Software Design Document

Mobile Smart Application Development

Group Member: 廖若凡、罗丹、熊伟峰、陈子健

Date: Jun 11th

Group: Group 1

Project Name: Meme Guide

Content

Project Introduction

This project is a Kotlin-based Android app called "Meme Guide". This project is committed to creating a hot meme query and communication platform that not only enables young users to communicate with hot memes, but also helps middle-aged and elderly groups to narrow the digital divide through the functions of meme encyclopedia search, intelligent recommendation, hot meme forum, chat room and so on.

Requirement Design

User Requirement

Due to the wide range of product user groups, large age range and other characteristics, user needs are very extensive. On the one hand, for young users, we hope that users can gain a sense of cultural identity in this software, resonate with insiders, and discuss and create new works on familiar Internet meme. For middle-aged and elderly users who are not familiar with the Internet, we hope to provide users with an inquiry and recommendation system that can conveniently query the network terms encountered around them, and avoid the embarrassing scene that they cannot understand when communicating with others as much as possible.

Function Requirement

According to the analysis of user requirements, functional requirements are mainly divided into two parts. According to the needs of different user groups, the product should have the following core functions:

1. **Query Search**

   This function can help users quickly search and understand the Internet meme or Internet phrase that they don't understand

2. **Meme Recommendation**

   The function can make recommendations based on the type or area of hot memes that users search for or view frequently

3. **Meme Forum**

   This function provides a platform for both young users to exchange memes, and also provides a platform for users who can't find out the meaning of a meme

4. **Message Sending Between Users**

   This function provides users with the function of private chat with other users, helping users to deeply communicate with users with similar interests, and thus generate user bonding and cultural identity

5. **Meme Article Publishing Function**

   This feature is the core of content production in the app, and any user can post an article or topic through this feature, allowing other users to discuss their content

Overall Design

Structure of Meme Guide

1. Home Page

Top search bar: Contains the user avatar display and search box.

Banner: Used to advertise a new meme or website feature.

List of memes: Displays the latest and most popular memes, including name, description, author, and release date.

Trending hashtags: Displays the most popular hashtags. Users can search for memes by hashtags.

Bottom navigation bar: used to jump to each page

2. Meme details page

Meme Details: Displays meme details, including name, author, description, tag, likes, and comments.

Comments section: Allows users to add comments and view other users' comments.

3. User profile page

User profile: Displays a user's profile, including an avatar, username, followers, and favorites

Browsing history

Drafts box

Favorites: Displays favorite meme

Edit Profile: Allows users to edit their profile and upload an avatar.

4. Search the page

Search Bar: Allows users to search for Meme by entering keywords.

Search Results: Displays a list of stems that match the search keyword.

Popular search items

Historical search records

5. Discover the page

Shows the meme's popularity rankings

Discover more terriers

6. Message page

Contact Display

Records contact information

7. Publish the page

Save and publish drafts

Title content editing

Publish a selection of covers

Features of the Application

1. User Management: Allows users to create and manage their accounts, including registration, login, reset passwords, etc.

2. Browse Meme Popularity rankings: Allows users to view a list of all popular meme popularity rankings.

3. Search Terriers: Allows users to search meme by keyword.
4. Meme details: Allows users to view the meme details, including the meaning, usage and origin of the meme.
5. Favorites: Allows users to add their favorite meme to their favorites for later viewing.
6. Sharing memes: Allows users to share their favorite memes on their social media platforms. Users can add comments when sharing.
7. Post meme introduction: Explain and share the meme you are interested in.

User Interface Design

Home page design

We based the page design on ConstraintLayout, which contains:

a. Search bar at the top (with search icon and avatar icon)
b. headers of recent hot topics
c. Main image (for displaying avatar stems or ads)
d. RecyclerView, using the LinearLayoutManager layout manager, for displaying the list of hot stems content
e. BottomBottomNavigationView, used to navigate the different application functions and contains a Floating Action Button (FAB) for adding new posts
f. A Fragment container at the bottom for embedding different Fragments.

Discovery page design

a. The top "Hottest" stalk, using the ImageView component. and the title and views, using the TextView component.
b. The RecyclerView component is used at the bottom to display other hot stems sorted by hotness.
c. The page design uses some resource files, such as @drawable, @menu, @color, etc., to set the background, icon, colour, etc. of the components.

