Testing Documentation

ArtistConnect

An app that allows artists and fans to connect and collaborate

Student 1 - Karl Doherty 19413086

Student 2 - Yury Chupahin 19416764

Project Supervisor: Dr Gareth Jones

Ad-hoc testing

Ad hoc testing is a testing approach that involves randomly testing different features and functionalities of the application without following any specific test plan or script. This type of testing is usually done by testers who are experienced and familiar with the application, as they can quickly identify any issues or bugs that may arise during the testing process.
In our social media app, we conducted ad hoc testing by randomly exploring different features such as the login and registration processes, post creation and commenting, user profile pages, and search functionalities. We also tested the application on different devices with varying screen sizes to ensure that it works well on all types of devices.

During the ad hoc testing process, we discovered several issues such as slow loading times, broken links, and inconsistencies in the UI design. We also identified several functional issues such as errors in the search functionality and issues with the post creation process.

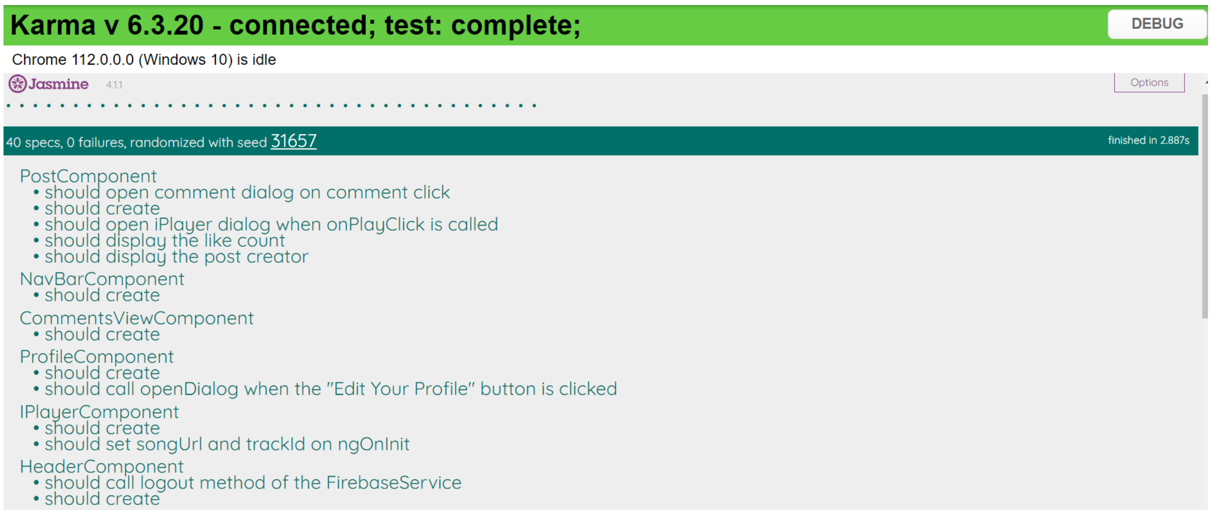
To address these issues, we worked closely to fix the bugs and improve the overall performance and user experience of the application. We also added additional test cases to our test plan to ensure that these issues do not reoccur in the future.

Overall, ad hoc testing proved to be a valuable testing approach for our social media app as it helped us identify several critical issues that we might have missed during structured testing. By incorporating ad hoc testing into our testing process, we were able to improve the quality and usability of our application, ultimately resulting in a better user experience for our customers.

