Lets Browse : An Android App

1. PROJECT SYNOPSIS

1.1 Title of the Project:-

Lets Browse

1.2 Objective of the Project:-

Lets Browse is a web browser based on Android Platform where users can easily access their desired websites using a beautiful User Interface. Users can use our application for saving websites as PDF and multi Tab Web Surfing and to view any site as a desktop view. Lets Browser is an app focused on quick search with google search as the Default Search Engine.

1.3 Project Category :-

Android application

1.4 Language(s) to be used:-

Kotlin
XML

1.5 HARDWARE AND SOFTWARE REQUIREMENTS:-

1.5.1 HARDWARE REQUIREMENTS:-

- ➢ Smartphone with Android
- ➢ Processor Frequency– greater than 1.40 GHz
- ➢ RAM – 1 GB or above.

1.5.2 SOFTWARE REQUIREMENTS :-

- ➢Android Studio for development
- ➢Android version greater than 4.0.1

1.5.3 ASSUMPTIONS and DEPENDENCIES:-

- ➢The users should have basic knowledge of the Smartphones. They must be trained well to handle the features provided by the Smartphones.
- ➢Website URL is entered by the user and may not be generated automatically.

1.6 Specific requirements:

1.6.1 External Interface Requirements: -

This chapter is an outline of the inputs and outputs of the project.

1.6.2 User Interfaces:-

Each part of the user interface intends to be as user friendly as possible. The fonts and buttons used will be intended to be very fast and easy to load on web pages. The pages will be kept light in space so that it won't take a long time for the page to load. All the buttons are provided on FAB (Floating Action Button) so that users can easily access all the controls of the Web Browser.

1.6.3 Hardware Interfaces:

- Operating System: Linux
- Processor: Pentium or Higher.
- RAM: 1 GB or more

1.6.4 Software Interfaces: -

- Development tool/IDE:Android Studio
- WebView
- WebChromeClient

2 Functional requirements:-

2.1 Bookmark Webpage:-

A bookmark is a web browser feature used to save a web site's URL address for future reference. Bookmarks save user and browser time, which is especially useful for Web pages with long URLs or accessing a specific part of the site that might not be the homepage for the site.

2.2 Save Webpage as Pdf:-

PDF (Portable Document Format) is a universal file format that can be viewed by anyone, even if they don't have the software that created the file. When the Page is Loaded then it Doesn't require the Internet and You can save your Web Pages as PDF files.

2.3 Desktop Site Support:-

Any website Can be Viewed in Desktop Mode Because some sites do not support mobile viewing or the mobile view is not correctly showing the User Interface of the website then the user can switch to desktop view to Properly Utilise the User Interface and get his/her desired task done.

2.4 Full Screen browser:-

This allows the user to hide the notification bar and provides an immersive Experience to the user while accessing his desired web resource.This can also be disabled in case the user wants to access his notification bar to view time or any notification of their interest.

2.5 Multi Tabs Support:-

This Browser supports multiple tabs to be accessed in a Single Session. Users can switch between different tabs of his choice at different points of time according to their Desired Website.Users are not forced to close the current site in order to access any other site .

3 System features:-

The coding is done with following characteristics in mind:

- Ease of design to code translation
- Code efficiency
- Memory efficiency
- Response time
- Maintainability
- Security
- Efficient and consistent logic

4 Other Non-Functional Requirements:

4.1 Performance Requirements:

Performance requirements define acceptable response times for system functionality.

• The system is supposed to be having good memory space and RAM should be Above 1GB preferably.

• The sound card and graphics card will have to be of good quality and capacity.

• The load time for user interface screens shall take no longer than three seconds.

• The log in information shall be verified within three seconds.

• Queries shall return results within three seconds.

4.2 Safety Requirements:

• The Bookmarks stored in the database in encrypted format.

4.3 Security Requirements:

• No Unauthorized Person can access the Web Browser from remote Locations.

4.4 Software quality attributes:

• Reliability: This Application is reliable enough to cater to multiple web page requests.

