|  |  |
| --- | --- |
| Movie Calendar | |
| Student names | Student numbers |
| Wenmeng Tan | 2017040366 |
| Jiaqiao Jiang | 2017040370 |
| Meiyue You | 2017040376 |

# Project Description

Our "movie calendar" is a multi-functional calendar for movie lovers. Under the premise of ensuring the original function of calendar viewing dates, it can recommend wonderful daily movies for users, and help users quickly go to third-party websites to watch the movie recommended that day.

To further suit fans' needs, our "movie calendar" also includes the abilities to schedule and check the weather for the next few days. We hope that users can get the movie information they want to see on our app, and make their own viewing plans and other schedules by combining the weather forecast of recent days. **Comfortable travel, beautiful movie.**

# Project Design

First of all, we wanted this calendar to have a beautiful interface. We searched various styles of UI design and pictures, and finally determined the current style after continuous attempts and improvements. **Photoshop** and **SAI** helped us a lot in this process.

It can be argued that **the design process and the implementation process go hand in hand**. From the perspective of users, we combine our experience of using various kinds of software to achieve as many functions as possible. We do as much as we can think of, add functions to our code.

Of course, our knowledge is very limited. In the process of completing the project, we did a lot of research and learnt some new components:

1. **react-navigation**: realize navigation and page switching.

2. **action-button**: realize the floating button. Click one to pop up the other two.

3. **Animated:** create a simple rotation effect.

4. **calendar-strip**: set a pointer to this component and reference its method through the pointer.

5. **datePicker**: select start and end dates.

6. **elements**: help us create a custom pop-up.

7. **linear-gradient:** realize the gradient effect.

8. **storage:** realize local storage.

9. **webView**: realize the function of jumping to third-party websites to watch films.

10. etc.

# Application Instructions

Open "movie calendar" and you can see **the picture of today's recommended movies** above and the **calendar** below. We can **click the movie classic lines to enter the movie details page**, where we can see more wonderful stills and lines of the movie, as well as various information of the director and actors, etc. By **clicking on ">"**, we can quickly **go to a third-party website to watch the films**.

If the movie appeals to us enough, we can put it on our **schedule** right away. **Click the plus button at the bottom right of the recommendation page to add the schedules**. Go to the schedule page, and we can view **the previous events or add new events**. Enter the event title, select the event start and end time, then a new plan is born!

If we choose to watch the movie in the cinema, don't forget to check **the recent weather forecast**. **Click the small weather icon on the top left of the recommendation page to go to the weather forecast interface**. We can see **the weather conditions today and for the next five days**. By sliding the page, we can also see today's wind direction, wind speed, air humidity and other information.

Of course, these functions are also **very suitable for us to use in our daily life**, while **checking the weather** and **making plans** we can also enjoy **colourful movie recommendations**！

# Development Notes:

(1) Using the method provided by the calendar-strip component to get the selected date. **The** "**method**" **differs from what we learned before**.

**Solution**: By setting a ref (pointer) in calendar-strip props we can get the day and month outside the component by invoking the "getDate()" and "getMonth()" methods provided by this component.

(2) One of the most challenging parts is **the asynchronous nature of JavaScript (JS)**, because JS has no execution order, it will **create confusion in the parts that need to be executed sequentially**. When fetching data many times in a loop, the speed of each fetch is different. This causes the resulting information not to be stored in ascending order by date.

**Solution**: Write a sort function which sorts by date, and get date-ascending data.

# Testing

To ensure portability, we **tested our App on many types of phones** to make sure the styles and layouts worked well. We also **got feedback from other students** to improve our project better.

**On the schedule page**:

1. Change the fixed distance value to the proportional value, make page layout display normally on different types of mobile phones.

2. When we slide to delete a plan, deleting the next one will make the previous one go into the state of being opened as well.

**On the weather page**, generate a new url when getting weather data for different dates. Make sure the dates are reasonable. For example:

1. There are 12 months in a year, 31 days in some, 30 days in others (regardless of February).

2. February has 29 days in a leap year and 28 in an ordinary year.

3. When generating the day of the week（Mon. Sun.）, the value of week will continue to add 1, notice if this value is over 7.

# Issues

On **movie recommendation** part:

The data of the movie recommendations should be enriched to fully understand people's preferences and **collect more types of movie information**.

On **schedule** part:

1. Select a specific date to **show the selected day's schedule**. Set an object or array to store the daily schedule and find the schedule based on the date.

2. **Mark if there is a plan for a certain day**, or if there are multiple schedules. Set a property for each day of the calendar to indicate whether there is any plan for that day.

On **Weather** part:

1. We hope the weather **forecast information can be updated in real time** andwe can **check other cities’ weather**, which needs to find a more comprehensive website to fetch data.

2. Use another set of weather icons to **show the night weather state**. This is easy to solve, just change the reference paths of the images by time.

# Extra Credit

1. Naming our software and set its icon.
2. Setting the launch image for the software.
3. Using "store" to implement local storage of schedules.
4. Using "webView" to jump to third-party websites and watch movies directly.
5. "Animated" is used to make a simple rotation animation to represent the loading effect.
6. Packing our project into an installation package（AndroidPackage： apk）.

# Conclusion and future improvements

**What we thought went well**：

(1) Collecting the classic movie quotes, show the film's introduction, cast, stills, and so on, the information is detailed and rich.

(2) Realizing the connection with the third party websites and provides the channels for users to watch movies directly.

(3) Clicking one button to display the other two to make the page cleaner, and using gradients to make the page prettier.

(4) We can choose the start and end time of the schedule, if the schedule information is not entered properly, the schedule will not be generated. This is very humanized and suitable for users to make long-term planning.

(5) Getting the effect which is similar to histogram by setting the styles of the elements in the "<FlatList />". A component can show various effects with a variety of different decorations.

(6) Making the background more beautiful by using PNG images.

(7) The rotation of the weather icon is used to represent the loading effect.

(8) We set up the launch image, name, and icon for our app to make it come alive.

**What enhancements we would like to make to the application and how we might code them**：

(1) Connecting to database, store user information and more movie information. Setting up user registration and login interface, customized for each user, set user’s favourite list.

(2) Clicking on the stills to see its larger version. Implementing this by "<ImageViewer />".

(3) Updating weather information in real time, reminding users of suitable clothing collocation for climate. Getting more detailed information through "fetch", writing the logic of recommending different clothing combinations according to different climates.

(4) Reminding the user of the schedule in the mobile notification bar. Getting the notification permission of mobile phone.