MioBook

مقدمه

در فازهای قبلی، از یک پایگاه‌داده رابطه‌ای مثل MySQL برای نگهداری داده‌های ساختارمند استفاده کردیم. اما برای داده‌هایی که نیاز به ساختاری منعطف‌تر و یا دسترسی سریع‌تر دارند، پایگاه‌داده‌های غیر رابطه‌ای (NoSQL) مناسب‌تر هستند.
در این فاز با مفهوم NoSQL آشنا می‌شویم و سپس از Redis به عنوان یک دیتابیس NoSQL برای مدیریت نشست (Session) کاربران استفاده می‌کنیم.

پایگاه‌داده‌های NoSQL

پایگاه‌داده‌های NoSQL برای پاسخ به محدودیت‌های پایگاه‌داده‌های رابطه‌ای (RDBMS) در مقیاس‌پذیری، عملکرد بالا و انعطاف‌پذیری در ذخیره و پرس‌وجوی داده‌ها ایجاد شده‌اند. این نوع پایگاه‌داده‌ها، برخلاف RDBMS-ها، ساختار مشخص و ثابتی ندارند و در دسته‌بندی‌های متفاوتی قرار می‌گیرند که چهار نوع آنها در جدول زیر آورده شده است:

نوع پایگاه‌داده	ساختار داده	مثال‌ها
Document Store	JSON, BSON	MongoDB, CouchDB
Key-Value Store	Key-Value Pairs	Redis, DynamoDB
Column-Family Store	Rows & Columns	Cassandra, HBase
Graph Database	Vertices & Edges	Neo4j, ArangoDB

پایگاه‌داده‌های NoSQL به‌دلیل طراحی غیررابطه‌ای خود، مزایای قابل‌توجهی نسبت به پایگاه‌داده‌های سنتی دارند. یکی از مهم‌ترین ویژگی‌های آنها مقیاس‌پذیری افقی (Horizontal Scaling) است؛ یعنی می‌توان با افزودن سرورهای جدید، ظرفیت ذخیره و پردازش سیستم را بدون ارتقای سخت‌افزار یک سرور اصلی (Vertical Scaling) افزایش داد.
علاوه بر این، NoSQL-ها به دلیل انعطاف‌پذیری در ساختار داده، امکان ذخیره انواع داده‌های نیمه‌ساخت‌یافته مانند JSON یا سندهای نامتقارن را فراهم می‌کنند (یعنی معمولا به Schema دقیق نیاز ندارند). این در پروژه‌های مدرن و متغیر بسیار سودمند است.

هر کدام از این ابزارها زبان کوئری و نکات مربوط به خود را دارد که در مهندسی داده، با آشنایی با آنها می‌توانیم بهترین پایگاه‌داده را برای مسئله خود انتخاب کنیم.
البته که این مزایا منجر به منسوخ شدن پایگاه‌داده‌های رابطه‌ای نمی‌شود. NoSQL-ها برای دست‌یابی به مزایای خود، مجبور به گذشتن از برخی از قابلیت‌های RDBMS-ها می‌شوند. به طور مثال، اکثر NoSQL-ها به طور کامل ACID نیستند و مثلاً برای تراکنش‌های مالی مناسب نیستند.
در مواردی مانند مدیریت نشست کاربران (Session Management) یا ذخیره‌سازی موقت داده‌ها (Caching) که نیاز به سرعت پاسخ بالا و تأخیر کم دارند، پایگاه‌داده Redis عملکرد خوبی نسبت به سیستم‌های سنتی دارند.

معرفی Redis

Redis یک پایگاه‌داده NoSQL از نوع Key-Value و مبتنی بر حافظه (in-memory) است که به دلیل عملکرد بسیار بالا و سادگی در استفاده، به‌ویژه در سیستم‌های real-time و توزیع‌شده، کاربرد گسترده‌ای دارد. برخلاف اکثر پایگاه‌داده‌ها که داده‌ها را روی دیسک ذخیره می‌کنند، ردیس داده‌ها را به‌صورت کامل در حافظه RAM نگه‌داری می‌کند و همین ویژگی باعث می‌شود که خواندن و نوشتن داده‌ها با سرعت بسیار بالایی انجام شود.
یکی دیگر از ویژگی‌های بارز ردیس، پشتیبانی از ساختارهای متنوع داده است. این ساختارها به توسعه‌دهندگان این امکان را می‌دهند تا منطق پیچیده‌تری نسبت به ذخیره مقدار خام (raw value) پیاده‌سازی کنند. برای مثال، با استفاده از ساختار Hash می‌توان همانند HashMap در جاوا، اطلاعات مختلف مربوط به یک کاربر را به صورت Key-Value داخل یک ساختار واحد ذخیره کرد.
Redis همچنین از Expiration Time، TTL یا همان Time To Live برای کلیدها پشتیبانی می‌کند. با این قابلیت می‌توان داده‌ها را تنها برای مدت زمان مشخصی در حافظه نگه داشت. این قابلیت در مدیریت نشست کاربران بسیار مفید است، زیرا می‌توان نشست‌هایی که مدت‌زمان مشخصی غیرفعال مانده‌اند را به‌صورت خودکار از سیستم حذف کرد. ردیس همچنین قابلیت persistence (ماندگاری داده) را از طریق snapshot یا فایل AOF (Append Only File) فراهم می‌کند تا در صورت خاموشی سیستم، داده‌ها از بین نروند.
در نهایت، Redis یک نرم‌افزار متن‌باز است که نصب و راه‌اندازی ساده‌ای دارد و با زبان‌های برنامه‌نویسی مختلف به خوبی یکپارچه می‌شود. در اکوسیستم Spring نیز پشتیبانی ردیس از طریق کتابخانه‌هایی مانند Spring Data Redis کامل است و به توسعه‌دهندگان این امکان را می‌دهد که با حداقل پیکربندی، از ردیس در پروژه‌های خود بهره ببرند.

