Name: Abhinav Gupta Roll No.: 190050003 Branch: CSE

DevCom
Recruitment Assignment

Android Assignment:

Research:

1.

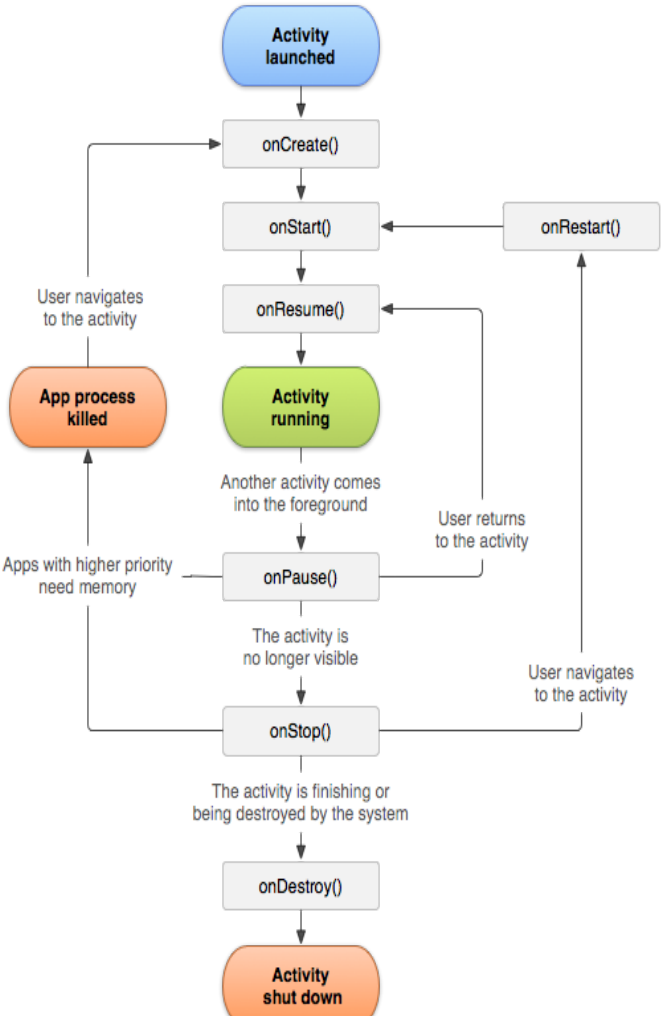
a. **Android** is a mobile operating system based on a modified version of the Linux kernel and other open-source software, designed primarily for touchscreen mobile devices such as smartphones and tablets.
Android is developed by a consortium of developers known as the **Open Handset Alliance**, with the main contributor and commercial marketer being **Google**.
An **Android app** is a software application running on the Android platform. Because the Android platform is built for mobile devices, a typical Android app is designed for a smartphone or a tablet PC running on the Android OS.

b. The most popular software for android app development is **Android Studio**. The software needs to have the **Android Software Development Kit (SDK),** it officially supports Android Studio only. But there are also other third party softwares which are better for android game development etc.
Android SDK includes a comprehensive set of development tools. These include a debugger, libraries, a handset emulator based on QEMU, documentation, sample code, and tutorials.

c. Android apps can be developed in Kotlin, JAVA, C++ languages with the help of Android SDK. These are the options we get in Android Studio.
InstiApp uses **JAVA** for frontend.

d. As a user navigates through, out of, and back to our app, the **Activity** instances in our app transition through different states in their lifecycle. The **Activity** class provides a number of callbacks that allow the activity to know that a state has changed: that the system is creating, stopping, or resuming an activity, or destroying the process in which the activity resides.

