه کافی ریزدانه باشند. ماژول‌های یک‌تکه و بزرگ معمولا مناسب نیستند. پروژه به تعدادی ماژول مشخص، مستقل و به اندازه کافی ریزدانه تقسیم شده باشد.
66	**پشتیبانی از چندزبانی و سهولت افزودن زبان‌های جدید**	در نرم‌افزارهایی که نیازمند پشتیبانی از چندزبانی (Multi-Language) هستند، این الزام وجود دارد. در این شرایط، از تکنیک‌های مناسب برای تسهیل افزودن زبان استفاده شود. اضافه کردن زبان با کمک افزایش فایل‌های متنی (فایل‌های ترجمه‌ها) ممکن باشد.
67	**رعایت معماری تمیز در ایجاد مرزها و جهت وابستگی‌ها**	رعایت اصول معماری تمیز .(Clean Architecture) بخش‌های مرکزی و اصلی‌تر معماری به فناوری‌های خاص و جزئیات خارج از کسب‌وکار (Details) وابسته نباشند، بلکه بخش‌های فرعی به بخش‌های اصلی وابسته باشند. امکان تغییر فناوری‌ها، با حداقل تغییر معماری ممکن باشد. مرزبندی بین ماژول‌ها با اصول معماری هم‌خوان باشد و جهت وابستگی‌ها از فرعی به اصلی (و نه برعکس) باشد.
68	**انسجام و استقلال نسبی مولفه‌ها**	وجود Cohesion بالا و حداقل Coupling بین مولفه‌ها.
69	**تناظر مولفه‌ها با قابلیت‌های کسب‌وکار**	وجود Bounded Contextهای مشخص و منطقی در تقسیم‌بندی مولفه‌های نرم‌افزار. مطابق با توصیه‌های رویکرد طراحی دامنه‌محور (Domain Driven Design)، هر مولفه متناظر با یک قابلیت کسب‌وکار (Business Capabilities) طراحی شده باشد.
70	**استفاده از رویکرد سرویس‌گرا یا میکروسرویس**	استفاده از سبک سرویس‌گرا یا میکروسرویس، برای افزایش قابلیت نگهداری نرم‌افزار. سبک میکروسرویس مورد ترجیح است. در مواردی که سبک میکروسرویس به طور کامل قابل پیاده‌سازی نیست، به‌روش‌های میکروسرویس تا حد امکان استفاده شود.
71	**امکان تنظیم برخی متغیرهای کسب‌وکار به صورت پویا توسط مدیر**	برخی از متغیرهای کسب‌وکار در زمان اجرا و توسط کاربر ادمین و از طریق واسط کاربری قابل تغییر باشند و تغییر آن‌ها نیازمند تغییر در متن برنامه‌ها و یا تغییر در فایل‌های استقرار (Configuration Files) یا تغییر در استقرار یا استقرار مجدد نباشد.
72	**پشتیبانی از موتور قوانین کسب‌وکار**	*نیازمندی‌های مربوط به موتور قوانین کسب‌وکار دقیق شود.*
73	**پشتیبانی از موتور گردش کار**	*نیازمندی‌های مربوط به موتور گردش کار دقیق شود.*
74	**پشتیبانی از فرم‌ساز**	*نیازمندی‌های مربوط به فرم‌ساز دقیق شود.*
75	**پشتیبانی از گزارش‌ساز**	*نیازمندی‌های مربوط به گزارش‌ساز دقیق شود.*

امنیت		
	الزام	**شرح الزام**
76	**ناحیه‌بندی مناسب سرورها و مولفه‌های نرم‌افزاری**	معماری امنیتی مناسبی طراحی شده باشد. نحوه تقسیم‌بندی سرورها به نواحی (Zones) مختلف و نحوه استقرار مولفه‌های نرم‌افزاری در سرورهای مختلف متناسب با دغدغه‌های امنیتی کارفرما باشد.
77	**تغییر تنظیمات پیش‌فرض زیرساخت‌های نرم‌افزاری**	تنظیمات پیش‌فرض مانند رمز عبور و پورت شبکه در زیرساخت‌هایی مثل پایگاه‌های داده، وب‌سرورها، صف‌ها و غیره تغییر کنند و مانند تنظیمات پیش‌فرض باقی نمانند.
78	**خودداری از ذخیره رمزها در متن کد**	ذخیره نکردن رمزها و سایر موارد محرمانه (مثل پورت یا آی‌پی) در متن برنامه‌ها.
79	**روال مدیریت، ذخیره و بازیابی مناسب برای رمز عبور کاربران**	رمزعبور کاربران به صورت رمزشده (Hash) ذخیره شود. اقدامات امنیتی لازم و کافی برای بازیابی رمز عبور در نظر گرفته شود. امکان اعمال نیازمندی‌های امنیتی در انتخاب رمز عبور کاربران وجود داشته باشد.
80	**نمایش ندادن پیغام خطای تکنیکال به کاربران**	در موارد بروز خطا، پیغام خطا که شامل نکات و جزئیات فنی (مثل متن پیغام Exception یا Stack-Trace) است به کاربر نمایش داده نشود.
81	**محدودیت زمان نشست‌های کاربری**	در صورت استفاده از Session، زمان زنده ماندن Sessionها محدود باشد. میزان محدودیت زمانی متناسب با نیازهای کسب‌وکار تعیین شود.
82	**ثبت لاگ مناسب برای رویدادهای مهم امنیتی**	برای رویدادهای مهمی مثل ورود و خروج کاربر، تلاش برای دسترسی به داده غیرمجاز و روند استفاده کاربر از سامانه، لاگ مناسب تهیه شود. امکان بررسی و گزارش این لاگ‌ها در شرایط لازم فراهم شود. شواهد و گزارش‌های لازم برای اطمینان از حصول این الزام به کارفرما ارائه شود.
83	**بررسی مستمر امنیتی متن برنامه‌ها با کمک ابزار تحلیل ایستای کد و کشف باگ‌های امنیتی**	معرفی ابزارهای استفاده‌شده و دریافت تایید کارفرما. ارائه گزارش‌های ارزیابی به صورت مستمر به کارفرما.
84	**استفاده از ضدویروس در بخش بارگذاری فایل‌ها**	جلوگیری از بارگذاری فایل‌های حاوی ویروس.
85	**استفاده از پروتکل‌ها، استانداردها و الگوریتم‌های مناسب رمزنگاری**	استفاده از الگوریتم‌های مناسب رمزنگاری و Hashing، استفاده از تکنیک‌های مرسوم MAC