نصب Redis-Server

همانطور که در پروژه قبلی یک سرور MySQL لوکال در سیستم خود بالا آوردیم و برنامه Spring خود را به آن متصل کردیم، در اینجا نیز باید ابتدا ردیس را نصب کنیم. می‌توانید با دنبال کردن این لینک، سرور ردیس را روی سیستم خود نصب کنید.
در ویندوز می‌توانید طبق راهنمای لینک از WSL استفاده کرده و یا از این ابزار استفاده کنید.

ابزار Redis-CLI

پس از بالا آمدن سرور ردیس، می‌توانیم با دستور redis-cli به آن متصل شویم. این ابزار می‌تواند به هر سرور ردیس خارجی نیز وصل شود:

```
redis-cli -h <host> -p <port> -a <password>
```

حال شما باید با انواع داده‌ها و دستورهای ردیس آشنا شوید. ردیس دارای ساختار داده‌های زیادی است که اصلی‌ترین آنها String، List، Hash، Set و Sorted Set است. برای آشنایی با آنها حتما این لینک را مطالعه کنید. پس از اینکه با دستورات و انواع داده در ردیس آشنا شدید، برای سرور خود یک رمز بگذارید.

شرح پروژه

در فازهای پیشین پروژه MioBook، کاربران پس از ثبت‌نام و ورود موفق به سیستم، به صفحات مختلف سایت دسترسی پیدا می‌کردند و اطلاعات آنها در پایگاه‌داده MySQL ذخیره می‌شد. با این حال، نحوه نگه‌داری نشست کاربران به صورت ماندگار در حافظه یا پایگاه‌داده در نظر گرفته نشده بود و در هر زمان صرفا یک نفر قابلیت لاگین کردن به سیستم را داشت. در این مرحله از پروژه، قصد داریم با استفاده از ردیس، مدیریت نشست کاربران را به شکل ایمن و سریع پیاده‌سازی کنیم تا سیستم ما بتواند چندین کاربر را به طور هم‌زمان پشتیبانی کند.

با اضافه کردن آخرین نسخه spring-boot-starter-data-redis به پروژه Maven خود، کتابخانه‌های مورد نیاز جهت اتصال و استفاده از ردیس در داخل کد جاوا و اسپرینگ را خواهید داشت. درباره نحوه استفاده از آن مطالعه کنید.

در این فاز، پس از ورود کاربر با نام کاربری و رمز عبور صحیح، یک توکن تصادفی (Session Token) تولید می‌شود و همراه با اطلاعات کاربر، در ردیس ذخیره خواهد شد. به طور مثال، می‌توان از یک UUID برای توکن استفاده کرد و آیدی کاربر را به آن نسبت داد.
این توکن به عنوان مجوز دسترسی، در پاسخ به فرانت‌اند بازگردانده شده و در درخواست‌های بعدی کاربر، از طریق هدر Authorization در ریکوئست‌های HTTP به سرور ارسال می‌شود. در سمت سرور، هر بار قبل از ارائه پاسخ به کاربر، باید اعتبار توکن بررسی شده و اطلاعات مربوط به آن از ردیس بازیابی شود.
هدف اصلی این تمرین، پیاده‌سازی کامل مکانیزم ورود، بررسی وضعیت نشست و خروج کاربر با استفاده از ردیس است. در صورت معتبر نبودن توکن، یا پایان یافتن زمان اعتبار آن، کاربر باید به صفحه ورود هدایت شود. همچنین، باید امکان خروج از سیستم (Logout) با حذف نشست از ردیس و فرانت‌اند نیز فراهم گردد.
شما باید در این تمرین، به نحوی Redis را پیکربندی و استفاده کنید که نشست کاربران به طور خودکار پس از نداشتن فعالیت در مدت مشخصی (مثلاً 20 دقیقه) منقضی شود (یعنی در صورتی که کاربر 20 دقیقه ریکوئست‌ای به سرور نفرستد، نشست او منقضی می‌شود). همچنین توجه به امنیت ذخیره‌سازی توکن در سمت فرانت‌اند و بررسی session در تمام API-های محافظت‌شده ضروری است.

نکات پایانی

- این تمرین در گروه‌های حداکثر دو نفره انجام می‌شود. برای تحویل آن کافی است که یکی از اعضای گروه، لینک مخزن گیت‌هاب و Hash مربوط به آخرین کامیت پروژه را در سایت درس آپلود کند. پروژه شما بر روی این کامیت مورد ارزیابی قرار می‌گیرد.
- حتما کاربر IE-S04 را به پروژه خود اضافه کنید.
- ساختار مناسب و تمیزی کد برنامه، بخشی از نمره همه پروژه‌های شما خواهد بود. بنابراین در طراحی ساختار برنامه و همچنین خوانایی کد دقت زیادی به خرج دهید.
- هدف این تمرین یادگیری شماست. لطفاً تمرین را خودتان انجام دهید. در صورت مشاهده شباهت بین کدهای دو گروه، از نمره هر دو گروه مطابق سیاستی که در کلاس گفته شده است کسر خواهد شد.
- سوالات خود را تا حد ممکن در گروه درس مطرح کنید تا سایر دانشجویان نیز از پاسخ آنها بهره‌مند شوند. در صورتی که قصد مطرح کردن سوال خاص‌تری داشتید، از طریق ایمیل با طراحان این تمرین ارتباط برقرار کنید.