• Availability: This Application will only be available till the system on which it is installed is running.

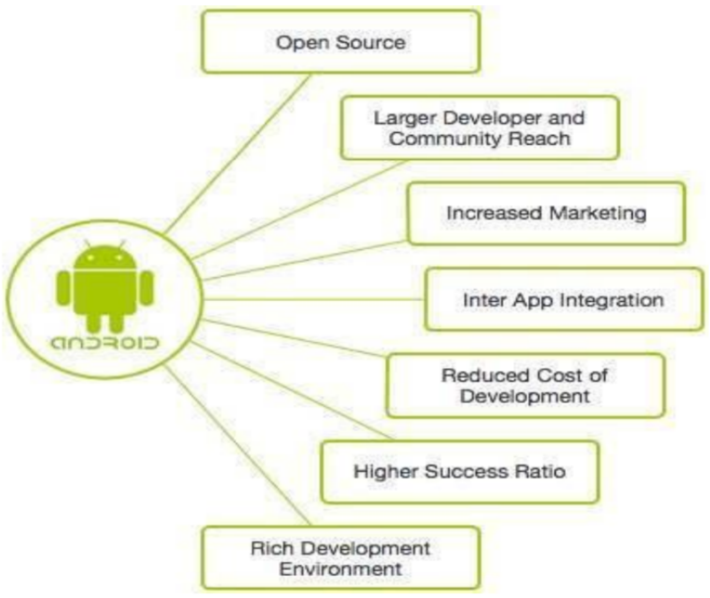
- Security: Application is secured with https Protocol So the requests made are secured. Android Apps are inherently secure because Applications run inside Virtual Machine Sandbox.
  - Classloader: Class loader in Java is a part of the Java Runtime Environment (JRE) which is used to dynamically load Java classes into the Java Virtual Machine. It adds security by separating the package for the classes of the local file system from those that are imported from network sources.
  - Bytecode Verifier: It checks the code fragments for illegal code that can violate access rights to objects.
  - Security Manager: It determines what resources a class can access such as reading and writing to the local disk.
- Object-oriented: Kotlin is an object-oriented programming language. Everything in Kotlin is an object. Object-oriented means we organize our software as a combination of different types of objects that incorporates both data and behaviour. Object-oriented programming (OOPs) is a methodology that simplifies software development and maintenance by providing some rules.

- Robust: Robust simply means strong. Kotlin is robust because:
  - It uses strong memory management.
  - There is a lack of pointers that avoids security problems.
  - There is automatic garbage collection in java which runs on the Java Virtual Machine to get rid of objects which are not being used by a Java application anymore.
  - There is exception handling and type checking mechanism in java. All These points make Kotlin robust.

5 Technology Used:

ANDROID :-

Android is a mobile operating system (OS) currently developed by Google, based on the Linux kernel and designed primarily for touchscreen mobile devices such as smartphones and tablets.


Open Source
Larger Developer and Community Reach
Increased Marketing
Inter App Integration
Reduced Cost of Development
Higher Success Ratio
Rich Development Environment
android

Features of Android

Android user interface is mainly based on direct manipulation, using touch gestures that loosely correspond to real-world actions, such as swiping, tapping and pinching, to manipulate on-screen objects, along with a virtual keyboard for text input. In addition to touchscreen devices, Google has further developed Android TV for televisions, Android Auto for cars, and Android Wear for wrist watches, each with a specialized user interface.

Initially developed by Android, Inc., which Google bought in 2005, Android was unveiled in 2007, along with the founding of the Open Handset Alliance – a consortium of hardware, software, and telecommunication companies devoted to advancing open standards for mobile devices. The Google Play store has had over one million Android applications (“app”) published, and over 50 billion applications downloaded.

Features of Android:-
Android is a powerful operating system competing with Apple iOS and supports great features.Few of them are listed below :−

- **Beautiful UI:** Android OS basic screen provides a beautiful and intuitive user interface.