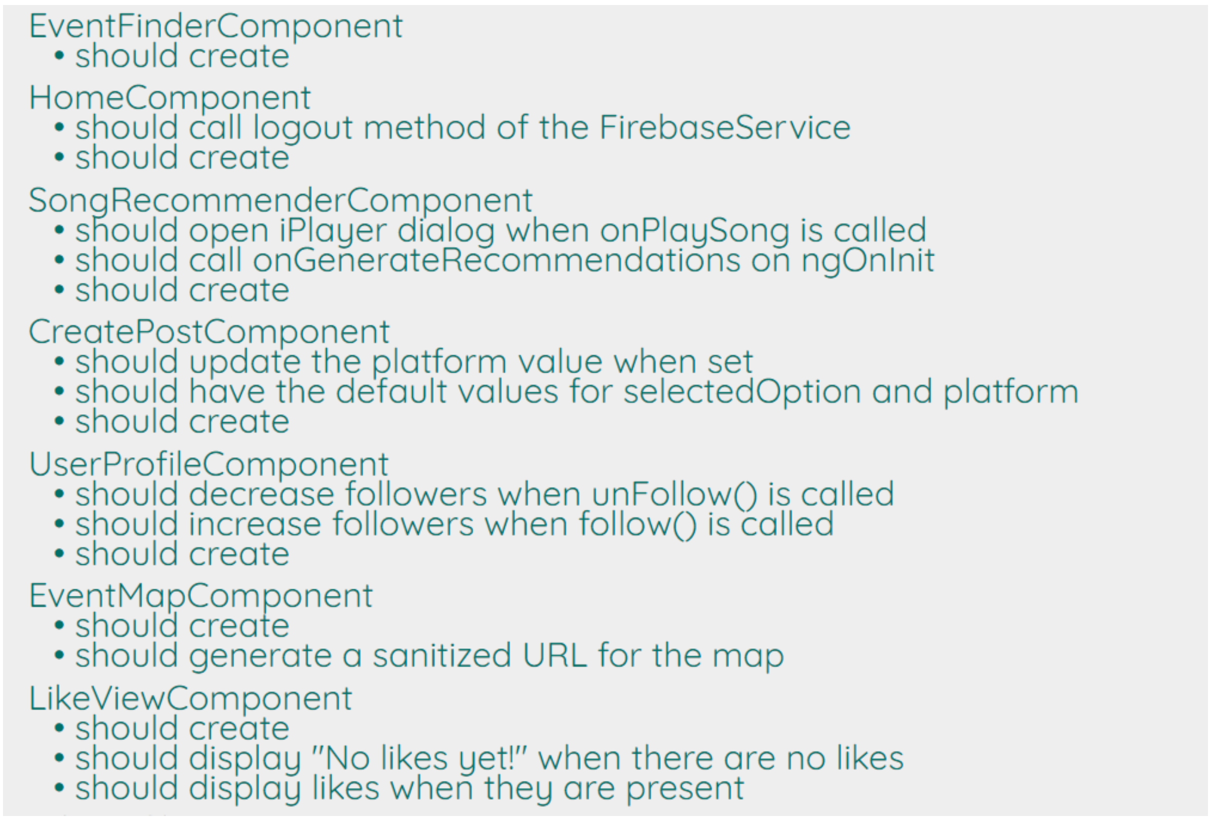
Unit testing

Unit testing is an essential part of the testing process for any application. To ensure that our app is functioning as expected, we have unit tested every component to make sure that it starts up correctly. We used Jasmine and Karma, which are widely used frameworks for unit testing in Angular. We also created unit tests for some of the major functions in the components to ensure that they are working correctly.

Unit testing is particularly useful for catching bugs early in the development process, before they have a chance to cause bigger problems. By testing individual components and functions in isolation, we can identify and fix issues quickly, without having to go through the entire application. This saves us a lot of time and effort in the long run, and helps us to deliver a high-quality app that is reliable and robust. Additionally, unit tests act as a form of documentation, as they provide insight into how individual components and functions should work, making it easier for new developers to understand and work on the codebase. Overall, unit testing is a valuable technique that helps us ensure the quality and reliability of our social media app.



```
Karma v 6.3.20 - connected; test: complete;                                DEBUG
Chrome 112.0.0.0 (Windows 10) is idle
Jasmine 4.1.1                                                              Options
........................................
40 specs, 0 failures, randomized with seed 31657                finished in 2.887s
PostComponent
  • should open comment dialog on comment click
  • should create
  • should open iPlayer dialog when onPlayClick is called
  • should display the like count
  • should display the post creator
NavBarComponent
  • should create
CommentsViewComponent
  • should create
ProfileComponent
  • should create
  • should call openDialog when the "Edit Your Profile" button is clicked
IPlayerComponent
  • should create
  • should set songUrl and trackId on ngOnInit
HeaderComponent
  • should call logout method of the FirebaseService
  • should create
```