86	**اعمال محدودیت در میزان دسترسی هر کاربر به منابع مختلف و تعداد عملیات منجر به خطا**	میزان دسترسی هر کاربر به منابع مختلف محدود شود. در صورت اشتباه کاربر (مثلا ورود رمز اشتباه) و یا انجام عملیات منجر به خرابی یا خطا (مثلا ایجاد گزارشی که به تایموت منجر می‌شود) توسط کاربر، تعداد انجام مجدد عملیات محدود شود.
87	**جلوگیری از حملات امنیتی معروف و شناخته‌شده**	با استفاده از چک‌لیست‌های امنیتی مناسب. ارائه گزارش بررسی‌ها، چک‌لیست‌ها و نتایج آن‌ها به کارفرما.
88	**جلوگیری از افشای داده‌های حساس**	برای داده‌های حساس، ملاحظات امنیتی مضاعفی در نظر گرفته شود (مثل رمزنگاری یا بررسی دسترسی کاربر از بیش از یک مکانیزم امنیتی) . تمهیدات اندیشیده‌شده به اطلاع کارفرما برسد و تایید کفایت آن‌ها از کارفرما دریافت شود.
89	**توجه به ریسک‌ها و راهکارهای مهم OWASP**	به ویژه ۱۰ ریسک برتر OWASP
90	**تغییر کلیدهای رمزنگاری به صورت دوره‌ای**	مثلا ماهیانه یک بار. دوره زمانی تغییر رمزها به تایید کارفرما برسد.
91	**استفاده از پروتکل‌های شناخته‌شده و مناسب برای برقراری ارتباط بین برنامه و با دیگر برنامه‌های خارج از محیط سازمان**	از پروتکل‌های مناسب همانند SSL و HTTPS در این زمینه استفاده شود.
92	**اعمال قوانین محدودکننده هنگام خروج داده به خارج از محصول**	به ویژه برای داده‌های مهم و محرمانه، برای اجرای روند خروج اطلاعات (اکسپورت) یا دریافت گزارش‌ها، ملاحظات امنیتی مضاعفی در نظر گرفته شود .تمهیدات اندیشیده‌شده به اطلاع کارفرما برسد و تایید کفایت آن‌ها از کارفرما دریافت شود.
93	**توانایی تشخیص تغییر غیرمجاز در داده‌های حساس ذخیره‌شده**	در موارد حساس و مهم، امکان تشخیص تغییر غیرمجاز وجود داشته باشد.
94	**دریافت گواهینامه افتا**	در مواردی که کارفرما لازم می‌داند.
95	**ارائه گزارش‌های تست‌های امنیتی**	آزمون‌های انجام‌شده و نتایج آن‌ها در اختیار کارفرما قرار گیرد.
معماری واسط کاربری		
	الزام	**شرح الزام**
96	**قابلیت واکنش‌گرایی صفحات**	صفحات واسط کاربری قابلیت واکنش‌گرایی (Responsiveness) داشته باشند.

97	**تست خودکار واسط کاربری**	آزمون‌های کافی و خودکار برای صحت‌سنجی واسط کاربری از طریق ابزارها و فناوری‌های مناسب انجام شود .نتایج آزمون‌ها در اختیار کارفرما قرار گیرد.
98	**وجود راهنماهای لازم برای کاربر حین تعامل با سامانه**	رویکرد مناسبی برای نمایش راهنما به کاربران اتخاذ شده باشد.
99	**توجه به تجربه کاربری و کاربرپسند بودن**	توجه کافی به User Friendly بودن و User Experience و ارائه شواهد اطمینان‌بخش به کارفرما درباره راهکارهای اتخاذشده در این زمینه و نتایج آن‌ها.
100	**استفاده از فناوری‌های بروز و استاندارد برای توسعه واسط کاربری**	پرهیز از استفاده از استانداردهای قدیمی و یا فناوری‌های منسوخ یا غیررایج در زمینه توسعه واسط کاربری.
101	**سرعت مناسب بارگذاری صفحات**	سرعت نمایش صفحات واسط کاربری مناسب باشد.
102	**اعتبارسنجی درخواست‌ها به صورت کلاینت‌ساید و سرورساید**	اعتبارسنجی درخواست‌ها (مثل درخواست کاربر و فرم‌های کاربری) هم به صورت کلاینت‌ساید و هم به صورت سرورساید
103	**توسعه نسخه موبایل**	در صورت صلاحدید کارفرما
104	**توسعه واسط وب به صورت PWA**	در صورت صلاحدید کارفرما
105	**دریافت منابع صرفا از طریق سرورهای سازمان**	تمام کتابخانه‌های Java Script، فایل‌های CSS، فونت‌ها و سایر منابع لازم برای انتقال به سمت کلاینت، از مسیر سرورهای اختصاصی سازمان بازیابی و دریافت شود و از سرور دیگری (مثل سرورهای عمومی اینترنتی) استفاده نشود. در غیر این صورت، پیمانکار مسئول کارکرد نادرست سامانه خواهد بود.

الزامات پیشنهادی

سبک و فناوری‌های اصلی معماری		
	الزام	**شرح الزام**
1	**وجود بنچمارک و آزمایش‌های کافی برای انتخاب فناوری‌ها**	انتخاب فناوری‌های اصلی معماری، براساس آزمایش‌های کافی و مستند انجام شده باشد. مثلا برای انتخاب فریم‌ورک‌ها یا فناوری‌های زیرساختی، بنچمارک‌ها و تست‌های کافی انجام شده باشد و نتایج آن‌ها موجود باشد و

		در اختیار کارفرما قرار گیرد .مثلا اگر Tomcat به عنوان سرور و RabbitMQ انتخاب شده، نیازمندی‌ها و آزمایش‌ها و بنچمارک‌های لازم برای این انتخاب‌ها به کارفرما گزارش شود.
کارایی		
	الزام	**شرح الزام**
2	**جداسازی روال تولید و استفاده از داده‌های نوشتنی و داده‌های خواندنی**	در موارد لازم با کمک تکنیک CQRS (Command Query Responsibility Segregation) روال ثبت داده و خوانش داده‌ها جداسازی شود. در موارد لازم و مفید، از غیرنرمال کردن داده‌ها (Denormalization) برای افزایش کارایی استفاده شود.
3	**تقسیم داده‌ها به Hot و Cold در موارد مفید**	در مواردی که به کارایی کمک می‌کند، داده‌ها به Hot و Cold تقسیم شوند و داده‌های Hot دارای سرعت واکشی بالاتری باشند.
4	**استفاده از روش‌های مناسب برای کاهش سربار عملیات**	در مواردی که به کارایی کمک می‌کند، از تکنیک‌هایی مثل Co-Location، Cleanup، کاهش لایه‌ها و سایر تکنیک‌های مفید برای کاهش سربار عملیات استفاده شود.
نگه‌داشت‌پذیری		
	الزام	**شرح الزام**
5	**پشتیبانی از مدل داده‌ای تطبیقی (Adaptive) در موارد لازم و مفید**	استفاده از Adaptive Data Model در مواردی که افزودن موجودیت‌ها یا تغییرات طراحی داده در زمان اجرا لازم است.
6	**برنامه‌نویسی جنبه‌گرا در موارد مفید**	استفاده از Aspect Oriented Programming (AOP) در مواردی که مفید است.
7	**استفاده از تکنیک‌های مرتبط با خط تولید نرم‌افزار در موارد مفید**	در موارد لازم و مفید، از تکنیک‌هایی مانند تولیدگر کد (Code Generator)، توسعه مولفه‌های زیرساختی پرکاربرد و غیره برای کاهش هزینه تولید نرم‌افزار استفاده شود.
معماری واسط کاربری		
	الزام	**شرح الزام**
8	**قابلیت استفاده از واسط کاربری توسط افراد توان‌خواه**	در صورت صلاحدید کارفرما

الزامات توسعه و کیفیت نرم‌افزار

الزامات ضروری

کیفیت کد		
	الزام	**شرح الزام**
1	**انتخاب زبان، چارچوب و سکوهای مناسب و رایج برنامه‌نویسی**	• توجه به ترجیحات و الزامات کارفرما، توجه به وجود متخصص کافی (Community) در کشور، تناسب زبان و چارچوب و سکوی برنامه‌نویسی انتخاب‌شده با نیازمندی‌های پروژه، استفاده از چارچوب‌ها (Frameworks)و سکوهای (Platforms) رایج، بروز و متناسب. • زبان‌ها، چارچوب‌ها و سکوهای رایج و مناسب مانند C#, Java, Python, .Net Core, React • زبان‌ها، چارچوب‌ها و سکوهای نامناسب مانند Delphi, Ext-JS, Silverlight
2	**استفاده از به‌روش‌ها، کتابخانه‌ها، فناوری‌ها و رویکردهای بروز و مناسب برنامه‌سازی**	استفاده نکردن از کتابخانه‌های غیررایگان یا کرک‌شده. استفاده نکردن یا نخریدن لایسنس ابزارها بدون احراز نیاز و تایید کارفرما. استفاده از فناوری‌های بروز و مناسب برای پروژه.
3	**توجه جدی به روش‌های برنامه‌نویسی تمیز**	• نام‌گذاری مناسب، اندازه مناسب کلاس‌ها و توابع، استفاده از سبک کد مناسب (Code Style)، استفاده موثر و کافی از توضیحات (Comment) و مستندسازی در متن برنامه‌ها) ابزارهای تولید مستند از کد یا (Document Generation همانند Javadoc یاPydoc • امکان مشاهده شواهد و گزارش‌های مناسب جهت حصول اطمینان برای کارفرما فراهم شود.
4	**توجه به بازآرایی کد و برنامه‌ریزی منظم برای آن**	آشنایی و توجه کافی به بوی بد (Bad Smells)، آشنایی و استفاده از روش‌های بازآرایی (Refactoring)، استفاده موثر از ابزارهای بازآرایی در محیط‌های توسعه (IDE)، برنامه‌ریزی منظم، ایجاد وظیفه و صرف زمان کافی برای بازآرایی برنامه‌ها
5	**نتایج ارزیابی قابل قبول ابزارهای تحلیل کد**	• استفاده موثر از ابزارهای تحلیل کد (Static Code Analysis) یا لینترها (Linters) جهت بررسی کیفیت کد، همانند SonarQube یا SonarLint. همچنین نتایج گزارش‌های تحلیلی این ابزارها باید قابل

