```swift
//
//  CustomDatePicker.swift
//  ElegantTaskApp
//
//  Created by 이윤지 on 2023/04/14.
//

import SwiftUI
import Foundation

struct CustomDatePicker: View {

    @Binding var taskDate: Date //⭐️
    @Binding var currentDate: Date
    @State var isPresented = false
    //버튼을 눌렀을때 월이 변해야함
    @State private var currentMonth: Int = 0
  //  @State var tasks: [TaskMetaData]
    @State private var isChecked: Bool = false
    @State private var isChecked2: Bool = false
   // @State var isOn = true
    @State var isOn2 = false
    @State var isOn3 = Bool()
    @Binding var selectedDate: Date // Add selectedDate state
    @State var isOn4 = false

    // ⭐️TaskStore 객체를 구독하도록 @ObservedObject 어노테이션을 추가, 추가버튼을
     눌렀을때 바로 업데이트 되도록 하는 것임. (.shared)를  붙여야함
    @ObservedObject var taskStore = TaskStore.shared

    var body: some View {
        ZStack{
            LinearGradient(gradient: Gradient(colors:
             [.purple.opacity(0.2), .pink.opacity(0.3)]), startPoint:
             .leading, endPoint: .topTrailing)
                .opacity(0.2)
                .ignoresSafeArea()
            ScrollView(.vertical, showsIndicators: false) {


                VStack(spacing: 17){
                    //날짜 Days..
                    let days: [String] = ["일","월","화","수","목","금","토"]

                    HStack(spacing: 20){


                        VStack(alignment: .leading, spacing: 10) {

                            Text(extraData()[0])
                                .font(.caption)
                                .foregroundColor(.pink)
                                .fontWeight(.semibold)
```

```
        //달력의 월 (폰트크기랑,색상)
        Text(extraData()[1])
            .font(.title.bold())
        //커스텀 데이터피커
        Image(systemName: "calendar")
            .font(.title3)
            .foregroundColor(.orange)
            .overlay(
                DatePicker("", selection: $currentDate,
                 displayedComponents: [.date])
                    .blendMode(.destinationOver)

            )


    }

    Spacer(minLength: 0)

    Button{
        withAnimation{currentMonth -= 1

        }
    } label: {
        Image(systemName: "chevron.left")
            .font(.title2)
    }

    Button{
        withAnimation{currentMonth += 1

        }

    } label: {
        Image(systemName: "chevron.right")
            .font(.title2)
    }

    HStack {
        Spacer()
        Button(action: {
            // Plus button action
            self.isPresented = true // AddTaskView를 열기
             위해 isPresented 값을 true로 변경
        }) {
            Image(systemName: "plus")

                .font(.title2)
            Text("추가")


        }
        .padding(.vertical,10)
        .padding(.horizontal,13)
```

```
            .background(Capsule()
                .fill(Color.orange.opacity(0.5).gradient))
        }
        .foregroundColor(.pink)
    }
    .padding(.horizontal)

    //🔥

    .sheet(isPresented: $isPresented) {
        AddTaskView(onAdd: { task,arg  in
            if TaskStore.shared.newtasks.firstIndex(where:
             { isSameDay(date1: $0.taskDate, date2:
             task.time) }) != nil {
                // 해당 날짜에 일정이 있는 경우
               //
                TaskStore.shared.newtasks
                [index].task.append(task)
            } else {
                // 해당 날짜에 일정이 없는 경우
                //
                 TaskStore.shared.newtasks
                 .append(TaskMetaData(task: [task],
                 taskDate: task.time, isOn3: isOn3))
            }
        }, selectedDate: selectedDate) // 선택된 날짜 전달
    }

    //날짜 뷰


    HStack(spacing: 0){
        ForEach(days,id: \.self){ day in
            //요일
            Text(day)
                .font(.callout)
                .fontWeight(.black)
                .frame(maxWidth: .infinity)
                .foregroundColor(.pink) //요일색상입니다.

