سرویس Reservation:

این سرویس دارای دو سرویس REST به شرح زیر می‌باشد:

1- سرویس رزرو

این متد برای ایجاد یک رزرو جدید با ارسال درخواست به صف پیام (Queue) استفاده می‌شود. استفاده از صف به دلیل افزایش عملکرد (Performance) در مواجهه با تعداد زیاد درخواست‌ها می‌باشد و همچنین به بهبود همزمانی (Concurrency) سرویس کمک می‌کند. با این روش، بار کاری به صورت موثرتری مدیریت می‌شود و از بروز خطاهای احتمالی که ناشی از همزمانی هستند، جلوگیری می‌شود.

همچنین، استفاده از Async به ما این امکان را می‌دهد که پردازش‌ها را به صورت غیرهمزمان انجام دهیم. این امر باعث می‌شود که وقتی درخواست رزرو ارسال می‌شود، کاربر منتظر نماند تا عملیات پردازش کامل شود و به جای آن قادر به ادامه تعامل با سرویس باشد. این امر به بهبود تجربه کاربری کمک کرده و ظرفیت پاسخ‌دهی سیستم را افزایش می‌دهد.

مثال استفاده:

curl --location 'http://localhost:803/api/v1/reservations/' \

--header 'Content-Type: application/json' \

--data '{

"userId": 1,

"startTime": "2025-02-12T14:00:00.000000",

"endTime": "2025-02-12T15:00:00.000000"

}'

ورودی‌ها:

CreateReservationRequestDto: شامل اطلاعات مربوط به رزرو (مانند شناسه کاربر و زمان شروع).

خروجی:

این متد خروجی ندارد.

خطاها:

InvalidUserException: اگر کاربر معتبر نباشد.

ValidationException: زمانی که زمان شروع رزرو قبل از زمان کنونی باشد.

NotFoundException: اگر هیچ فضای خالی برای رزرو پیدا نشود.

مراحل اجرایی:

1. بررسی وجود کاربر: با استفاده از شناسه کاربری، اعتبار کاربر بررسی می‌شود.
2. تأیید زمان شروع: زمان شروع رزرو باید بعد از زمان کنونی باشد.
3. جستجوی نزدیک‌ترین فضای خالی:لیست فضاهای خالی در ابتدای برنامه از جدول

AvailableSlot کوئری زده شده و در کش Redis ذخیره می‌شوند.مدت زمان نگهداری داده‌ها در Redis به صورت کانفیگی تعیین شده است.نحوه ذخیره‌سازی در Redis به صورت SortedSet می‌باشد و سپس بر اساس الگوریتم جستجوی دودویی (Binary Search) نزدیک‌ترین فضای خالی برگردانده می‌شود.

1. رزرو در صورت پیدا کردن فضای خالی:برای کنترل همزمانی و جلوگیری از رزرو یک زمان برای کاربران متفاوت از Optimistic Lock استفاده می‌شود.پس از به‌روزرسانی مقدار فیلد is\_reserved، یک رکورد جدید در جدول Reservation در وضعیت Pending ثبت می‌شود.سپس این اسلات از Redis حذف می‌شود.تمامی این عملیات در یک تراکنش انجام می‌شود.

این روش طراحی به ما این امکان را می‌دهد که در زمان‌های با بار بالا، با بهینه‌سازی عملکرد و بالا بردن همزمانی، تجربه بهتری برای کاربران فراهم کنیم.

1. لغو یک رزرو

این متد برای لغو یک رزرو و به روزرسانی اطلاعات زمانهای خالی پیاده سازی شده است.

curl --location 'http://localhost:8030/api/v1/reservations/' \

--header 'Content-Type: application/json' \

--data '{

"userId": 1,

"startTime": "2025-02-12T14:00:00.000000",

"endTime": "2025-02-12T15:00:00.000000"

}'

ورودی‌:

CancelReservationRequestDto : شامل اطلاعات مربوط به لغو رزرو (مانند شناسه رزرو، دلیل رزرو و توضیحات).

خروجی:

در خروجی پارامتر از نوع Boolean برگردانده می شود.

خطاها:

NotFoundException: اگر رزرو با شناسه ورودی پیدا نشود.

مراحل اجرایی:

1. بازیابی اطلاعات رزرو بر اساس شناسه رزرو در ورودی
2. به روزرسانی وضعیت رزرو در جدول Reservation
3. ایجاد یک رکورد جدید در جدول CancelReservation
4. به روزرسانی اطلاعات زمانهای خالی در جدول AvailableSlots
5. ذخیره اطلاعات زمانهای خالی در کش ردیس

ایندکسهای مورد نیاز روی جداول:

جدول AvailablSlots

برای بهبود جستجوهای زمانی، ایندکس‌ روی Start\_time گذاشته شده است همچنین با توجه به اینکه جستجوروی تمام زمان‌های رزرو شده یا نشده زیاد صورت می گیرد روی این فیلد هم ایندکس گذاشته شده است.

جدول Reservation:

در صورتی که نیازبه جستجوی رزروهای یک کاربر خاص داشته باشیم ایندکس روی این فیلد می تواند سرعت جستجو را افزایش دهد.

برای بهبود جستجوهای زمانی مثلا رزروهای بین دو تاریخ خاص ایندکس گذاری روی این فیلد می تواند مفید باشد.