		مشاهده باشد. مثلا تعداد ایرادهای جدی در گزارش ابزار تحلیل، زیاد نباشد. • امکان مشاهده گزارش‌های خودکار و آماده از طریق ابزار مناسب، برای کارفرما فراهم شود. • معیارهای Quality Gate را مشخص کنیم، مثلا: ○ No new blocker issues ○ Code coverage greater than 50% ○ Code coverage on new code greater than 60% ○ Duplicated lines < 10% ○ Technical Debt Ratio < 10%
6	**روال مناسب مرور کد**	استفاده جدی از روش مرور کد (Code Review) برای همه برنامه‌های تولیدشده. هر کد جدید توسط حداقل یک عضو دیگر تیم مرور شود) از طریق Merge Request یا .(Pull Request این کار توسط ابزار مناسب (مثلا ابزار GitLab) مدیریت شود و به صورت دستی و غیرسیستماتیک نباشد. زمان و دقت کافی برای مرور کد صرف شود .امکان مشاهده گزارش‌های خودکار و آماده از طریق ابزار مناسب، برای کارفرما فراهم شود.
7	**استفاده موثر از اصول طراحی نرم‌افزار**	استفاده مناسب از اصول طراحی نرم‌افزار، مثل اصول SOLID و الگوهای طراحی نرم‌افزار
8	**وجود فهرست بروز و مدون بدهی‌های فنی**	وجود فهرست مستند از ایرادهای برنامه، بدیهی‌های فنی و گام‌های بهبود آن‌ها.
کیفیت مستندات و مدیریت دانش		
	الزام	**شرح الزام**
9	**استفاده موثر و جاری از ابزار مناسب برای مدیریت دانش و مستندات**	• استفاده از یک ابزار و سامانه مناسب به عنوان مخزن مستندات و مدیریت دانش که دارای امکاناتی مثل جستجو و تعامل کاربران باشد. ترجیحا ابزارهای مبتنی بر ویکی مثل Confluence یا Azure wiki انتخاب شود. • دسترسی کارفرما به این ابزار در طول پروژه فراهم شود.
10	**ساختارمند بودن مجموعه اسناد در سامانه مدیریت دانش**	دسته‌بندی و گروه‌بندی مناسب اسناد در سامانه مدیریت دانش وجود دستورالعمل شفاف برای جایگذاری اسناد جدید در گروه‌بندی‌های از پیش تعیین‌شده
11	**وجود اسناد کلیدی و بروز و باکیفیت، به ویژه در زمینه معماری نرم‌افزار، طرح تضمین کیفیت**	• همواره سند معماری نرم‌افزار و سند طرح تضمین کیفیت نرم‌افزار به صورت بروز و دقیق در اختیار کارفرما قرار داشته باشد. رعایت قالب مورد نظر کارفرما در مورد این دو سند حیاتی است.

		• با توجه به ماهیت پروژه و نظر کارفرما، اسنادی مثل معماری امنیتی، مهاجرت داده‌ها، طرح آزمون خودکار و اسناد دیگری هم ممکن است در زمره اسناد مهم قرار می‌گیرد. فهرست اسناد باید به تایید کارفرما برسد.
12	**استفاده موثر از سامانه مدیریت دانش در ثبت تجارب، روندها و اشتباه‌های رایج تیم توسعه جهت اشتراک دانش**	امکان مشاهده شواهد و گزارش‌های مناسب جهت حصول اطمینان در این زمینه برای کارفرما فراهم شود.
13	**وجود روال‌های فنی ورود و خروج اعضا از تیم**	روال‌های مستند Onboarding و Offboarding وجود داشته باشد و گام‌های مستندسازی و مدیریت دانش در این روال‌ها تعیین شده باشد. مستندات هم در این روال‌ها به صورت جدی استفاده شوند و همچنین در خلال همین روال‌ها بروزرسانی شوند.
14	**وجود روال کالبدشکافی و مستندسازی نتایج آن**	به‌ویژه در زمان بروز حوادث (Incidents)، روال کالبدشکافی (Postmortem Analysis) با رویکردی مناسب اجرا شود و نتایج آن مستند شود .گزارش‌ها در اختیار ذی‌نفعان و کارفرما قرار گیرد .از رویکرد و قالب‌های مناسب (مثل روش گوگل) در این زمینه استفاده شود.
آزمون نرم‌افزار		
	الزام	**شرح الزام**
15	**آزمون پذیرش و آزمون سیستمی به صورت دستی**	• انجام آزمون دستی (Manual Test) توسط کارشناسان آزمون به اندازه کافی برای هر نسخه قبل از نصب در محیط عملیاتی. • فراهم‌سازی شرایط شبیه محیط عملیات در محیط آزمون. • دسترسی کارفرما به گزارش‌ها و نتایج آزمون.
16	**راه‌اندازی و نگهداری محیط آزمون**	وجود محیط آزمون (محیط دمو یا Staging یا UAT) . آزمون‌های خودکار و دستی در محیط آزمون انجام شود. امکان دسترسی نمایندگان کارفرما به محیط آزمون و استفاده عملی و دستی از این محیط همواره فراهم باشد.
17	**توجه جدی و کافی به آزمون واحد خودکار**	• استفاده از فناوری مناسب (مانند xUnit) برای آزمون واحد (Unit Testing) • توجه جدی و کافی به آزمون واحد • پوشش کافی آزمون واحد (Test Coverage) به ویژه برای بخش‌های مهم نرم‌افزار • محاسبه و پایش مستمر درصد پوشش آزمونهای خودکار • توجه به Testable بودن طراحی و پیاده‌سازی