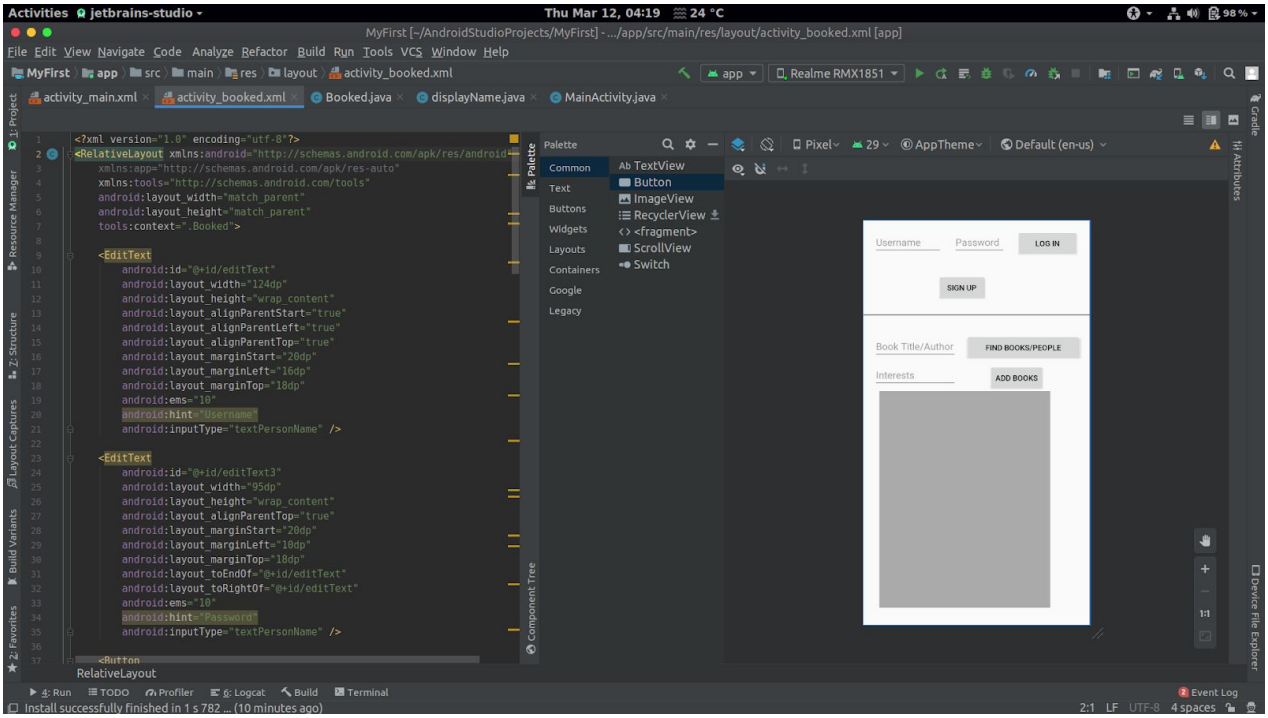
e. UI elements are Button, CheckBox, SeekBar, RadioButton, EditText, TextView.



Name: Abhinav Gupta Roll No.: 190050003 Branch: CSE

f. **Java** is platform-independent as Java uses compiler and interpreter both. Java source code is converted into bytecode by the compiler(bytecode is same for all platforms, all we need is Java Virtual Machine to interpret) which is interpreted by the interpreter during run-time. The Java platform is designed with multithreading capabilities built into the language. That means we can build applications with many concurrent threads of activity, resulting in highly interactive and responsive applications. Java and C++ are both object-oriented languages and have somewhat similar syntax.
Kotlin, in most cases, is handier than Java as it resolves some Java issues and has some more features like string templates, lambda functions, operator-overloading, smart-casts.

Task:

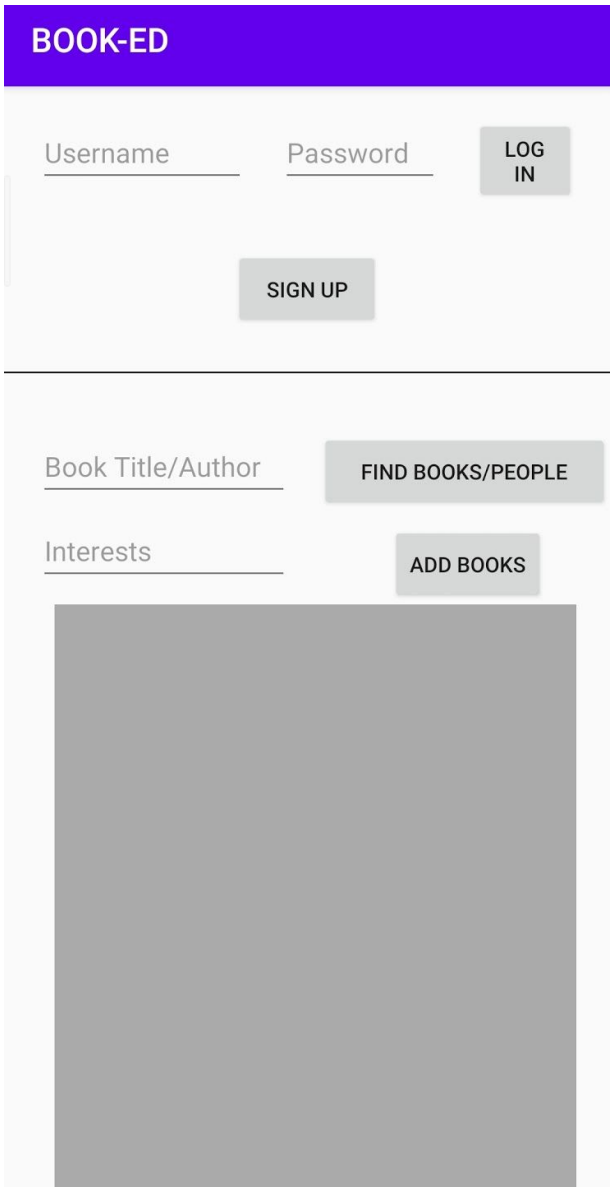
1. **LinearLayout** organizes elements along a single line. We can specify whether that line is vertical or horizontal using certain attributes. The orientation is horizontal by default. **RelativeLayout** is a view group that displays child views in relative positions. The position of each view can be specified as relative to sibling elements (such as to the left-of or below another view) or in positions relative to the parent **RelativeLayout** area (such as aligned to the bottom, left or center).
2. Features that the screen must include are as follows:
    - SignUp and Login buttons(with the username and password fields).
    - Search field for books based on title/author and a search field for people with common interests (user can input particular interests but it is not supposed to be necessary if the user has logged in.
    - A ListView that displays the search results of both the types.
    - I have also used a horizontal divider to differentiate between login and search areas.
3. 

