ABSTRACT
With the increase of people's awareness of sports, people pay more and more attention to their physical state during exercise, but most of the existing apps on the market that record sports routes and basic information are more complex. Based on this demand, we designed and implemented a lightweight IOS app that uses GPS to display the user’s route and speed in real-time while running and displays the full route after the run.

KEYWORDS
IOS app, GPS, Sports

INTRODUCTION
Because most of the sports recording apps on the market are relatively complicated, we hope to design and implement a lightweight walking information recording software. It has a simple user interface and is very easy to use. In terms of functionality, we want it to record and display the user's location in real time, and in addition, it should show the user's real-time pace when running.

When the workout is over, the complete route the user ran will be marked on the map, with different colors representing different speeds. If the user chooses to save this time, the basic information will be saved and can be reviewed.

In addition, we have also designed a set of badge reward systems. If the distance traveled by users reaches different levels in each exercise, the system will display different badges.

DESIGN
Software functions:
We have three main functions: New Run, Past Runs and Badges.

UI
We have designed a simple UI by using StoryBoard:
IMPLEMENTATION:

We have used these technical methods:

**Swift:**

Swift is a powerful and intuitive programming language for macOS, iOS, watchOS, tvOS and beyond. Writing Swift code is interactive and fun, the syntax is concise yet expressive, and Swift includes modern features developers love. Swift code is safe by design, yet also produces software that runs lightning-fast.

**Core data & NSFetchResultsController:**

Core data is pretty powerful in iOS app development. You can always use CoreData to save your app's permanent data for offline use. By using it, you can just define your data's types and relationships. CoreData, at runtime, will manage everything for you. which means you don't have to access in a database directly. just imagine it is an agent between you and the real database. you don't have to do those SQL operations yourself.

And for this app, we often have to fetch data from CoreData, so thats when you need this NSFetchResultsController. The best thing about it is, for example, you have 100 data, and 100 table cells to represent your data in the app. if you update one of them, you normally have to do Corresponding operation your database and update the whole UI. but using NSFetchResultsController, it will call one of its functions called delegate to only update the datum that has been modified, instead of update the whole UI. So you dont do anything most of the time.

They are handy but i have to say using it incorrectly is gonna cause crashes at runtime of the app. And it did take us a long time to debug.

**Storyboard:**

Storyboard is a visual representation of the user interface of an iOS application, showing screens of content and the connections between those screens. A storyboard is composed of a sequence of scenes, each of which represents a view controller and its views; scenes are connected by segue objects, which represent a transition between two view controllers. Xcode provides a visual editor for storyboards, where you can lay out and design the user interface of your application by adding views such as buttons, table views, and text views onto scenes. In addition, a storyboard enables you to connect a view to its controller object, and to manage the transfer of data between view controllers. Using storyboards is the recommended way to design the user interface of your application because they enable you to visualize the appearance and flow of your user interface on one canvas.

**EVALUATION:**

After our test, the APP can effectively record the real-time position of the user in time, correctly calculate the user's movement speed, and correctly mark the complete movement route after the end of the exercise. Users can successfully review their exercise history in the past run page.