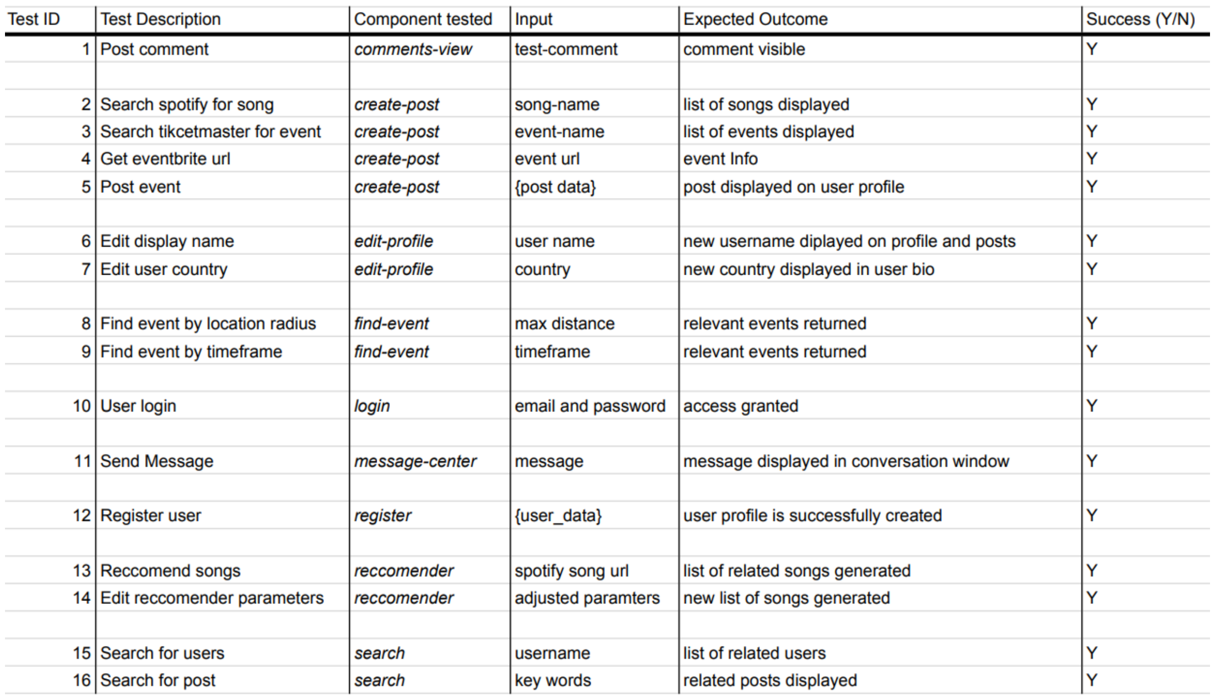
```
EventFinderComponent
  • should create
HomeComponent
  • should call logout method of the FirebaseService
  • should create
SongRecommenderComponent
  • should open iPlayer dialog when onPlaySong is called
  • should call onGenerateRecommendations on ngOnInit
  • should create
CreatePostComponent
  • should update the platform value when set
  • should have the default values for selectedOption and platform
  • should create
UserProfileComponent
  • should decrease followers when unFollow() is called
  • should increase followers when follow() is called
  • should create
EventMapComponent
  • should create
  • should generate a sanitized URL for the map
LikeViewComponent
  • should create
  • should display "No likes yet!" when there are no likes
  • should display likes when they are present
```

Intergration testing

Integration testing is an essential part of any software testing process. It helps ensure that the different components of the application are working together correctly. For our application, we performed manual integration testing to check if the application was correctly running with Firebase.

Although we did not have automatic tests for integration testing, we made sure to test certain features of the application manually. This included testing the functionality of the registration and login processes, creating and viewing posts, commenting on posts, editing profiles, and searching for other users, posts, and events.

We also added test cases to our spreadsheet to keep track of the manual tests performed and their results. This allowed us to document the testing process and make sure that all features were thoroughly tested before releasing the application.
While automatic testing is ideal for integration testing, manual testing can still be effective in finding bugs and issues. By performing manual integration testing, we were able to identify and fix issues with the application, ensuring a smooth and error-free user experience



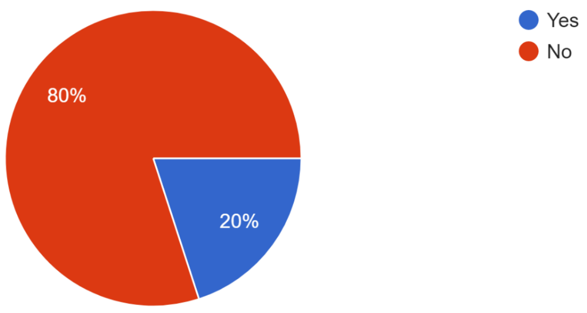
Test ID	Test Description	Component tested	Input	Expected Outcome	Success (Y/N)
1	Post comment	*comments-view*	test-comment	comment visible	Y
2	Search spotify for song	*create-post*	song-name	list of songs displayed	Y
3	Search tikcetmaster for event	*create-post*	event-name	list of events displayed	Y
4	Get eventbrite url	*create-post*	event url	event Info	Y
5	Post event	*create-post*	{post data}	post displayed on user profile	Y
6	Edit display name	*edit-profile*	user name	new username diplayed on profile and posts	Y
7	Edit user country	*edit-profile*	country	new country displayed in user bio	Y
8	Find event by location radius	*find-event*	max distance	relevant events returned	Y
9	Find event by timeframe	*find-event*	timeframe	relevant events returned	Y
10	User login	*login*	email and password	access granted	Y
11	Send Message	*message-center*	message	message displayed in conversation window	Y
12	Register user	*register*	{user_data}	user profile is successfully created	Y
13	Reccomend songs	*reccomender*	spotify song url	list of related songs generated	Y
14	Edit reccomender parameters	*reccomender*	adjusted paramters	new list of songs generated	Y
15	Search for users	*search*	username	list of related users	Y
16	Search for post	*search*	key words	related posts displayed	Y