Message page design

The message page uses a recyclerView to populate the message list cells, with a single message consisting of a contact avatar on the left and a contact name and message abbreviated on the right.

Personal page design

- Personal section at the top

i. On the left is the user's avatar, a RoundedImageView, a component from the third party library Makeramen, used to achieve the rounded rectangle effect.
ii. Next to the avatar is the user's username, and below the username are the number of

followers and favorites. These three components are constrained to fit different resolution devices by setting their position and size on the page.

- Below this is the menu bar section

i. This consists of four Button components used to customise the personal page menu bar.
ii. Each button contains an icon and a text label, and the layout_weight property allows them to be evenly distributed horizontally. 4 buttons are set in their position and size in the menu bar by a constraint relationship.
iii. The background colour of these buttons is transparent, the drawableLeft property is used to set the left-hand icon, the paddingStart property sets the spacing between the icon and the text, and the gravity property sets the left alignment of the text.

- The top and bottom sections are within two View components,

which are used to add a separator between the menu bar and the personal section. The top View component is bound to the bottom of the followers TextView component and the bottom View component is bound to the top of the bottom navigation bar.

- The bottom of the page is the navigation bar

Publishing page design

a. Use Button to set Cancel and Save to cancel the edit operation and save the draft operation.
b. The bottom use EditText component to set the title input field and content input field, set the widget can choose the input font size colour or insert pictures.
c. The bottom uses the ImageView to set the cover replacement.
d. The lowest part uses Button to set up the publish button.

Technical Difficulties and Challenges

During the development of this project, the main technical challenges we faced were:

1. adding jump between page Activities
2. binding the bottom navigation bar
3. using the Recyclerview component
4. Android ui interface design and implementation
5. Packaging apk and software testing

Jumping between pages of Activity is relatively simple, i.e. the page to be jumped is bound in the Manifest then call the intent function in the kt file, but adding transition animation to the Activity class is not simple. After consulting a lot of information we found that the Activity class provides an overridePendingTransition method that can be used to set the entry and exit animations. The xml file for the animation can then be set to achieve the desired animation transition.The design of the bottom navigation bar was the first difficulty encountered in the project development. When we designed the bottom navigation bar of the software, we looked up a lot of information through the internet, and when we checked the

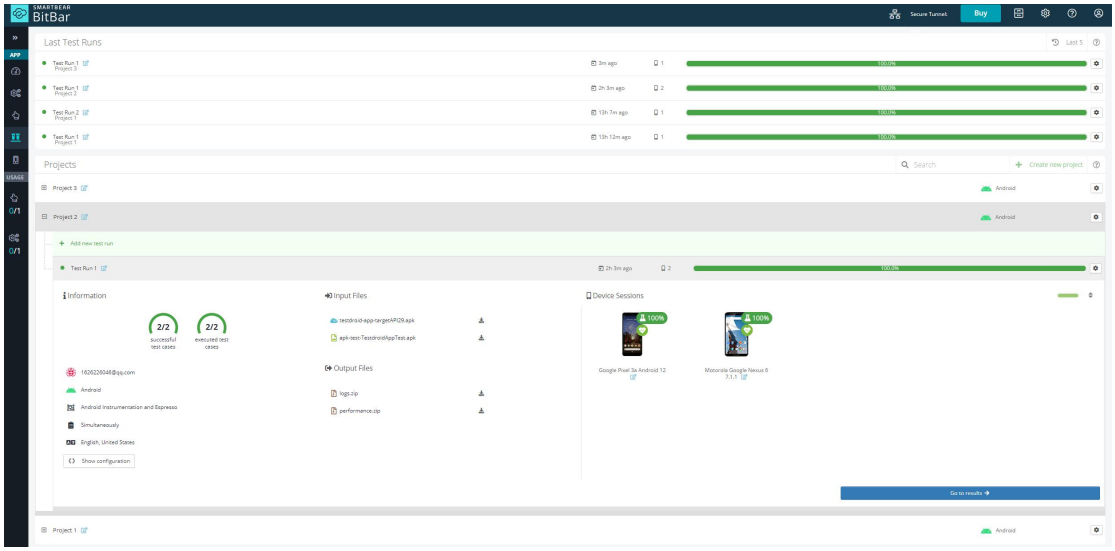
official documentation of Android Studio, we successfully found the relevant components of navigation and applied them to our project, but the shortcoming is that the default component is the Fragment instead of the Activity, so This led to a problem with the display of the bottom navigation bar when jumping from page to page in the app.

