MICROSERVICES

المقدمة:

تقنية مهمة جدا وقوية جدا ولازم تكون عارفها لو انت Web Developer التقنية دي حاليا مستخدمة بشكل كبير جدا في الشركات العالمية ولو عاوز تشتغل بره الوطن العربي فلازم تكون بتشتغل بالتقنية عشان هي للأسف مش مستخدمه في الوطن العربي.

خلاص بقي كفاية كلام بالألغاز هنتكلم النهاردة ان شاء الله عن تقنية ال Microservices وهي تقنية قوية جدا في بناء المواقع العملاقة والقوية خلونا نعترف عليها اكتر.

INTRO

البداية:

السلام عليكم ورحمة الله وبركاته قبل ما نتعرف علي مفهوم ال Microservices لازم تعرف الأول اللي خلانا نعمل النظام ده.

الحكاية كلها يا سيدي ان زمان كنا في كل المواقع بنستخدم ال Monolith في بناء المواقع وال "Monolith" باختصار هو نمط تصميم برمجي يشير إلى تصميم تطبيق أو نظام برمجي ككل، حيث يتم بناء التطبيق ككيان واحد ضخم ومتكامل يحتوي على جميع المكونات والموديلات المختلفة في طبقة واحدة.

هتقولي برضو فين المشكلة هقولك لأ بقي اصبر عليا المهم بقي ال Monolith بيمثل تناقضًا لأنماط التصميم الأخرى مثل النمط الميكروسيرفس والنمط المتقسم إلى عناصر مستقلة.
يتميز التصميم بالمونوليث بتجميع جميع المكونات والوظائف داخل وحدة واحدة ضخمة، حيث يكون النظام بأكمله مُعبّرًا في تنفيذ واحد ويتم تجميعه ونشره كوحدة واحدة. وبالتالي، يكون الوصول إلى مكونات النظام والتعامل معها يتم داخل الحدود الواحدة لهذا النمط.
يُستخدم التصميم بالمونوليث في تطبيقات البرمجيات التي تحتوي على أجزاء متعددة مرتبطة بشكل وثيق وتعتمد على بنية موحدة وتواصل مباشر بين المكونات. يُعدّ التصميم بالمونوليث مناسبًا للتطبيقات البسيطة والمتوسطة الحجم التي تتطلب موثوقية وتبسيطًا في التطوير والاختبار. ومع ذلك، قد يواجه التصميم بالمونوليث بعض التحديات عندما يتعلق الأمر

بالتطوير والتحديث والتوازن بين الأداء والقابلية للتوسع. فعندما يكبر النظام وتزداد تعقيداته، يصبح من الصعب إجراء التغييرات الكبيرة بشكل فعال دون التأثير على المكونات الأخرى. وبالتالي، قد يؤدي التصميم بالمونوليث إلى قيود في تطوير البرمجيات وإدارتها على المدى الطويل. على الرغم من هذه التحديات، لا يزال التصميم بالمونوليث مستخدمًا في بعض الحالات التي تتطلب تواصلًا وثيقًا بين المكونات، ويمكن استخدامه بنجاح في التطبيقات ذات الحجم المتوسط أو المشاريع البسيطة.

وعيوب الMonolith اللي أدت لإستعمال نظام ال Microservices كالأتي:

1. قابلية التوسع المحدودة: عندما ينمو حجم التطبيق أو تتغير متطلبات النظام، يصبح من الصعب توسيع Monolith بشكل فعال. يتطلب ذلك تغييرات كبيرة في الكود وإعادة تجميع النظام بأكمله، مما يتسبب في توقف مؤقت للخدمة ويعقد عملية التنقل والتطوير.
2. صعوبة الصيانة والاختبار: نظرًا لأن جميع المكونات متكاملة في Monolith، فإن إجراء الاختبار وإصلاح الأخطاء قد يصبح أكثر صعوبة. يصبح من الصعب تحديد نقاط الفشل وتعقبها في النظام، كما يتعذر اختبار المكونات بشكل منفصل دون التأثير على المكونات الأخرى.
3. التواصل المحدود: قد يصعب إنشاء اتصالات فعالة ومستقلة بين المكونات المختلفة داخل Monolith. يتطلب ذلك تضمين رموز التطبيق وتبادل البيانات بين المكونات المختلفة داخل حدود النظام الواحد، مما يقيد المرونة والتبادل بين المكونات.
4. تأثير الأخطاء الجانبية: عند حدوث خطأ في أحد المكونات داخل Monolith، يمكن أن يتسبب ذلك في تأثير سلبي على المكونات الأخرى. يمكن لخطأ بسيط أن يؤثر على استقرار النظام بأكمله ويتطلب إصلاحًا شاملاً.
5. تأثير التقنيات المحدودة: تحديث التقنيات والأدوات البرمجية الجديدة يمكن أن يكون أمرًا صعبًا في تصميم Monolith. يمكن أن يتسبب ذلك في تجاوز النظام بشكل كامل أو تأثير العمليات الأخرى في حالة تحديث أحد المكونات. باختصار، تصميم Monolith يأتي مع قيود وتحديات معينة فيما يتعلق بالتوسع والصيانة والاختبار. ينبغي التفكير جيدًا في هذه العيوب قبل اتخاذ القرار بشأن استخدام هذا النمط التصميم لتطبيقات البرمجيات.