        }

    }

    //Dates
    //lazy grid
    let columns = Array(repeating: GridItem(.flexible()),
     count: 7)
    //날짜 그리드의 설정
    LazyVGrid(columns: columns, spacing: 10) {
        ForEach(extractDate()) { value in

            //⭐️
          //  CardView(value: value, isOn3: isOn3)
```

```
            CardView(value: value, isOn3: isOn3,
             selectedDate: selectedDate)


                .background(
                    Capsule()
                        .foregroundColor(.pink)
                        .padding(.horizontal, 8)
                        .opacity(isSameDay(date1:
                         value.date, date2: currentDate) ?
                         0.5 : 0)
                )
                .onTapGesture {
                    currentDate = value.date
                    selectedDate = value.date
                }
        }

     }
    .onReceive(TaskStore.shared.$newtasks) { _ in
                // TaskStore.shared.newtasks 배열이 업데이트되면
                 뷰가 자동으로 업데이트됩니다.
            }

    VStack(spacing: 20) {
        Text("일정 목록 - \(currentDate.lunarDateString())")
            .font(.title2.bold())
            .frame(maxWidth: .infinity, alignment: .leading)
            .padding(.vertical,20)

        ForEach(TaskStore.shared.newtasks, id: \.id) {
         taskMetaData in
            if isSameDay(date1: taskMetaData.taskDate,
             date2: currentDate) {

                  ForEach(taskMetaData.task, id: \.id) {
                   task in

                      HStack{
                          Image("cakedrawing1")
                         // Text("아")
                          VStack(alignment: .leading,
                           spacing: 10) {


                              Text(task.time, style: .time)
                              Text(task.title)
                                  .font(.title2.bold())
                                  .onChange(of:
                                   currentMonth) { newValue
                                   in
                                      // 월달 업데이트하기
                                      currentDate =
                                   getCurrentMonth()
```

```
                                                DispatchQueue.main
                                               .async {
                                                      print("블루마틴-\
                                               (TaskStore.shared
                                               .newtasks)")
                                                  }
                                              }

                                        }
                                        Spacer()
                                        Button("삭제") {
                                             TaskStore.shared
                                              .delete(taskMetaData)
                                          //
                                           TaskStore.shared
                                           .update(taskMetaData)
                                        }
                                        .padding(.trailing)



                                     }
                                   .padding(.vertical, 10)
                                   .padding(.horizontal)
                                   .frame(maxWidth: .infinity, alignment:
                                    .leading)
                                   .background(
                                       Color(.systemPink)
                                           .opacity(0.3)
                                           .cornerRadius(10)
                                   )
                               }
                           }
                       }

                       if TaskStore.shared.newtasks.isEmpty {
                           Text("일정이 없습니다.")
                       }
                   }

                   .padding()

               }

               .onChange(of: currentMonth) { newValue in
                   // 월달 업데이트하기
                   currentDate = getCurrentMonth()

               }

           } .padding(.vertical)
       }
   }
```

```
@ViewBuilder
func CardView(value: DateValue,isOn3: Bool,selectedDate: Date)->some
 View{

    VStack{

        if value.day != -1{
            if let task = TaskStore.shared.newtasks.first(where: { task
             in
               // print("카드뷰의 일정- \(task)") // Task 값 출력
                return isSameDay(date1: task.taskDate, date2:
                 value.date)
            }){


                Text("\(value.day)")

                    .font(.title3.bold())
                   //
                    .foregroundColor(isSameDay(date1: task.taskDate,
                     date2: currentDate) ? .white : .primary)
                    .frame(maxWidth: .infinity)
                Spacer()

                Circle()
                    .foregroundColor(isSameDay(date1: task.taskDate,
                     date2: currentDate) ? .white : (isOn3 ?
                     Color.green : Color("Pink")))
                    .frame(width: 8, height: 8)
                    .overlay(
                        task.isOn3 ?
                            Image("cakedrawing1")
                               // .resizable()
                                //.scaledToFit()
                            : nil
                    )
                //오늘 날짜에 일정이 있다면 파란색 동그라미가 뜹니다.
                    .overlay(
                        isSameDay(date1: value.date, date2: Date()) ?