User testing

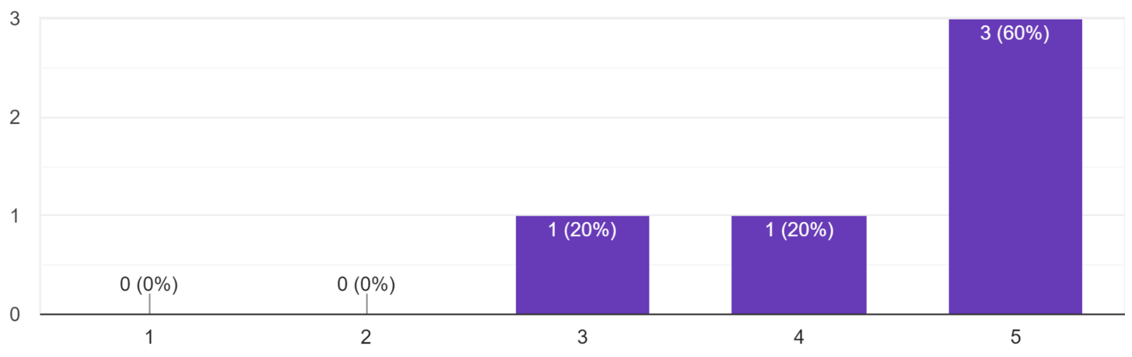
User testing is an important aspect of testing in software development as it helps to ensure that the application is user-friendly and meets the needs of the target audience. In our project, we conducted several rounds of user testing to gather feedback and insights from real users.
We started with an initial UI testing phase where we created prototypes of the application and had a group of users interact with it. This allowed us to see what features and design elements were effective and what needed to be improved.
As we progressed with development, we invited a random selection of users to use the app in its nearly finished state and fill out a survey after completion. This helped us to identify any usability issues, bugs, and areas for improvement.
Additionally, we gave users specific tasks to complete while we observed and recorded their behaviours. This allowed us to see how easily users were able to navigate the application and complete common actions, such as creating a post or searching for other users.

Overall, user testing was a valuable tool in our development process as it helped us to create an application that is more user-friendly and tailored to the needs of our target audience.

Did you encounter any errors while managing your account?
5 responses

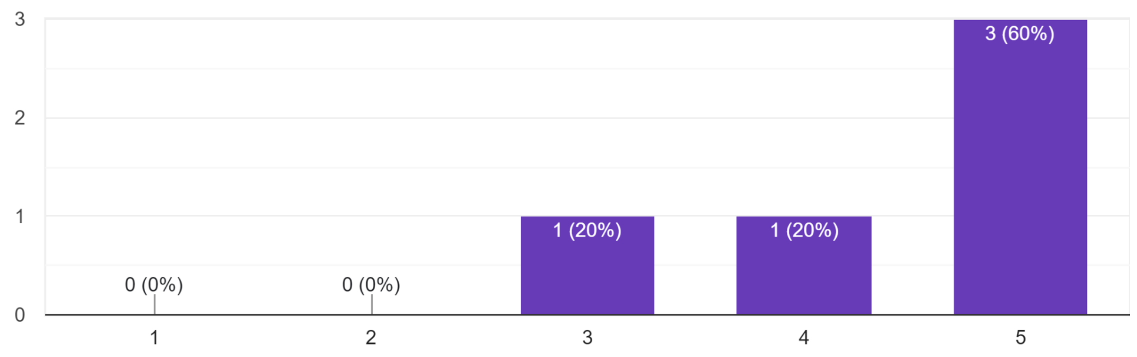


How satisfied were you with the available search options for finding users or content?
5 responses



How accurate did you find the recommender system of the application?

5 responses



3
2
1
0
0 (0%)
0 (0%)
1 (20%)
1 (20%)
3 (60%)
1
2
3
4
5