		• توجه به هرم آزمون (آزمون واحد زیاد، آزمون یک‌پارچه‌سازی متوسط و آزمون پایانی کمتر) • توجه ویژه به میزان کافی پوشش آزمون در تغییرات پروژه (Incremental Coverage) • ارائه گزارش‌ها و شاخص‌های اطمینان‌بخش از کیفیت و کمیت آزمون‌های واحد و نتایج آن‌ها به کارفرما
18	**توجه کافی به آزمون یک‌پارچه‌سازی خودکار**	• استفاده از فناوری مناسب (مانند xUnit) برای آزمون یکپارچه‌سازی (Integration Testing) • ارائه گزارش‌ها و شاخص‌های اطمینان‌بخش از کیفیت و کمیت آزمون‌های یک‌پارچه‌سازی و نتایج آن‌ها به کارفرما.
19	**توجه کافی به آزمون پذیرش خودکار**	وجود روال و ابزار مناسب برای آزمون‌های پذیرش(Acceptance Test) ، انجام آزمون نهایی (End-to-End Test) و آزمون واسط کاربری به صورت خودکار. امکان رصد ارتباط بین نیازمندی‌ها و نتایج آزمون.
20	**محاسبه خودکار پوشش آزمون‌های کارکردی**	محاسبه Test Coverage به صورت خودکار و توسط ابزارهای مناسب مثل SonarQube
21	**مدل‌سازی آماری بار و استرس کاربران**	مدل‌سازی آماری مناسب برای رفتار کاربران جهت استفاده در آزمون بار و استرس
22	**امکان دسترسی مستقیم کارفرما به گزارش‌ها و داشبوردهای آزمون**	امکان دسترسی مستقیم کارفرما به ابزارهای مدیریت آزمون فراهم شود.
23	**استفاده از ابزار، رویه‌ها و اسکریپت‌های مناسب برای آزمون کارایی به صورت خودکار**	• انجام آزمون کارایی(Performance) ، آزمون بار (Load Test) و آزمون استرس با کمک ابزار مناسب مثل JMeter و.Gatling • تنظیم و طراحی بار مناسب در شرایط مختلف. • ارائه گزارش‌های مختلف که نشانگر مقدار سنجه‌های موردنیاز، از جمله زمان پاسخ، TPS، درصد خطاها، میزان مصرف منابع و ... در حین اجرای آزمون‌های بار و استرس باشد.
24	**تامین داده‌های آزمون مناسب**	وجود رویکرد، ابزار و روش مناسب برای تولید، تهیه یا تامین Test Data مشابه داده‌های محیط واقعی و استفاده از آن در آزمون‌ها.
مدیریت کد		
	الزام	**شرح الزام**
25	**استفاده از ابزار مناسب به عنوان مخزن کد**	ابزارهایی مثل Git و Azure DevOps به عنوان Code Repository مناسب هستند. ابزارهایی مثل CVS و SVN و TFS مناسب نیستند. در

		صورت استفاده از Azure، استفاده از مخزن Git مناسب است و استفاده از مخزن TFVC مناسب نیست.
26	**فرایند استاندارد و مشخص برای تعامل برنامه‌نویسان با مخزن کد**	فرایندهای معروف و مشخصی همانند Git Flow ، GitHub Flow و GitLab Flow برای تعامل با مخزن کد وجود دارد. هر فرایند برای برخی پروژه‌ها مناسب و برای برخی شرایط دیگر نامناسب است. البته این فرایندها ربطی به فناوری مخزن کد ندارند) مثلا GitHub Flow مخصوص استفاده در GitHub نیست .(این الزام ناظر به این موضوع است که فرایند مناسبی انتخاب شده باشد و به خوبی از این فرایند تبعیت شود. همچنین تکنیک‌هایی مثل Hotfix و Patch Version و Cherry Pick در موارد لازم مورد توجه قرار گیرد.
27	**استفاده از نسخه‌بندی معنایی**	نسخه‌بندی معنایی (Semantic Versioning) روشی مناسب برای تعیین شماره نسخه‌های یک نرم‌افزار است.
28	**ارتباط سیستماتیک بین تغییرات کد و تسک‌ها**	از طریق برقراری ارتباط بین مخزن کد و و ابزار مدیریت وظایف (Issue Tracker)، امکان ردیابی ارتباط هر تغییر در کد با نیازمندی نرم‌افزار یا وظایف تیم وجود داشته باشد. به عبارت دیگر، وقتی کد تغییر می‌کند، مشخص باشد این تغییر متناظر با چه نیازمندی یا وظیفه‌ای بوده است.
29	**ارتباط سیستماتیک بین آرتیفکت‌های قابل نصب و تگ‌های موجود در مخزن کد**	نصب یک نسخه از یک ماژول روی سرورها تنها در صورتی انجام می‌شود که متناظر با وضعیت آن یک تگ با نام نسخه آن ماژول روی گیت وجود داشته باشد.
30	**قرارگیری همه فایل‌ها و منابع لازم برای نصب نرم‌افزار بر روی مخزن کد**	همه منابع لازم جهت نصب نرم‌افزار باید روی مخزن کد) مثلا Git) قرار گیرند تا ساخت، انتشار و نصب نرم‌افزار به منبعی خارج از مخزن کد نیاز نداشته باشد و به صورت خودکار از مخزن کد قابل انجام باشد.
31	**نگه‌داری نسخ نرم‌افزار و فرآورده‌های مهم در یک مخزن نرم‌افزاری**	نگه‌داری نسخه‌های ساخته‌شده (مثل فایل‌های jar, dll, Docker Image) در یک مخزن نرم‌افزاری مناسب همانند Nexus یا Artifactory
ساخت، انتشار، نصب و بروزرسانی		
	الزام	**شرح الزام**
32	**نصب و بروزرسانی متن کد در محل کارفرما**	تحویل کد و منابع لازم بر روی مخزن کد به نحوی که قابلیت ساخت و نصب فقط با کمک مخزن کد در محل کارفرما وجود داشته باشد. به عبارت دیگر تحویل نرم‌افزار به صورت تحویل باینری یا برنامه قابل اجرا نباشد.

		ایجاد روال‌های نصب از روی متن برنامه نیز به عهده پیمانکار یا تیم توسعه است.
33	**خودکارسازی فرایند ساخت نرم‌افزار**	ساخت (Build) نرم‌افزار به صورت خودکار با ابزار مناسب. ساخت نرم‌افزار به صورت دستی انجام نشود و شامل مراحل انجام یا تنظیمات دستی نباشد.
34	**یک‌پارچه‌سازی مستمر با کمک ابزار مناسب**	استفاده موثر از رویکرد Continuous Integration در فرایند توسعه با کمک ابزار مناسب مانند GitLab CI, Azure DevOps, Jenkins
35	**وجود مرحله مستقل آزمون واحد در مراحل ساخت و تحویل**	در پایپلاین CI/CD مرحله Unit Test به خوبی جای گرفته باشد. معیارهای مناسب برای کنترل این مرحله (همانند پوشش آزمون و Incremental Code Coverage) تنظیم شده باشد.
36	**وجود مرحله مستقل آزمون یک‌پارچگی در مراحل ساخت و تحویل**	در پایپلاین CI/CD مرحله Integration Test به صورت مجزا جای گرفته باشد.
37	**وجود مرحله مستقل آزمون پذیرش در مراحل ساخت و تحویل**	در پایپلاینCI/CD، یک آزمون خودکار سطح بالا مثل Acceptance Test یا آزمون سیستمی جای گرفته باشد.
38	**وجود مرحله تحلیل کد در مراحل ساخت و تحویل**	در پایپلاین CI/CD مرحله Static Code Analysis به صورت مجزا و با کمک ابزار مناسب (مانند Sonar Qube) جای گرفته باشد. دروازه‌ها و معیارهای کافی و مناسب جهت کنترل در این مرحله تنظیم شده باشد.
39	**اتصال محیط آزمون به پایپلاین نصب**	وجود محیط آزمون (Staging) و اتصال آن به فرایند CI/CD به نحوی که قبل از نصب در محیط عملیاتی، نصب در محیط آزمون انجام شود و برخی آزمون‌ها در این محیط اجرا شوند.
40	**خودکارسازی نصب و بروزرسانی**	نصب نرم‌افزار به صورت خودکار با ابزار مناسب انجام شود. نصب نرم‌افزار به صورت دستی انجام نشود و شامل مراحل انجام یا تنظیمات دستی نباشد. ابزار مناسب و اسکریپت‌های لازم برای نصب کاملا خودکار آماده شود. جزئیات روال نصب به تایید کارفرما برسد.
41	**ارائه گزارش خودکار از شاخص‌های مهم دواپس**	شاخص‌هایی مثل DORA از قبیل میانگین زمان استقرار، فرکانس استقرار و نرخ شکست تغییر به صورت خودکار و از طریق ابزار قابل محاسبه باشد . امکان مشاهده و استفاده مستمر کارفرما از این گزارش‌ها فراهم باشد.
42	**خودکارسازی امکان بازگشت به نسخه قبلی**	در مواقعی مثل نصب ناموفق، رخداد خطا هنگام نصب یا بروز ایراد جدی بعد از نصب، امکان بازگشت خودکار (Rollback) وجود داشته باشد. برای بازگشت به نسخ قبلی، نیاز به انجام دستی مراحل نباشد و این کار به صورت خودکار با دستور نیروی پشتیبان قابل انجام باشد.