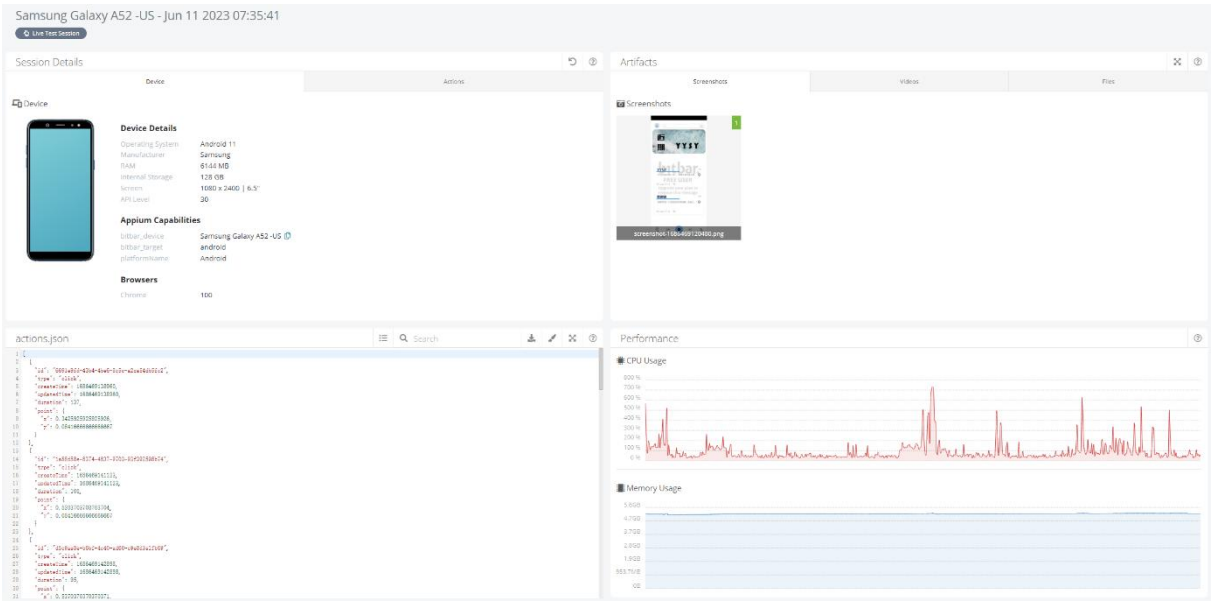
The most difficult part of this project was the requirement to use the Recyclerview component between different pages. Although the teacher had talked about the usage and principles of this component in class, there were still a lot of unknowns and doubts when it came to using the component in my own project and implementing it. One of the issues that plagued the team for a long time was the binding of the adapters in the Recyclerview, as the team members were vague about the usage of the adapters and the concept of how they work, so they repeatedly hit a brick wall when they first tried. However, after a long period of time, we finally overcame this major hurdle and with the experience of successfully implementing the first Recyclerview, the rest of the project was simply a matter of making it work.