- **Connectivity:**GSM/EDGE, IDEN, CDMA, EV-DO, UMTS, Bluetooth, Wi-Fi, LTE,5G,NFC and WiMAX.

- **Storage:** SQLite, a lightweight relational database that is used for data storage purposes.

- **Web browser:** Based on the open-source Web Kit layout engine, coupled with Chrome's V8 JavaScript engine supporting HTML5 and CSS3.

- **Multi-touch:** Android has native support for multi-touch which was initially made available in handsets such as the HTC Hero.

- **Multi-tasking:** User can jump from one task to another and same time various application can run simultaneously.

- **GCM:** Google Cloud Messaging (GCM) is a service that lets developers send short message data to their users on Android devices, without needing a proprietary sync solution.

- **Android Beam:** A popular NFC-based technology that lets users instantly share, just by touching two NFC-enabled phones together.

ANDROID STUDIO:

Android Studio is an integrated development environment (IDE) for developing for the Android platform. It was announced on May 16, 2013 at the Google I/O conference. Android Studio is freely available under the Apache License 2.0.

Based on JetBrains IntelliJ IDEA software, Android Studio is designed specifically for Android development. It is available for download on Windows, Mac OS X and Linux, and replaced Eclipse Android Development Tools (ADT) as Google's primary IDE for native Android application development.

Android Studio Logo

Programming Languages used in Developing Android Applications:

1. Java

2. Kotlin

Developing the Android Application using Kotlin is preferred by Google, as Kotlin is made an official language for Android Development, which is developed and maintained by JetBrains. Previously Java was considered the official language for Android Development. Kotlin is made official for Android Development in Google I/O 2017.

Android Core Building Blocks:-

An android component is simply a piece of code that has a well-defined life cycle e.g. Activity, Receiver, Service etc.

The core building blocks or fundamental components of android are activities, views, intents, services, content providers, fragments and AndroidManifest.xml.

Activity:

An activity is a class that represents a single screen. It is like a Frame in AWT(Abstract Window Toolkit).

View:

A view is the UI element such as button, label, text field etc. Anything that you see is a view.

Intent:

Intent is used to invoke components.

Service:

Service is a background process that can run for a long time. There are two types of services: local and remote. Local service is accessed from within the application whereas remote service is accessed remotely from other applications running on the same device.

Content Provider:

Content Providers are used to share data between the applications.

Fragment:

Fragments are like parts of activity. An activity can display one or more fragments on the screen at the same time.

AndroidManifest.xml:

It contains information about activities, content providers, permissions etc. It is like the web.xml file in Java EE.

Android Virtual Device (AVD):

It is used to test the android application without the need for mobile or tablet etc. It can be created in different configurations to emulate different types of real devices.

Android User Interface:-

In an Android application, the user interface is built using View and ViewGroup objects. There are many types of views and view groups, each of which is a descendant of the View class.

View objects are the basic units of user interface expression on the Android platform. The View class serves as the base for subclasses called "widgets," which offer fully implemented UI objects, like text fields and buttons. The ViewGroup class serves as the base for subclasses called "layouts," which offer different kinds of layout architecture, like linear, tabular and relative.

A View object is a data structure whose properties store the layout parameters and content for a specific rectangular area of the screen. A View object handles its own measurement, layout, drawing, focus change, scrolling, and key/gesture interactions for the rectangular area of the screen in which it resides. As an object in the user interface, a View is also a point of interaction for the user and the receiver of the interaction events.

Android Layout:-

Android Layout is used to define the user interface that holds the UI controls or widgets that will appear on the screen of an android application or activity screen. Generally, every application is a combination of View and ViewGroup.

A layout defines the structure for a user interface in your app, such as in an activity. All elements in the layout are built using a hierarchy of View and ViewGroup objects. A View usually draws something the user can see and interact with. Whereas a ViewGroup is an invisible container that defines the layout structure for View and other ViewGroup objects, as shown in figure 1.