43	**خودکارسازی مهاجرت پایگاه داده**	تبیین ساز و کار و ابزار مناسب برای اعمال تغییرات پایگاه‌داده پیش از نصب نسخه جدید (DB Migration)
44	**نصب در کانتینرها در محیط عملیاتی**	استفاده موثر و مناسب از کانتینرها (مثل Docker) در محیط عملیاتی (Production)
45	**مدیریت کانتینرها با ابزار مناسب**	استفاده از ابزار مناسب مدیریت کانتینرها (مثل Kubernetes یا Nomad) در محیط عملیاتی (Production)
46	**وجود یادداشت‌های انتشار**	برای هر انتشار متن یادداشت انتشار مناسب (Release Note) وجود داشته باشد. ترجیحا این متن به صورت خودکار (بر اساس یادداشت‌های مربوط به فعالیت‌های هر نسخه) ایجاد شود و نه به صورت دستی.
47	**وجود برنامه و روال مناسب برای انتشار نرم‌افزار**	وجود Release Plan با گام‌های مناسب. در انتشار نرم‌افزار باید از روال‌ها و ابزارهای مناسب استفاده شود. گام‌ها و خروجی‌ها متناسب با نیاز کارفرما/بهره‌بردار تعیین شده باشد.
امنیت فرایند توسعه		
	الزام	**شرح الزام**
48	**وجود روال خودکار و منظم جهت پشتیبان‌گیری از منابع و فراورده‌های نرم‌افزاری**	از منابع مهم همانند مخزن کد، سامانه مدیریت پروژه، سامانه مستندات و سایر فراورده‌های نرم‌افزاری پشتیبان‌گیری منظم و خودکار انجام شود و نسخ پشتیبان در محل مناسب قرار گیرند.
49	**وجود روال مناسب احراز هویت برای حفظ محرمانگی همه سرورها و داده‌ها در محیط توسعه**	نه تنها در محیط عملیاتی، بلکه در محیط توسعه و فرایند تولید نرم‌افزار هم محرمانگی و سطوح دسترسی به مخزن کد، مدیریت مستندات و سایر منابع کنترل شود.
50	**وجود رویکرد متمرکز کنترل دسترسی برای همه منابع و سرورهای محیط توسعه**	استفاده از شناسه و رمز یکسان برای همه‌ی منابع از جمله مخزن کد، مدیریت مستندات، CI/CD با کمک ابزار مناسب و تعیین سطوح دسترسی مختلف برای حفظ امنیت
51	**امکان قطع سریع همه دسترسی‌های یک فرد به همه منابع و سرورهای محیط توسعه و محیط عملیات**	امکان قطع آنی دسترسی‌های یک فرد (مثلا فرد مشکوک یا فردی که قطع همکاری کرده) به همه منابع محیط توسعه (نظیر مخزن کد و مستندات) و محیط عملیاتی (نظیر امکان لاگین در سامانه)
52	**کنترل دسترسی مدیریتی به سرورهای محیط توسعه و عملیات**	جلوگیری از دسترسی غیرمجاز به دسترسی‌های مدیریتی سرورها و ابزارهای مهم. برخی اقدامات لازم در این زمینه عبارتند از: تغییر تنظیمات پیش‌فرض ابزارهای مورد استفاده (مثل رمز عبور و پورت) در محیط توسعه و محیط

		عملیاتی، اعمال سیاست‌های محدودکننده دسترسی باز (از طریق اینترنت) به سرورها
53	**آموزش، مرور و یادآوری روش‌های برنامه‌سازی امن به تیم توسعه**	آشنایی توسعه‌دهندگان با اشتباه‌های رایج برنامه‌نویسی که ایجاد ریسک امنیتی می‌کنند. وجود روال، چک‌لیست و آموزش لازم برای توجه برنامه‌نویسان به این موضوع.
54	**تحلیل متن برنامه‌ها برای کشف رخنه امنیتی به صورت منظم در فرایند توسعه با کمک ابزار مناسب**	بهره‌گیری موثر و منظم از ابزارهای مناسب برای اسکن متن برنامه‌ها جهت تحلیل ریسک امنیتی و کشف رخنه‌ها.
55	**پرهیز از نصب نرم‌افزارهای غیرمعمول و اضافی**	نصب هر نرم‌افزار و بسته نرم‌افزاری باید به اطلاع و تایید کارفرما برسد . نصب هیچ کتابخانه یا عامل نرم‌افزاری نباید مغایر با سیاست‌های امنیتی کارفرما باشد.
56	**نداشتن دسترسی مستقیم افراد و نرم‌افزارها به پایگاه‌داده**	افراد (توسعه‌دهندگان و مدیران) و برنامه‌ها (برنامه‌های دسکتاپ و هر مولفه نرم‌افزاری غیر از لایه داده) نباید امکان دسترسی مستقیم به پایگاه‌داده در محیط عملیاتی را داشته باشند. بدیهی است که سرورهای پایگاه‌داده در محیط عملیاتی باید در منطقه محرمانه باشند که مستقیما از اینترنت قابل دسترسی نباشند.
57	**پرهیز از قفل نرم‌افزاری یا سخت‌افزاری**	هرگونه استفاده از قفل‌های نرم‌افزاری یا سخت‌افزاری ممنوع است، مگر در موارد خاصی که به درخواست کارفرما باشد.
58	**رصد بروزرسانی‌های لازم امنیتی**	مطالعه، رصد و بررسی وصله‌ها و بروزرسانی‌های لازم برای دفع حملات و مشکلات امنیتی و ارائه پیشنهاد استفاده از آن‌ها به کارفرما.
59	**بروزرسانی سامانه نرم‌افزاری مطابق سیاست‌ها و ابلاغیه‌های امنیتی کارفرما**	بروزرسانی نرم‌افزار و اصلاح سیاست‌ها و سازوکارهای امنیتی آن جهت سازگاری نرم‌افزار با وصله‌ها و سیاست‌های مورد نظر کارفرما به طور منظم و دوره‌ای. در مواردی که انجام این کار برعهده بخش زیرساخت است، با هماهنگی و مشارکت پیمانکار انجام شود.
مدیریت امنیت عملیات		
	الزام	**شرح الزام**
60	**دسترسی برخی کارها فقط از طریق حضور در محل**	برخی دسترسی‌های مهم، حیاتی و تاثیرگذار، فقط در صورت حضور فیزیکی کاربر/مدیر/ادمین ممکن باشد. این موضوع با کمک روال‌های امنیتی مبتنی بر محل حضور فرد رعایت شود.
61	**فرایند مناسب مدیریت راز**	عدم نگهداری راز در کد. استفاده از ابزار مناسب مدیریت راز (مانند Vault)

