تمرین کامپیوتری شماره 2

عنوان: Concurrent ticket reservation system

درس: مبانی رایانش توزیع شده

استاد راهنما: دکتر رضا شجاعی[۱]

رشته: مهندسی کامپیوتر

دستیاران آموزشی: محمدرضا ولی [۲]

نیمسال دوم سال تحصیلی 03-1402

[۱] نشانی پست الکترونیکی: r.shojaee@ut.ac.ir

[۲] نشانی پست الکترونیکی: mvali@ut.ac.ir

پیاده سازی Concurrent ticket reservation system

در این تمرین شما یک سیستم رزرو بلیت را به صورت Concurrent با استفاده از زبان Go پیاده‌سازی می‌کنید. این سیستم به چند Client به صورت همزمان اجازه می‌دهد که بلیت خود را برای یک Event مشخص رزرو کنند در حالی که Fairness و data consistency سیستم حفظ می‌شود.

1. راه‌اندازی پروژه جدید

یک دایرکتوری جدید برای پروژه خود بسازید (go mod init)، فایل main.go را در این دایرکتوری تعریف کرده و ساختارهای داده لازم را برای Event و Ticket تعریف کنید (Event and Ticket struct)

```
type Event struct {
    ID          string
    Name        string
    Date        time.Time
    TotalTickets int
    AvailableTickets int
}

type Ticket struct {
    ID      string
    EventID string
}
```

2. پیاده‌سازی سرویس رزرو بلیت

Struct مربوط به TicketService را برای منطق مدیریت Event و رزرو بلیت‌ها پیاده‌سازی کنید.

متدهایی را در این Struct برای ساخت Event جدید، نشان دادن لیست Eventهای در دسترس و رزرو بلیت برای یک Event مشخص را تعریف کنید.

از ساختار داده Concurrent (sync.Map) برای ذخیره Eventها و بلیت‌های متناظر با هر Event استفاده کنید.

در ادامه کدهای اولیه مربوط به متدهای CreateEvent، ListEvents و BookEvents در اختیار شما قرار داده شده است.

```
func (ts *TicketService) CreateEvent(name string, date time.Time, totalTickets int) (*Event, error) {
    event := &Event{
        ID:          generateUUID(), // Generate a unique ID for the event
        Name:        name,
        Date:        date,
        TotalTickets: totalTickets,
        AvailableTickets: totalTickets,
    }

    ts.events.Store(event.ID, event)
    return event, nil
}
```

```
func (ts *TicketService) ListEvents() []*Event {
    var events []*Event
    ts.events.Range(func(key, value interface{}) bool {
        event := value.(*Event)
        events = append(events, event)
        return true
    })
    return events
}
```

```
func (ts *TicketService) BookTickets(eventID string, numTickets int) ([]string, error) {
    // Implement concurrency control here (Step 3)
    // ...

    event, ok := ts.events.Load(eventID)
    if !ok {
        return nil, fmt.Errorf("event not found")
    }

    ev := event.(*Event)
    if ev.AvailableTickets < numTickets {
        return nil, fmt.Errorf("not enough tickets available")
    }

    var ticketIDs []string
    for i := 0; i < numTickets; i++ {
        ticketID := generateUUID()
        ticketIDs = append(ticketIDs, ticketID)
        // Store the ticket in a separate data structure if needed
    }

    ev.AvailableTickets -= numTickets
    ts.events.Store(eventID, ev)

    return ticketIDs, nil
}
```

3. پیاده‌سازی Concurrency control

نقاط بحرانی (Critical sections) را در کد خود شناسایی کنید (قسمت‌هایی که دسترسی همزمان می‌تواند منجر به Race condition یا Data inconsistency شود)

از رویکردهای Synchronization (sync.Mutux، sync.RWMutex یا Channels) برای کنترل دسترسی به Shared data استفاده کنید تا از Data consistency مطمئن شوید.

تابع رزرو را به گونه‌ای پیاده کنید که از Lock یا رویکردهای Synchronization قبل از Modify کردن موجودی بلیت‌ها استفاده می‌کند.

4. پیاده‌سازی Client interface

یک Client interface ساده بسازید که به کاربران اجازه می‌دهد تا با این سیستم رزرو بلیت ارتباط برقرار کنند.

