یک شرکت که برای کارمندان خود سرویس نهاردر نظر گرفته است و غذا توسط آشپزخانه شکت پخته میشود.

در حال حاضر افراد تا روز تهیه غذا (ساعت 9 روز غذا) امکان رزرو غذا بصورت دستی دارند. در حال حاظر مشکلاتی در روال دستی برای شرکت ایجاد شده قصد دارد یک سیستم رزرو غذا برای کارمندان خود ایجاد کند تا افراد بتوانند بر اساس تقویم غذایی تعریف شده برای خود غذا رزرو کنند.

* منوی غذا توسط آشپزخانه تعریف می شود که اطلاعاتی نظیر نام ، قیمت و مواد لازم را شامل می شود
* بر اساس موجودی و برنامه ریزی آشپزخانه برای هر هفته یک تقویم غذایی تعریف می شود(برای هر روز یک غذا) و کارکنان می توانند غذای خود را در روز های مورد نظر رزرو کنند.
* غذا ها بصورت نیم پرس و 1 پرس سرو میشود و هر کار مند می تواند تا 2 پرس غذا برای هر روز رزرو نماید
* حد اکثر توان پخت برای هر روز در تقویم مشخص می شود و مجموع رزرو ها نباید از حد اکثر توان پخت آشپزخانه بیشتر شود برا مثال اگر حد اکثر توان پخت 50 پرس می باشد تعداد رزرو نباید بیشتر از 50 پرس شود
* بر اساس تنظیمات سیستم برای هر روز می توان تا 1 روز قبل غذا رزرو کرد (این تنظیمات قابل تغییر می باشد)
* در انتهای هر ماه مبلغ غذای های رزرو شده برای هر نفر گزارش گیری میشود و از فیش حقوقی کارمندان کسر می گردد.
* public async Task<Result> Handle(MakeReservationCommand request, CancellationToken cancellationToken)
* {
* var hasReservedFood = await ReservationRepository
* .AnyReservationExistsAsync(request.PersonalId, request.Date);
* if (!hasReservedFood)
* {
* return Result.Failure(ReservationError.ReservationExistsError);
* }
* var foodCalendar = await FoodCalendarRepository.GetByIdAsync(request.FoodId, cancellationToken);
* var limitation = await LimitationRepository.GetLimitationsAsync(cancellationToken);
* var reservationCount = await ReservationRepository
* .PersonalReservedFoodAtDateAsync(request.Date, request.PersonalId);
* var hasReachedLimitation = limitation.FoodReservationLimit.Value <
* (reservationCount?.Sum() ?? 0) + request.WholeFoodCount + request.HalfFoodCount;
* var today = new DateOnly(DateTime.UtcNow.Year, DateTime.UtcNow.Month, DateTime.UtcNow.Day);
* var hasReachedDeadline = (today.ToDateTime(new TimeOnly(0, 0 ,0)) - request.Date.ToDateTime(new TimeOnly(0,0,0))).Days
* < limitation.ReservationDeadline.Value;
* var totalReservedFoodCount = await ReservationRepository.GetTotalReservedFoodCountAsync(request.FoodId, cancellationToken);
* var totalFoodLimitHasReached = totalReservedFoodCount
* .Sum(e => e.WholeFoodCount + e.HalfFoodCount \* 0.5) > foodCalendar!.MaximumCapacity.Value;
* if (hasReachedLimitation)
* {
* return Result.Failure(ReservationError.FoodReserveLimitationHasReachedError);
* }
* if (hasReachedDeadline)
* {
* return Result.Failure(ReservationError.FoodReserveDeadlineHasReachedError);
* }
* if (totalFoodLimitHasReached)
* {
* return Result.Failure(ReservationError.TotalFoodLimitHasReachedError);
* }
* var newReservation = ReservationAggregate
* .MakeReservation(GuidProvider, request.Date,
* new ReservedFood(request.FoodId, request.CalendarId),
* new(request.WholeFoodCount, request.HalfFoodCount),
* request.PersonalId);
* if (!newReservation.IsSuccess)
* {
* return Result.Failure(newReservation.Errors.FirstOrDefault()!);
* }
* await ReservationRepository
* .AddReservationAsync(newReservation.Data!, cancellationToken);
* await ReservationRepository
* .SaveChangesAsync(cancellationToken);
* return Result.Success();
* }
* }