لاگ و مانیتورینگ		
	الزام	**شرح الزام**
62	**ثبت لاگ مناسب برای رویدادهای مهم سامانه**	برای رویدادهای مهم سامانه، لاگ ثبت شود. مثلا: کنترل دسترسی کاربران، تغییر در داده‌های مهم، دسترسی به منابع مهم، دسترسی غیرطبیعی یا غیرمجاز، خطا یا خرابی. شواهد و گزارش‌های لازم برای اطمینان از حصول این الزام به کارفرما ارائه شود.
63	**بهره‌گیری از استاندارد و کتابخانه مناسب ثبت لاگ**	مانند SLF4J وLog4net
64	**استفاده صحیح و بجا از سطوح مختلف لاگ در پروژه**	سطوح مختلف لاگ (Log Levels) به خوبی و در کاربرد مناسب مورد استفاده قرار گرفته باشند (مثل سطح Debug و سطح Warning و غیره) . همچنین گزارش‌های اطمینان‌بخش در این زمینه به کارفرما ارائه شود.
65	**استفاده از ابزار مناسب برای تجمیع و تحلیل لاگ**	از ابزار مناسب همانند ELK استفاده شود. در صورت ارائه زیرساخت لازم توسط کارفرما، پیمانکار لاگ‌های سامانه را با زیرساخت مورد نظر کارفرما یک‌پارچه نماید. در غیر این صورت، زیرساخت لازم (مثلا ELK) را پیمانکار راه‌اندازی نماید. گزارش‌های مناسب در این سامانه ایجاد شده و امکان مشاهده گزارش‌های مورد نظر در سامانه مدیریت لاگ، از طریق دسترسی به داشبوردهای لازم، برای کارفرما فراهم شود. لازم است در این بند شفاف‌سازی شود که آیا راه‌اندازی ابزار تحلیل لاگ و ابزار تجمیع لاگ به عهده ارائه کننده سرویس می‌باشد یا این ابزار در اختیار وی قرار می‌گیرد و وی صرفاً مسئول پیکربندی و استفاده است. در صورت تفکیک وظایف مرزبندی باید کاملاً شفاف گردد.
66	**گزارش و هشدار خودکار مبتنی بر لاگ**	وجود داشبوردهای مفید برای پایش و هشدار شاخص‌های مهم با کمک تحلیل لاگ‌ها
67	**تشخیص حادثه (Incident) بر اساس تحلیل لاگ**	ایجاد تیکت Incident به صورت خودکار بر اساس تحلیل لاگ
68	**استفاده از ابزار مناسب برای مانیتورینگ سامانه نرم‌افزاری**	از ابزار مناسب (مثل Prometheus) برای مانیتورینگ سامانه نرم‌افزاری استفاده شود. در صورت ارائه زیرساخت لازم توسط کارفرما، پیمانکار امکانات و تسهیلات لازم را با زیرساخت موردنظر کارفرما یک‌پارچه نماید. در غیر این صورت، زیرساخت لازم را پیمانکار راه‌اندازی نماید.

		هشدارها و گزارش‌های مناسب در این سامانه ایجاد شده و امکان مشاهده گزارش‌های لازم در سامانه مانیتورینگ برای کارفرما فراهم شود. در موارد موردصلاحدید کارفرما، امکان ارسال هشدار از رسانه‌های مختلف (ایمیل و پیامک) فراهم شود.
69	**گزارش و هشدار خودکار از طریق سامانه مانیتورینگ**	وجود داشبوردهای مفید برای پایش و هشدار شاخص‌های مهم مبتنی بر سامانه مانیتورینگ .دسترسی به هشدارها و گزارش‌های مانیتورینگ برای کارفرما فراهم شود.
70	**محاسبه خودکار شاخص دسترس‌پذیری نرم‌افزار**	میزان دسترس‌پذیری (Availability) سامانه همواره به صورت خودکار محاسبه شود. نحوه محاسبه این شاخص، به تایید کارفرما برسد .کارفرما همواره به صورت آنلاین به این گزارش دسترسی داشته باشد. نقض توافق سطح سرویس (SLA) به کارفرما به صورت خودکار گزارش شود.
71	**محاسبه خودکار شاخص نرخ خطای عملیات کاربران**	نرخ خطایی که کاربران در انجام عملیات دریافت می‌کنند (خطا، تایم‌اوت، کندی بیش از حد و هر اشکال دیگری که مانع تکمیل طبیعی عملیات کاربر شود) به صورت خودکار محاسبه شود. نحوه محاسبه این شاخص، به تایید کارفرما برسد .کارفرما همواره به این گزارش آنلاین دسترسی داشته باشد. نقض توافق سطح سرویس (SLA) به کارفرما به صورت خودکار گزارش شود.
72	**روال مناسب برای مدیریت حوادث**	رویه و ابزار مناسب Incident Management مورد استفاده قرار گیرد. این روال به تایید کارفرما برسد. گزارش‌های خودکار شاخص‌های کلان (همانند تعداد حوادث در بازه‌های زمانی، زمان رفع اشکال و غیره) در اختیار کارفرما قرار گیرد.
فرایند توسعه نرم‌افزار		
	الزام	**شرح الزام**
73	**انتخاب متدولوژی مناسب توسعه و رعایت بخش‌های حیاتی متدولوژی**	استفاده جدی از یکی از متدولوژی‌های چابک مناسب با شرایط پروژه (مانند Scrum و Kanban و Lean). رعایت بخش‌های اصلی متدولوژی منتخب (مثل Sprint Planning و Standup Meetings در اسکرام)
74	**بهره‌گیری از ابزار مناسب جهت مدیریت چابک فرایند توسعه**	استفاده موثر از ابزارهای مناسب همانند ابزارهای Issue Tracking, Project Management, Agile Boards (همانند Jira یا Azure). دسترسی همیشگی کارفرما جهت مشاهده گزارش‌ها، بوردها و جزئیات وظایف در طول فرایند توسعه فراهم شود.

75	**وجود دستاوردهای ملموس در برنامه‌ریزی چرخه‌های توسعه**	در برنامه‌ریزی هر چرخه زمانی (Sprint) ، اهداف مشخص و ملموسی وجود داشته باشد .تاریخچه برنامه اسپرینت‌ها و همچنین خروجی‌ها و اهداف هر اسپرینت برای کارفرما قابل مشاهده و قابل درک باشد.
76	**به‌کارگیری نقش‌های اصلی متدولوژی منتخب**	نقش‌های کلیدی مورد تاکید متدولوژی منتخب تعیین شوند و متصدی این نقش‌ها برای تیم توسعه و همچنین برای کارفرما مشهود باشند .مثلا در صورت انتخاب متدولوژی اسکرام، متصدی نقش‌های Scrum Master و Product Owner مشخص باشند.
77	**حضور نقش فعال با وظیفه تعامل با ذی‌نفعان**	وجود افرادی در تیم که به طور منظم، کافی و فعال، با ذی‌نفعان و بازیگران سیستم (مثل کاربران، مشتریان، مدیران، بهره‌برداران و کارفرما) در ارتباط هستند و بازخوردها و نظرات آن‌ها را رصد می‌کنند.
78	**وجود روال‌های مناسب و کارآمد برای تعامل با ذی‌نفعان**	روندهای ساده و کارآمد برای دریافت نظرات ذی‌نفعان وجود داشته باشد. انجام اقدامات اجرایی منوط به دریافت تایید ناظر باشد.
79	**سرعت عمل در دریافت و رسیدگی به نظرات ذی‌نفعان**	سرعت در دریافت نظرات، سرعت بالا در رسیدگی به نظرات.
کیفیت تیم		
	الزام	**شرح الزام**
80	**عدم وابستگی به افراد**	تیم به یک یا چند عضو خود وابستگی بیش از حد نداشته باشد. در صورت از دست رفتن بعضی از اعضای تیم، بقیه تیم بتوانند به خوبی به کار ادامه دهند و به تدریج اعضای جدید را جایگزین افراد قبلی نمایند. تیم، پایدار و مقاوم باشد و با رفتن یک یا دو عضو، از بین نرود و یا در کارایی آن خلل شدید ایجاد نشود.
81	**تحصیلات مناسب**	بخش قابل توجهی از اعضای تیم، در دانشگاه‌های معتبر و در رشته‌های مرتبط تحصیل کرده باشند. میزان اعتبار دانشگاه‌ها، متناسب با اهمیت پروژه تعیین می‌شود. میزان ارتباط رشته‌های مورد تحصیل، متناسب با مقیاس پروژه و نقش افراد تعیین می‌شود.
82	**سابقه سنوات کار اعضا**	تعداد سنوات تجربه و سابقه شغلی اعضای تیم کافی باشد. در هر نقش، متناسب با نیازمندی‌های نقش سوابق کافی وجود داشته باشد. مثلا متصدیان نقش‌های مهم کم‌تجربه نباشند. میانگین سابقه افراد تیم، پایین نباشد.