View Hierarchy:-

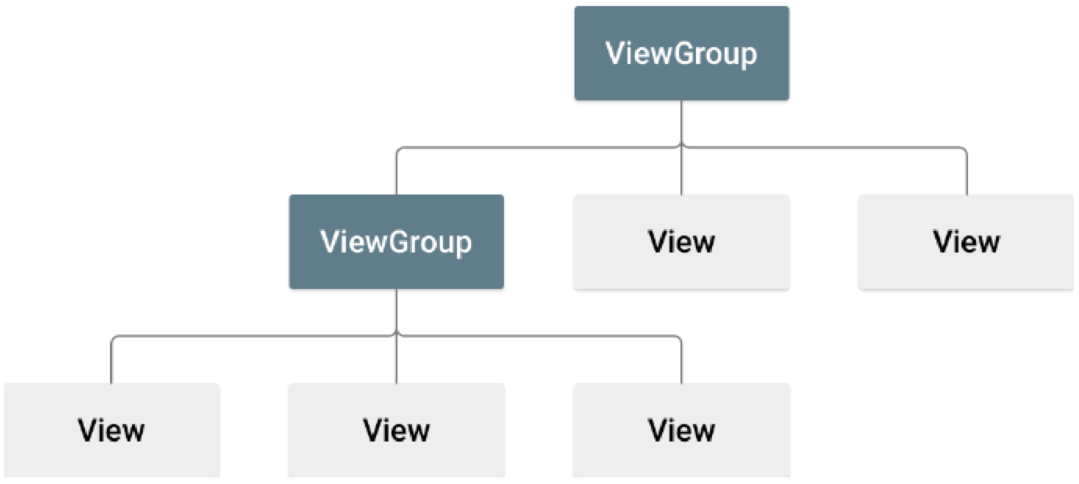


Figure 1. Illustration of a view hierarchy, which defines a UI layout

Adapters

Sometimes you'll want to populate a view group with some information that can't be hard-coded, instead, you want to bind your view to an external source of data. To do this, you use an AdapterView as your view group and each child

View is initialized and populated with data from the Adapter. The AdapterView object is an implementation of ViewGroup that determines its child views based on a given Adapter object. The Adapter acts like a courier between your data source (perhaps an array of external strings) and the AdapterView, which displays it. There are several implementations of the Adapter class, for specific tasks, such as the CursorAdapter for reading database data from a Cursor, or an ArrayAdapter for reading from an arbitrary array.

Styles and Themes

Perhaps you're not satisfied with the look of the standard widgets. To revise them, you can create some of your own styles and themes.

- A style is a set of one or more formatting attributes that you can apply as a unit to individual elements in your layout. For example, you could define a style that specifies a certain text size and color, then apply it to only specific View elements.
- A theme is a set of one or more formatting attributes that you can apply as a unit to all activities in an application, or just a single activity. For example, you could define a theme that sets specific colors for the window frame and the panel background, and sets text sizes and colors for menus. This theme can then be applied to specific activities or the
- entire application.

Styles and themes are resources. Android provides some default style and theme resources that you can use, or you can declare your own custom style and theme resources.

USE OF KOTLIN LANGUAGE

WHAT IS KOTLIN ?

Kotlin is an open-source, statically-typed programming language that supports both object-oriented and functional programming. Kotlin provides similar syntax and concepts from other languages, including C#, Java, and Scala, among many others. Kotlin does not aim to be unique—instead, it draws inspiration from decades of language development. It exists in variants that target the JVM (Kotlin/JVM), JavaScript (Kotlin/JS), and native code (Kotlin/Native).

Kotlin is an expressive and concise programming language that reduces common code errors and easily integrates into existing apps.

Why is Android development Kotlin-first?