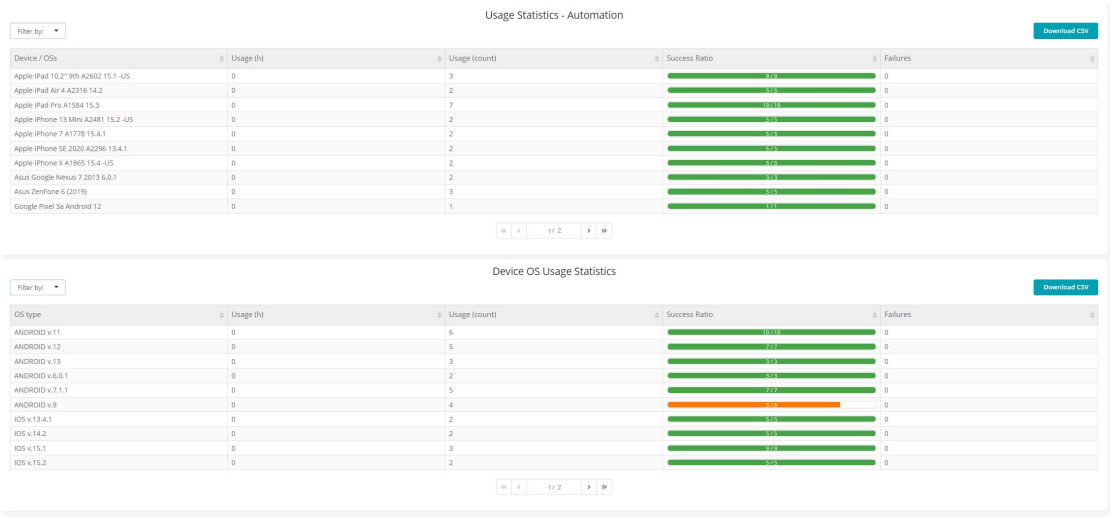
When designing the ui interface, we also encountered major challenges in terms of consistency of the page design language and the actual implementation. Firstly, the division of labour during the prototyping phase resulted in a large sense of fragmentation across pages, while many of the component templates had some styling restrictions, all of which affected the consistency of the design language to some extent. Therefore, we discarded many components and templates from the web and wrote a lot of the front-end code ourselves. For the navigation bar component used we also found an interface to modify its icon, content and colour, further unifying the design and style of the entire application.

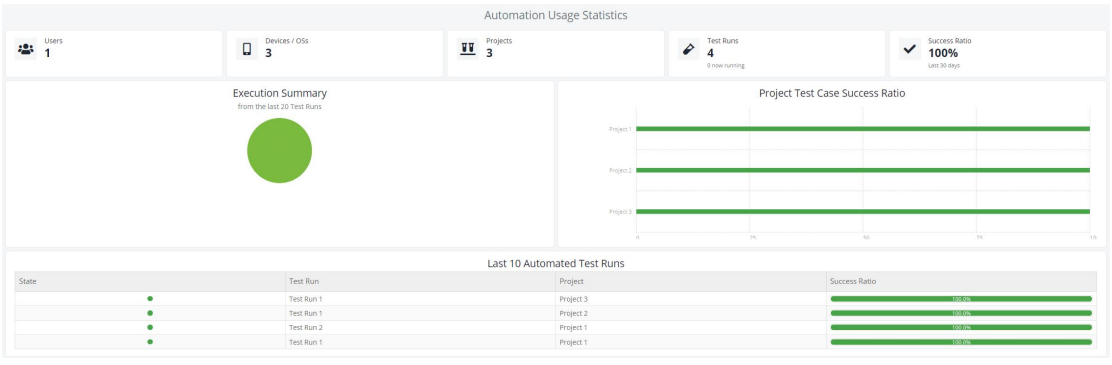
For software testing, we used the paid Bitbar testing tool, but the packaged apk file kept reporting errors. After four days of searching and modifying, by adjusting the sdk version and modifying the exported properties, we finally packaged the first installable and usable apk file. We passed various tests with Bitbar and achieved good results.

Testing and user experience analysis:



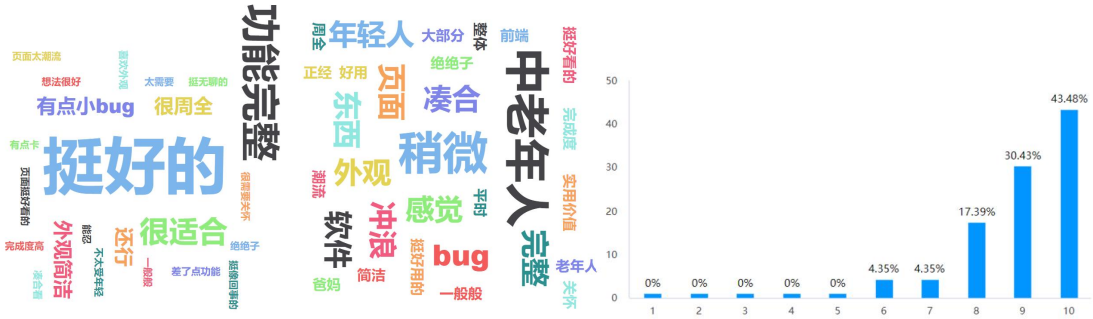

Samsung Galaxy A52 -US - Jun 11 2023 07:35:41
Live Test Session
Session Details
Device
Actions
Device Details
Operating System Android 11
Manufacturer Samsung
RAM 6144 MB
Internal Storage 128 GB
Screen 1080 x 2400 | 6.5"
API Level 30
Appium Capabilities
bitbar_device Samsung Galaxy A52 -US
bitbar_target android
platformName Android
Browsers
Chrome 100
Artifacts
Screenshots
Videos
Files
Screenshots
actions.json
Search
Performance
CPU Usage
Memory Usage


Usage Statistics - Automation
Filter by:
Download CSV
Device / OSs
Usage (h)
Usage (count)
Success Ratio
Failures
Apple iPad 10.2" 9th A2602 15.1 -US
Apple iPad Air 4 A2316 14.2
Apple iPad Pro A1584 15.3
Apple iPhone 13 Mini A2481 15.2 -US
Apple iPhone 7 A1778 15.4.1
Apple iPhone SE 2020 A2296 13.4.1
Apple iPhone X A1865 15.4 -US
Asus Google Nexus 7 2013 6.0.1
Asus ZenFone 6 (2019)
Google Pixel 3a Android 12
Device OS Usage Statistics
Filter by:
Download CSV
OS type
Usage (h)
Usage (count)
Success Ratio
Failures
ANDROID v.11
ANDROID v.12
ANDROID v.13
ANDROID v.6.0.1
ANDROID v.7.1.1
ANDROID v.9
IOS v.13.4.1
IOS v.14.2
IOS v.15.1
IOS v.15.2


Automation Usage Statistics
Users
1
Devices / OSs
3
Projects
3
Test Runs
4
0 now running
Success Ratio
100%
Last 30 days
Execution Summary
from the last 20 Test Runs
Project Test Case Success Ratio
Last 10 Automated Test Runs
State
Test Run
Project
Success Ratio
Test Run 1
Project 3
Test Run 1
Project 2
Test Run 2
Project 1
Test Run 1
Project 1

After we finished the development of the software, we packaged it into an apk and carried out many related software tests. At first there were many unexpected errors and various problems (sdk version, Android incompatibility issues, etc.). However, after each error, we were able to get through all the tests with no problems.

After passing the tests, we asked 23 students to fill out a user experience questionnaire and the results showed that 90 per cent of them thought that the software had good basic functions, the UI design was reasonable and user-friendly, and the overall feeling of using the software was good. Four per cent thought that some of the bugs in the software had some impact on the experience. Overall, our software meets the expectations of the general public, and although it has some bugs, it is a relatively good software.



Project Summary

Generally, this project has completed the entire process of developing a software product from idea generation, feasibility analysis and market analysis, to prototyping, ui design, front-end interface implementation, front-end interaction logic implementation, to data import, software testing and internal user feedback. The product has a high degree of completion, a rich and beautiful interface, and has the characteristics of bridging the market demand gap, which is not available in many other projects.

During the development process, as we were still relatively new to the use of new technologies, we encountered high learning costs, high trial and error costs, and frequent vulnerabilities, which affected the development efficiency to a certain extent. When it came to adding ui animations, we found that the Fragment class was much easier to implement than the Activity class and would be more compatible with some existing components. Due to our inexperience, we had a lot of trouble with this as we had used a lot of Activity classes before this. Prior to the use of the division of labour flow chart, there was still a large amount of unreasonable work distribution in our group, resulting in a gap in workload for some members of the group. When prototyping the front-end implementation, we found that Android's style design was less flexible and differed significantly from the CSS style design we were familiar with, which caused some trouble in our development. However, in the end, through extensive internet research, we were able to design a more consistent ui interface without applying a template. We also encountered many problems when doing software

testing, and after trying about seven testing sites, we ended up using the bitbar testing tool which requires a fee. Just upload the packaged apk file and test code.

In future Android development, we should supplement our front-end ui pre-knowledge well in advance to avoid as much ineffective trial and error during development as possible. We should start implementing a division of labour table much earlier to monitor the progress of each team member in real time. In the prototyping phase, we should also start with a unified design language or make the division of labour more focused to avoid a sense of fragmentation between different pages.