83	**تاریخچه تشکیل تیم/پیمانکار**	تیم یا شرکت پیمانکار، سابقه کافی در حوزه مورد نظر داشته باشد. مثلا به تازگی تاسیس نشده باشد.
84	**تنوع نقش‌های موجود در تیم**	نقش‌ها و مهارت‌های متنوع و لازم در تیم حضور داشته باشند. مثل برنامه‌نویس، مدیر دیتابیس، برنامه‌نویس ارشد یا معمار، مالک محصول، اسکرام مستر و غیره.
85	**سابقه در حوزه فنی پروژه مورد نظر**	اعضای تیم، سوابق مرتبطی در حوزه فنی و فناوری‌های مورداستفاده داشته باشند. مثلا در گذشته از زبان برنامه‌نویسی، پایگاه‌داده، سبک معماری، ابزارها، زیرساخت‌ها و فناوری‌های مورد نظر در پروژه به اندازه کافی استفاده کرده باشند. یا مثلا اگر قرار است در پروژه جاری از زبان جاوا استفاده شود، سابقه استفاده کافی از زبان جاوا در تیم وجود داشته باشد.
86	**سابقه در حوزه کسب‌وکاری پروژه مورد نظر**	اعضای تیم، سوابق مرتبطی در حوزه کاری مورد نظر داشته باشند. در دامنه مورد نظر متخصص باشند و سابقه داشته باشند. مثلا اگر پروژه در حوزه مالی است، اعضای تیم سابقه کافی در دامنه پروژه‌های حوزه‌های مالی داشته باشند.
87	**سابقه در پروژه‌هایی در تراز پروژه مورد نظر**	اعضای تیم، سوابق کافی در پروژه‌هایی با مقیاس پروژه مورد نظر داشته باشند. مثلا اگر پروژه مورد نظر یک پروژه بزرگ و ملی است، اعضای تیم سابقه کار در پروژه‌های بزرگ و ملی را در کارنامه داشته باشند.
88	**وجود افراد شاخص در تراز پروژه**	به تعداد کافی، افراد خبره و شاخص با درک و دانش عمیق از فرایند و دامنه در تیم حضور داشته باشد. کسانی که تجربه و توانایی حل مساله در پروژه مورد نظر داشته باشند. متناسب با اهمیت و مقیاس پروژه، تعداد و عمق افراد شاخص در تیم مشخص می‌شود.

الزامات پیشنهادی

آزمون نرم‌افزار		
	الزام	**شرح الزام**
1	**استفاده از ابزار مناسب مدیریت آزمون‌های خودکار و دستی**	وجود رویکرد و ابزار مناسب مدیریت آزمون (Test Management Tool) برای مدیریت آزمون‌های خودکار و آزمون‌های دستی و حفظ گزارش‌ها و نتایج آن‌ها (ابزارهایی مثل Test Link)

الزامات یک‌پارچه‌سازی

الزامات ضروری

الزامات عمومی تعامل‌پذیری		
برخی از الزامات مربوط به یک‌پارچه‌سازی سامانه‌ها و تعامل‌پذیری آن‌ها، عمومی و کلی هستند و به سامانه خاصی وابسته نیستند. به ویژه الزامات مرتبط با نحوه توسعه و شیوه استفاده از APIها و سرویس‌ها، در این محور گنجانده شده است. برخی از کلیدواژه‌های مرتبط با این حوزه عبارتند از: Interoperability, API, Web Service, RESTful architecture		
	الزام	**شرح الزام**
1	**استفاده از وب‌سرویس برای تعامل با سایر سامانه‌های نرم‌افزاری**	برای ارتباط سامانه نرم‌افزاری با سایر سامانه‌ها، چه سامانه‌های داخل سازمان و چه سامانه‌های خارج از سازمان، از وب‌سرویس استفاده شود. از ارتباط در لایه پایگاه‌داده و سایر روش‌ها (به جز وب‌سرویس) پرهیز شود، مگر در مواردی که پیشاپیش به تایید کارفرما رسیده باشد.
2	**استفاده از پروتکل‌های ارتباطی استاندارد مناسب مثل REST**	از پروتکل‌های جدید، کارامد، استاندارد و مورد تایید سازمان برای تبادل اطلاعات استفاده شود. در صورت عدم وجود ترجیح یا الزام مشخص از طرف سازمان، در طراحی سرویس‌ها وAPI ها، REST بر SOAP ترجیح داده شود. همچنین JSON بر XML ترجیح داده شود.
3	**مستندنگاری APIها با استفاده از ابزار و استاندارد مناسب**	• استفاده از ابزارها یا استانداردهایی مثل Swagger یاOpen API • پیاده‌سازی وب‌سرویس‌ها به صورت Self-Document
4	**تست APIها با استفاده از ابزار مناسب**	پیاده‌سازی آزمون‌های خودکار برای تست وب‌سرویس‌ها با رویکرد و ابزارهای مناسب مثل REST-Assured، Postman، Swagger وKatalon نتایج و گزارش‌های آزمون، در اختیار کارفرما قرار گیرد.
5	**استفاده از API Gateway یا Message Broker مناسب برای ارتباط با سامانه‌های خارجی**	دریافت و ارائه API از/به خارج از سامانه نرم‌افزاری، از طریق یک واسط پیام مناسب (مثلا یک API Gateway) باشد و دغدغه‌هایی مثل ثبت لاگ، مدیریت دسترسی، دسترس‌پذیری و غیره از این طریق مدیریت شود.
یک‌پارچه‌سازی کاربران و ساختار سازمانی		
در سازمان، ممکن است سامانه متمرکزی برای مدیریت کاربران، نقش‌ها، دسترسی‌ها و ساختار سازمانی وجود داشته باشد یا در آینده راه‌اندازی شود. در صورت صلاحدید سازمان و در زمان موردنظر سازمان، پیمانکار موظف خواهد بود از زیرساخت متمرکز مدیریت کاربران تبعیت کند و سامانه نرم‌افزاری را با این زیرساخت، یکپارچه نماید.		

برخی از کلیدواژه‌های مرتبط با این حوزه عبارتند از: Central Authentication, Identity and Access Management, SSO, Access Levels		
	الزام	**شرح الزام**
6	**یک‌پارچه‌سازی با تعریف کاربران در سازمان**	در صورت صلاحدید سازمان، روال یکپارچه‌سازی و تبعیت از کاربران تعریف‌شده در سامانه مرکزی مدیریت کاربران، پیاده‌سازی شود. *جهت بررسی داخلی: در صورت وجود سامانه مرکزی مدیریت کاربران در سازمان و در صورت صلاحدید سازمان و در زمان موردنظر سازمان*
7	**یک‌پارچه‌سازی با ساختار سازمانی تعریف‌شده در سازمان**	در صورت صلاحدید سازمان، روال یکپارچه‌سازی و تبعیت از ساختار سازمانی تعریف‌شده در سازمان، پیاده‌سازی شود. سازوکار بروزرسانی و همگام‌سازی مناسبی در نظر گرفته شود و این سازوکار، به تایید سازمان برسد.
8	**یک‌پارچه‌سازی با Single Sign On در سازمان**	در صورت صلاحدید سازمان، از سامانه مرکزی ورود کاربران (SSO) برای ورود کاربران استفاده شود.
9	**یک‌پارچه‌سازی با Single Sign Out در سازمان**	در صورت صلاحدید سازمان، از سامانه مرکزی SSO برای خروج کاربران هم استفاده شود.
10	**یک‌پارچه‌سازی با سطوح دسترسی کاربران و نقش‌ها در سازمان**	یکپارچه‌سازی و تبعیت از نقش‌ها، سطوح دسترسی و دسترسی‌های تعریف‌شده در سامانه مرکزی مدیریت کاربران.
یک‌پارچه‌سازی با میان‌افزارهای مدیریت عملیات		
برخی میان‌افزارهای محیط عملیات (Production Middleware) در سازمان راه‌اندازی می‌شود و سامانه‌های نرم‌افزاری موظف به استفاده و یک‌پارچه‌سازی با این میان‌افزارها می‌شوند. به عنوان مثال برای این میان‌افزارها، می‌توان به ESB، زیرساخت مانیتورینگ، زیرساخت ثبت لاگ و زیرساخت امضای دیجیتال اشاره کرد. برخی از کلیدواژه‌های مرتبط با این حوزه عبارتند از: Service Bus, Monitoring, Log Management, Digital Signature		
	الزام	**شرح الزام**
11	**یک‌پارچه‌سازی با گذرگاه خدمات سازمان**	در صورت صلاحدید سازمان، اتصال به سایر سامانه‌ها از طریق گذرگاه مرکزی خدمات سازمان (ESB یا API Management) انجام شود. اقدامات و تمهیدات لازم برای این امکان فراهم شود.
12	**ارتباط با سایر سامانه‌ها صرفا از طریق گذرگاه خدمات سازمان**	در صورت نیاز سازمان، اتصال به سایر سامانه‌ها صرفا از طریق گذرگاه مرکزی سازمان (مثلا ESB) انجام شود و هیچ ارتباطی از هیچ طریق دیگری با سایر سامانه‌ها انجام نپذیرد مگر با اطلاع و تایید کتبی کارفرما.