تعريف الMicroservices :

وكده انت عرفت احنا عملنا نظام الMicroservices اللي هو نمط تصميم برمجي يهدف إلى تقسيم تطبيق أو نظام برمجي إلى مكونات صغيرة ومستقلة وركزلي علي مستقلة دي عشان النظام بيتحول الي عدة أنظمة كل واحد ملهوش علاقة بالتاني يعني كل واحد ليه قواعد البيانات الخاصة والسيرفر الخاص بيه كأنه برنامج كامل لوحده وبيطلق عليها "الميكروسيرفس" (Microservices). يتم تطوير ونشر كل ميكروسيرف بشكل منفصل ويتم تعيينه للقيام بوظيفة معينة داخل التطبيق. وأكيد بيحتاجو يتواصلو مع بعض فبيتواصلوا مع بعض من خلال واجهات برمجة التطبيق (API) الخارجية.

مميزت الMicroservices :

نيجي بقي لمميزات ال Microservicesاللي خلت شركات كتير تستخدمه وخلته احسن من ال Monolith:

1. القابلية للتطوير والتوسع: يعزز تصميم Microservices المرونة والتوسعية. يمكن تطوير ونشر كل ميكروسيرفس بشكل مستقل، وبالتالي يمكن تطوير أجزاء محددة من التطبيق بشكل مستقل دون التأثير على بقية النظام. يتيح هذا النمط التوازن بين حجم المشروع ونطاق التوسع.
2. سهولة الصيانة والاختبار: يتيح تصميم Microservices إجراء الاختبار وإصلاح الأخطاء بشكل مستقل في كل ميكروسيرفس. يمكن فحص واختبار كل ميكروسيرفس بشكل منفصل دون التأثير على بقية النظام، مما يسهل عمليات الصيانة وتحسين جودة البرمجيات.
3. التكنولوجيات المستقلة: يمكن استخدام لغات البرمجة وتقنيات قواعد البيانات المختلفة لكل ميكروسيرفس بشكل منفصل. هذا يتيح للفرق المطورة اختيار التقنيات التي تناسب أفضل احتياجات كل ميكروسيرفس والاستفادة من تطورات التكنولوجيا.
4. الفصل بين المسؤوليات: يوفر تصميم Microservices تقسيمًا واضحًا للمسؤوليات والوظائف. يمكن تعيين مهام محددة لكل ميكروسيرفس، مما يبسط إدارة النظام وفهمه ويسهل التعاون بين الفرق المختلفة.
5. المرونة في التكيف والابتكار: يسمح تصميم Microservices بتطوير وتحسين الميزات بشكل منفصل دون التأثير على بقية النظام. يمكن إدخال التغييرات والابتكارات بسرعة أكبر، مما يمنح المؤسسات مرونة أكبر للتكيف مع احتياجات العملاء وتغيرات السوق.

6. التوازن بين الأداء والقابلية للتطوير: يمكن تحقيق أداء ممتاز واستجابة سريعة في تصميم Microservices. يمكن تحسين وتنفيذ كل ميكروسيرفس بشكل مستقل للحصول على الأداء الأمثل، كما يمكن توفير مقاييس الأداء والتحسينات في المكونات الضرورية فقط.

بس لإن احنا عارفين ان الحلو مابيكملش ففي برضو عيوب لل Microservices:

1. التعقيد والتعقيد: تصميم Microservices يزيد من تعقيد النظام البرمجي. بدلاً من وجود تطبيق واحد متكامل، يصبح هناك العديد من الميكروسيرفس المنفصلة التي يجب تطويرها ونشرها وتشغيلها بشكل منفصل. يتطلب ذلك مهارات فنية وإدارية متقدمة للتعامل مع هذا التعقيد.
2. صعوبة إدارة النظام: يتطلب تصميم Microservices إدارة متقدمة للنظام بأكمله. يجب التفكير في تنسيق وتحكم ومراقبة الميكروسيرفس المختلفة وضمان توافقها وتعاونها مع بعضها البعض. هذا يتطلب أدوات وعمليات إدارة متقدمة وتكاملًا فعالًا بين المكونات.
3. الاعتمادية والاستقرار: تصميم Microservices يزيد من تعقيد تحقيق الاستقرار والاعتمادية. يعتمد النظام على عدة مكونات مستقلة تعمل معًا، وقد يكون من الصعب توفير ارتفاع مستوى الاعتمادية في كل الميكروسيرفس. يجب اتخاذ تدابير إضافية للتعامل مع أخطاء المكونات وإدارة التحمل والتكامل.
4. تعقيد اختبار واكتشاف الأخطاء: توفير اختبار شامل واكتشاف الأخطاء في بيئة Microservices قد يكون تحديًا. يجب اختبار كل ميكروسيرفس بشكل مستقل وتكاملها فيما بعد للتأكد من أن النظام بأكمله يعمل بشكل صحيح. تتطلب الاختبارات المتكررة والتكاملية استراتيجيات وأدوات متقدمة.
5. تكاليف التشغيل والبنية التحتية العالية جدا: تصميم Microservices يزيد من التحديات المتعلقة بتكاليف التشغيل والبنية التحتية. يتطلب وجود عدة ميكروسيرفس مستقلة توفير بنية تحتية متقدمة واحتياجات تشغيلية مستقلة لكل مكون. قد يكون هذا مكلفًا من حيث التكاليف والجهود المطلوبة.

وبكده نكون اتعرفنا علي ال Monolith وعيوبه واتعرفنا علي ال Microservices ومميزاته وعيوبه

وخليك عارف ان مع عيوب الmonolith في شركات كتير جدا وخاصة في الوطن العربي لسه بتستخدمه عشان ال Microservices بيحتاج تكلفه عالية وعدد كبير من المبرمجين.

وبس كدا احنا خلصنا لو كملت معانا لحد هنا فأحب أقولك ان في جيفاواي لما نوصل الف مشترك 10 حسابات شات جي بي تي.

وما طولش عليك بقي والسلام عليكم ورحمة الله وبركاته.