                            Text("오늘")
                                .font(.caption)
                                .foregroundColor(.white)
                                .padding(4)
                                .background(Color.blue)
                                .clipShape(Capsule())
                                .offset(x: 0, y: 20) : nil
                    )

            }
            else{

                Text("\(value.day)")
                //🤣날짜 색깔을 바꾸는 것.
                    .foregroundColor(.primary)
```

```
                    .font(.title3.bold())
                    .frame(maxWidth: .infinity)


                Spacer()


                //선택한 날짜가 오늘일때, "오늘" 설정
                if isSameDay(date1: value.date, date2: Date())


                {
                    Text("오늘")
                        .font(.caption)
                    //"오늘"글자색을 흰색으로 합니다.
                        .foregroundColor(.white)
                        .padding(4)
                        .background(Color.orange.gradient.opacity(0.7))
                        .clipShape(Capsule())
                        .offset(x: 0, y: 3)

                }

            }
        }
    }
    .padding(.vertical,9)
    .frame(height: 50, alignment: .top)



}


// 데이터 체킹하기

func isSameDay(date1: Date, date2: Date)->Bool{
    let calendar = Calendar.current

    return calendar.isDate(date1, inSameDayAs: date2)
}

// extrating 디스플레이에 년도와 월 추출하기
func extraData()->[String] {
    let formatter = DateFormatter()
    formatter.dateFormat = "YYYY MMMM"
    let date = formatter.string(from: currentDate)

    return date.components(separatedBy: " ")
}

func getCurrentMonth()->Date{
    let calendar = Calendar.current

    //현재 월 얻기
```

```
        guard let currentMonth = calendar.date(byAdding:
                .month, value: self.currentMonth, to: Date()) else{
            return Date()
        }
        return currentMonth
    }

    func extractDate() -> [DateValue] {
        let calendar = Calendar.current
        let currentMonth = getCurrentMonth()

        var days = currentMonth.getAllDates().compactMap { date ->
         DateValue in
            let day = calendar.component(.day, from: date)
            return DateValue(day: day, date: date)

        }

        let firstWeekday = calendar.component(.weekday, from:
         days.first?.date ?? Date())

        for _ in 0..<firstWeekday-1 {
            days.insert(DateValue(day: -1, date: Date()), at: 0)
        }


      //  let date15 = days.first(where: { calendar.component(.day, from:
       $0.date) == 15 })


        return days
    }


}



// 현재 월날짜를 얻기 위한 확장
extension Date{
    func getAllDates() -> [Date]{

        let calendar = Calendar.current

        //시작날짜 얻기
        let startDate = calendar.date(from: Calendar.current.dateComponents(
            [.year, .month], from: self))!

        let range = calendar.range(of: .day, in: .month, for: startDate)!


        //날짜 얻기
        return range.compactMap { day -> Date in
```

```
            return calendar.date(byAdding: .day, value: day - 1, to:
             startDate)!
        }

    }

}


//현재날짜를 음력날짜로 변환하기 위한 확장

extension Date {
    func lunarDateString() -> String {
        let chineseCalendar = Calendar(identifier: .chinese)
        let formatter = DateFormatter()
        formatter.calendar = chineseCalendar
        formatter.locale = Locale(identifier: "ko_KR")
        formatter.dateStyle = .medium
        formatter.timeStyle = .none
        _ = formatter.string(from: self)
        let lunarComponents = chineseCalendar.dateComponents([.year,
         .month, .day], from: self)
        let lunarMonth = lunarComponents.month ?? 0
        let lunarDay = lunarComponents.day ?? 0
        let lunarString = "음력\(String(format: "%02d",
         lunarMonth)).\(String(format: "%02d", lunarDay))"
        return lunarString
    }
}


struct CustomDatePicker_Previews: PreviewProvider {
    static var previews: some View {
        MainContentView()
    }
}



//extension Date {
//    func lunarDateString() -> String {
//        let chineseCalendar = Calendar(identifier: .chinese)
//        let formatter = DateFormatter()
//        formatter.calendar = chineseCalendar
//        formatter.locale = Locale(identifier: "ko_KR")
//        formatter.dateFormat = "MM.dd"
//        let chineseDate = formatter.string(from: self)
//        return chineseDate
//    }
//}
```