13	**یک‌پارچه‌سازی با زیرساخت مدیریت لاگ در سازمان**	در صورت صلاحدید سازمان، لاگ سامانه به سامانه مدیریت مرکزی لاگ در سازمان ارسال شود. براساس قواعد مورد نظر سامانه مرکزی مدیریت لاگ در سازمان، لاگ سامانه تهیه و ارسال شود.
14	**یک‌پارچه‌سازی با زیرساخت مانیتورینگ در سازمان**	در صورت صلاحدید سازمان، امکان اتصال به سامانه مرکزی مانیتورینگ سازمان فراهم شود.
15	**یک‌پارچه‌سازی و اتصال به زیرساخت‌های امضای دیجیتال**	در صورت صلاحدید سازمان، امکان اتصال و استفاده از زیرساخت مرکزی امضای دیجیتال (مثل CA و PKI) در سازمان فراهم شود.

یک‌پارچه‌سازی قوانین و فرایندهای سازمانی

برخی از فرایندهای سازمانی، فراسامانه‌ای هستند و از چند سامانه نرم‌افزاری می‌گذرند. به عبارت دیگر، انجام برخی فرایندهای سازمان، نیازمند تعامل و ارتباط بین چند سامانه نرم‌افزاری هستند. هر سامانه، امکان تعامل در اجرای فرایندها را باید فراهم نماید.
برخی از کلیدواژه‌های مرتبط با این حوزه عبارتند از: Business Process Management, Business Rules

	الزام	شرح الزام
16	**امکان خودکارسازی فرایندهای سازمانی و پشتیبانی و یک‌پارچه‌سازی با BPMS مرکزی سازمان**	در صورت صلاحدید سازمان، امکان اتصال و تعامل با سامانه BPMS مرکزی سازمان فراهم شود.
17	**امکان خودکارسازی قوانین سازمانی و پشتیبانی و یک‌پارچه‌سازی با BRMS مرکزی سازمان**	در صورت صلاحدید سازمان، امکان اتصال و تعامل با سامانه BRMS مرکزی سازمان جهت بروزرسانی قوانین و مقررات کسب‌وکار، فراهم شود.

یک‌پارچه‌سازی داده‌ها

در برخی سامانه‌ها، لازم است داده‌ها با برخی سامانه‌های دیگر یک‌پارچه یا هم‌گام شود. در این صورت، لازم است تمهیدات لازم جهت یک‌پارچه‌سازی و هماهنگی داده‌ها انجام شود.
برخی از کلیدواژه‌های مرتبط با این حوزه عبارتند از: Data Integration, Master Data Management

	الزام	شرح الزام
18	**وجود طرح مناسبی برای یک‌پارچه‌سازی داده‌ها**	طرح و برنامه مناسبی برای یک‌پارچه‌سازی داده‌ها با سایر سامانه‌ها (در موارد لازم) وجود داشته باشد.
19	**استفاده از رویکرد، ابزار و پلتفرم مناسب برای یک‌پارچه‌سازی داده**	برای یک‌پارچه‌سازی داده‌ها با سایر سامانه‌ها (در موارد لازم) رویه و ابزار مناسب و کارامدی انتخاب شده باشد.
20	**توجه به امنیت در یکپارچه‌سازی داده‌ها**	توجه کافی به امنیت در یک‌پارچه‌سازی داده‌ها.

21	**تعریف و تبیین داده‌های مرجع Master Data**	داده‌های مرجع (Master Data) مشخص شود و از سایر داده‌ها تمایز داده شود .این تمایز به تایید کارفرما برسد.
22	**به‌کارگیری روال Master Data Management**	جلوگیری از تعارض بین داده‌های مشابه در سامانه‌های متفاوت با کمک MDM انجام شود.
23	**استفاده از راهکار مناسبی برای انتقال، تبدیل و بارگذاری داده‌ها**	در مواردی که نیاز به مهاجرت داده یا بارگذاری داده وجود دارد، از برنامه، ابزار و رویکردی مناسب و کارآمد استفاده شود. گام‌های لازم، تمهیدات لازم و مخاطرات فنی این کار به اطلاع و تایید کارفرما برسد.
24	**یک‌پارچه‌سازی با زیرساخت مدیریت داده‌های مرجع سازمان**	در صورت وجود سازوکارها و زیرساخت مدیریت داده‌های مرجع (MDM) در سازمان، از سازوکارها و استانداردهای مربوطه در سازمان تبعیت شود.

یک‌پارچه‌سازی انباره داده و هوش تجاری

انباره داده مرکزی داده و سامانه‌های گزارش‌گیری و هوش تجاری، به داده‌های سامانه‌های مختلف نیاز دارند. هر سامانه نرم‌افزاری، باید امکان ارسال داده‌ها به انباره داده مرکزی را فراهم نماید.

برخی از کلیدواژه‌های مرتبط با این حوزه عبارتند از: Datawarehouse, Business Intelligence

	الزام	شرح الزام
25	**امکان ارسال داده‌ها به انباره‌داده مرکزی**	در صورت صلاحدید سازمان، امکان ارسال داده به سامانه مرکزی انباره داده و هوش تجاری فراهم باشد .مدل‌های داده و توضیحات کافی جهت طراحی فرایندهای انتقال داده، در اختیار سازمان و پیمانکاران انباره داده قرار گیرد.
26	**همکاری در تهیه ETL مناسب و کارامد جهت ارسال داده به انباره‌داده مرکزی سازمان**	پیمانکار در طراحی اسکریپت‌های لازم برای انتقال داده به انباره داده مرکزی، همکاری مناسب و کافی داشته باشد.

یک‌پارچه‌سازی واسط کاربری

ممکن است لازم باشد که واسط کاربری هر سامانه نرم‌افزاری، در پنجره واحد یا پورتال مرکزی سازمان قابل رویت باشد. به عبارت دیگر ممکن است سازمان، زیرساخت یک‌پارچه‌ای برای ارائه واسط کاربری به کاربران نهایی در نظر بگیرد. در این صورت، هر سامانه باید از طریق همین زیرساخت یک‌پارچه دیده شود و اقدامات لازم برای این یک‌پارچه‌سازی انجام شود.

برخی از کلیدواژه‌های مرتبط با این حوزه عبارتند از: User Interface, Single Window, Portal

	الزام	شرح الزام
27	**قرارگیری و یک‌پارچه‌سازی واسط کاربری در پنجره واحد سازمان**	در صورت صلاحدید سازمان، واسط کاربری سامانه در پنجره واحد قرار گیرد و تمهیدات و اقدامات لازم برای این کار توسط پیمانکار انجام شود.

28	**تهیه واسط کاربری با تبعیت از سازوکار مورد نظر سازمان**	در صورت وجود استاندارد، رویه یا قالب یک‌پارچه در سازمان برای طراحی واسط کاربری، از این الزامات تبعیت شود.