We reviewed feedback that came directly from developers at conferences, our Customer Advisory Board (CAB), Google Developers Experts (GDE), and through our developer research. Many developers already enjoy using Kotlin, and the request for more Kotlin support was clear. Here's what developers appreciate about writing in Kotlin:-

- **Expressive and concise:** You can do more with less. Express your ideas and reduce the amount of boilerplate code. 67% of professional developers who use Kotlin say Kotlin has increased their productivity.
- **Safer code:** Kotlin has many language features to help you avoid common programming mistakes such as null pointer exceptions. Android apps that contain Kotlin code are 20% less likely to crash.
- **Interoperable:** Call Java-based code from Kotlin, or call Kotlin from Java-based code. Kotlin is 100% interoperable with the Java programming language, so you can have as little or as much of Kotlin in your project as you want.
- **Structured Concurrency:** Kotlin coroutines make asynchronous code as easy to work with as blocking code. Coroutines dramatically simplify background task management for everything from network calls to accessing local data.

What does Kotlin-first mean?

When building new Android development tools and content, such as Jetpack libraries, samples, documentation, and training content, we will design them with Kotlin users in mind while continuing to provide support for using our APIs from the Java programming language.

Comparison to Java:

Some Java issues addressed in Kotlin:

Kotlin fixes a series of issues that Java suffers from:

1. Null references are controlled by the type system.
2. No raw types
3. Arrays in Kotlin are invariant(This means that Kotlin does not let us assign an Array<String> to an Array<Any>, which prevents a possible runtime failure)
4. Kotlin has proper function types, as opposed to Java's SAM-conversions
5. Use-site variance without wildcards
6. Kotlin does not have checked exceptions

What Kotlin has that Java does not:

1. Lambda expressions + Inline functions = performant custom control structures
2. Extension functions
3. Null-safety
4. Smart casts
5. String templates
6. Properties
7. Primary constructors
8. First-class delegation
9. Type inference for variable and property types
   Singletons
10. Declaration-site variance & Type projections
11. Range expressions
12. Operator overloading
13. Companion objects
14. Data classes
15. Separate interfaces for read-only and mutable collections

Android’s Kotlin-first approach:

At Google I/O 2019, we announced that Android development will be increasingly Kotlin-first, and we’ve stood by that commitment. Kotlin is an expressive and concise programming language that reduces common code errors and easily integrates into existing apps. If you’re looking to build an Android app, we recommend starting with Kotlin to take advantage of its best-in-class features.

In an effort to support Android development using Kotlin, we co-founded the Kotlin Foundation and have ongoing investments in improving compiler performance and build speed. To learn more about Android's commitment to being Kotlin-first, see Android's commitment to Kotlin.

Why is Android development Kotlin-first?

We reviewed feedback that came directly from developers at conferences, our Customer Advisory Board (CAB), Google Developers Experts (GDE), and through our developer research. Many developers already enjoy using Kotlin, and the request for more Kotlin support was clear. Here’s what developers appreciate about writing in Kotlin:

- **Expressive and concise:** You can do more with less. Express your ideas and reduce the amount of boilerplate code. 67% of professional developers who use Kotlin say Kotlin has increased their productivity.
- **Safer code:** Kotlin has many language features to help you avoid common programming mistakes such as null pointer exceptions. Android apps that contain Kotlin code are 20% less likely to crash.
- **Interoperable:** Call Java-based code from Kotlin, or call Kotlin from Java-based code. Kotlin is 100% interoperable with the Java programming language, so you can have as little or as much of Kotlin in your project as you want.
- **Structured Concurrency:** Kotlin coroutines make asynchronous code as easy to work with as blocking code. Coroutines dramatically simplify background task management for everything from network calls to accessing local data.

What does Kotlin-first mean?

When building new Android development tools and content, such as Jetpack libraries, samples, documentation, and training content, we will design them with Kotlin users in mind while continuing to provide support for using our APIs from the Java programming language.

Some common Kotlin patterns with Android

This topic focuses on some of the most useful aspects of the Kotlin language when developing for Android.

Work with fragments

Inheritance

You can declare a class in Kotlin with the class keyword. In the following example, LoginFragment is a subclass of Fragment. You can indicate inheritance by using the : operator between the subclass and its parent.

In this class declaration, LoginFragment is responsible for calling the constructor of its superclass, Fragment.