توابع لازم برای نشان دادن لیست Eventهای در دسترس و رزرو بلیت برای یک Event مشخص توسط کاربر را تعریف کنید.

از Goroutines برای هندل کردن درخواست چند کاربر به صورت همزمان استفاده کنید.

5. پیاده‌سازی Fairness و Resource management

مکانیزمی برای اطمینان از Fairness در سیستم رزرو بلیت پیاده سازی کنید که از Starvtion جلوگیری می‌کند و به تمام کاربران شانسی برابر برای رزرو بلیت می‌دهد.

از روش‌هایی مثل Semaphor یا Leaky bucket برای اعمال محدودیت بر روی تعداد درخواست‌های همزمان و جلوگیری از Resource exhaustion استفاده کنید.

6. پیاده‌سازی Logging و Error handling

مکانیزمی برای Logging اطلاعات رزرو بلیت در حین عملیات رزرو تعریف کنید.

از مکانیزم‌های Error handling برای هندل و Recovery ارورهایی که ممکن است در حین رزرو بلیت یا تعامل کاربر با سیستم ایجاد شود استفاده کنید.

7. Caching

یک مکانیزم Caching برای افزایش کارایی سیستم از طریق Cache کردن آن دسته از Eventهایی که توسط تعداد زیادی کاربر به صورت منظم مورد دسترسی قرار می‌گیرند در کد خود پیاده‌سازی کنید.

- ## جمع بندی و نکات پایانی

- مهلت تحویل: **1403/02/09**
- پروژه در گروه‌های 4-2 نفره انجام می‌شود. (گروه بندی در سامانه ایلرن نیز انجام می‌شود و تحویل تمرین به صورت گروهی خواهد بود)
- تمام اعضای گروه می بایست کار را تقسیم کنند و این تقسیم کار در گزارش نهایی مشخص شود؛ نمره هر فرد از تمرین بر اساس میزان مشارکت ذکر شده در فایل گزارش خواهد بود.
- برای پیاده سازی این تمرین از زبان برنامه‌نویسی Go استفاده کنید.
- کدهای نمونه ذکر شده در این تمرین، برای درک بهتر قرار داده شده‌اند و می‌توانید از آن‌ها استفاده کنید یا از ابتدا کدهای خود را بنویسید؛ همچنین از آنجایی که هدف تمرین درک بهتر Concurrency در زبان Go است، می‌توانید تا جایی که خواسته‌های تمرین نادیده گرفته نشوند، از پیچیدگی‌های سیستم رزرو بلیت که با خواسته‌های مسئله ارتباطی ندارند صرف نظر کنید و آن‌ها را پیاده سازی نکنید.
- دقت کنید گزارش نهایی شما می‌بایست همانند یک Document باشد و شامل توضیح کد و ساختار کد، همچنین نتیجه نهایی اجرای کد و اسکرین شات‌های دقیق از تمام مراحل باشد؛ همچنین کدهای خود را به همراه این Document به صورت فایل فشرده با فرمت زیر آپلود کنید:

CA2_<firstmember_lastname>_<secondmember_lastname>.rar

- ساختار صحیح و تمیزی کد برنامه، بخشی از نمره‌ی این پروژه شما خواهد بود. بنابراین در طراحی ساختار برنامه دقت به خرج دهید.
- برای هر قسمت کد، گزارش دقیق و شفاف بنویسید. کدهای ضمیمه شده بدون گزارش مربوطه نمره‌ای نخواهند داشت.
- هدف این تمرین یادگیری شماست. لطفا تمرین را خودتان انجام دهید. در صورت مشاهده‌ی مشابهت بین کدهای دو گروه، مطابقت سیاست درس با گروه متقلب و تقلب دهنده برخورد خواهد شد. (استفاده از ابزارهای هوش مصنوعی، توجیهی برای شباهت گزارش و کدهای دو گروه نیست و در صورتی که ایده خود را از کدهای موجود در گیتهاب می‌گیرید، حتما منبع خود را در گزارش ذکر کنید)
- سؤالات خود را تا حد ممکن در گروه درس مطرح کنید تا سایر دانشجویان نیز از پاسخ آن بهره‌مند شوند. در صورتی که قصد مطرح کردن سؤال خاص‌تری دارید، از طریق ایمیل زیر ارتباط برقرار کنید.
    - mvali@ut.ac.ir

موفق باشید