Name: Abhinav Gupta Roll No.: 190050003 Branch: CSE

4. The given photo of interphase consists of Buttons, PlainTexts, and ListView. Subdivision of views in these categories is apparent from the image. And I have used **RelativeLayout** in generating this interface, not **LinearLayout**, by which I was not able to generate differently aligned views. With RelativeLayout also things were not as smooth as **ConstraintLayout** feature of AndroidStudio.

5. A pic of my first app in my device: Interface only(till now)

6. XML code can be seen in the repository. Although most of the XML code is generated by the software itself, I just dragged the Views and performed tweaks only in the XML code.



Backend Assignment:

Research:

1.
    a. Cycle of events is as follows:
        - We enter a URL into a web browser

- The browser looks up the IP address for the domain name via DNS
- The browser sends a HTTP *request* to the server
- The server sends back a HTTP *response*
- The browser begins rendering the HTML
- The browser sends requests for additional objects embedded in HTML (images, CSS, JavaScript) and repeats steps 3-5.
- Once the page is loaded, the browser sends further async requests as needed.

b. Each URL have a unique IP address of the server on which it is being hosted(It may be in Banglore or any other place). The mapping of URL to the IP address is done by Domain Name System (DNS). A browser gets resources only after getting IP address of the hosting server. So browser searches through various caches to find the corresponding IP address, if it can't find a match then the ISP server puts a DNS query on Internet to find the IP address.

c. Network layer uses IP or Internet Protocol which being a connection less protocol treats every packet individually and separately leading to lack of reliability during a transmission. When data is sent from one host to another, each packet may take a different path even if it belongs to the same session. This means the packets may/may not arrive in the right order. Therefore, IP relies on the higher layer protocols to provide reliability.
   - **TCP** is a layer 4 protocol which provides acknowledgement of the received packets and is also reliable as it resends the lost packets.
   - **UDP** is also a layer 4 protocol but unlike TCP it doesn't provide acknowledgement of the sent packets. Therefore, it isn't reliable and depends on the higher layer protocols for the same. It is used in video and voice streaming.

d. When we visit websites on the internet, they are each hosted by a "**server**". A **server** is a computer located somewhere in the world that's connected to the internet, and that computer's job is to "serve" webpages to internet users that want to view them.
   So **setting up a server** would imply that we have some hardware and/or software that resolves requests sent by clients by transmitting required data that the client has requested, following some standard protocols.

e. Apache and NginX are web servers, they act as intermediates between the servers and clients.
   When a visitor wants to load a page on our website, for instance, their browser sends a request to our server and Apache returns a response with all the requested files (text, images, etc.). The server and the client communicate through the HTTP protocol and Apache is responsible for the smooth and secure communication between the two machines.

NginX works almost the same as Apache but can deal with more than 10k requests at the same time as it has a thread-based structure. On the other hand, NginX treats every request in a single thread.

f. The friend's device must be connected to the same network, and then typing *IP_address/filename* (where IP address is of my computer) will open the HTML webpage in that device.

Think:

1. The code for models.py is in the repository.
2.
   a. If we create 2 more models, one Author and other for relation between author and books (similar as in bodies file), we can find the books by following queries:

      ```
      findauthor = Author.objects.filter(name = 'authorname')
      AuthorBookRelation.objects.filter(Author = findauthor)
      ```

      These queries will give us the query set of books with this particular author
      .
   b. Code in the repository under "models_task2.py"
3.

Initially all events have weight of 1000 units.

a. Max. priority is given to the upcoming events (i.e. **the events that are going to happen most recently have the highest priority**), the factor by which the weight of the event is affected is calculated by a **Gaussian** centered at O, with scaled remaining time as input.

Next, a bonus is given to events if the user follows the conducting body. This contains 2 terms time-dependent as well as a time-independent factor(this is multiplied by start time factor which is the same as above).

If **the event is ended then it is given the max penalty of 600 units**, followed by a further exponential penalty which is more if the event have ended earlier.

If an **event has not been tagged to a restricted audience, then also the event is given a penalty** which is directly proportional to difference of tags of the event to the total tags defined for users.

If **the event is introduced much earlier then also a mild linear penalty is given.**

Longer events' weight is also depreciated by a factor dependent on their length.

Bonus promotion is given to events if there is some promotion factor specified on creation of the event.

b. We can define a genre field in Books model. Also, we will define a dictionary associated with a user which counts the number of times a particular genre is clicked by the user. And we can display the feed according to decreasing order of clicks on genres, further, we can extend this on authors also as we have fields asociated with them also.