Within LoginFragment, you can override a number of lifecycle callbacks to respond to state changes in your Fragment. To override a function, use the override keyword.

Nullability and initialization

some of the parameters in the overridden methods have types suffixed with a question mark ?. This indicates that the arguments passed for these parameters can be null. Be sure to handle their nullability safely.

In Kotlin, you must initialize an object's properties when declaring the object. This implies that when you obtain an instance of a class, you can immediately reference any of its accessible properties. The View objects in a Fragment, however, aren’t ready to be inflated until calling Fragment#onCreateView, so you need a way to defer property initialization for a View.

The lateinit lets you defer property initialization. When using lateinit, you should initialize your property as soon as possible.

Property delegation

When initializing properties, you might repeat some of Android's more common patterns, such as accessing a ViewModel within a Fragment. To avoid excess duplicate code, you can use Kotlin’s *property delegation* syntax.

Property delegation uses reflection, which adds some performance overhead. The tradeoff is a concise syntax that saves development time.

Nullability

Kotlin provides strict nullability rules that maintain type-safety throughout your app. In Kotlin, references to objects cannot contain null values by default. To assign a null value to a variable, you must declare a *nullable* variable type by adding ? to the end of the base type.

Interoperability

Kotlin's strict rules make your code safer and more concise. These rules lower the chances of having a NullPointerException that would cause your app to crash. Moreover, they reduce the number of null checks you need to make in your code.

Often, you must also call into non-Kotlin code when writing an Android app, as most Android APIs are written in the Java programming language.

Nullability is a key area where Java and Kotlin differ in behavior. Java is less strict with nullability syntax.

As an example, the Account class has a few properties, including a String property called name. Java does not have Kotlin’s rules around nullability, instead relying on optional *nullability annotations* to explicitly declare whether you can assign a null value.

Because the Android framework is written primarily in Java, you might run into this scenario when calling into APIs without nullability annotations.

Platform types

If you use Kotlin to reference an unannotated name member that is defined in a Java Account class, the compiler doesn't know whether the String maps to a String or a String? in Kotlin. This ambiguity is represented via a *platform type*, String!.

String! has no special meaning to the Kotlin compiler. String! can represent either a String or a String?, and the compiler lets you assign a value of either type. Note that you risk throwing a NullPointerException if you represent the type as a String and assign a null value.

To address this issue, you should use nullability annotations whenever you write code in Java. These annotations help both Java and Kotlin developers.

Handling nullability

If you are unsure about a Java type, you should consider it to be nullable. As an example, the name member of the Account class is not annotated, so you should assume it to be a nullable String.

If you want to trim name so that its value does not include leading or trailing whitespace, you can use Kotlin's trim function. You can safely trim a String? in a few different ways. One of these ways is to use the *not-null assertion operator*, !!

The !! operator treats everything on its left-hand side as non-null, so in this case, you are treating name as a non-null String. If the result of the expression to its left is null, then your app throws a NullPointerException. This operator is quick and easy, but it should be used sparingly, as it can reintroduce instances of NullPointerException into your code.

Using the safe-call operator, if name is non-null, then the result of name?.trim() is a name value without leading or trailing whitespace. If name is null, then the result of name?.trim() is null. This means that your app can never throw a NullPointerException when executing this statement.

While the safe-call operator saves you from a potential NullPointerException, it does pass a null value to the next statement. You can instead handle null cases immediately by using an *Elvis operator* (?:)

If the result of the expression on the left-hand side of the Elvis operator is null, then the value on the right-hand side is assigned to accountName. This technique is useful for providing a default value that would otherwise be null.

Android API changes

Android APIs are becoming increasingly Kotlin-friendly. Many of Android's most-common APIs, including AppCompatActivity and Fragment, contain nullability annotations, and certain calls like Fragment#getContext have more Kotlin-friendly alternatives.

For example, accessing the Context of a Fragment is almost always non-null, since most of the calls that you make in a Fragment occur while the Fragment is attached to an Activity (a subclass of Context). That said, Fragment#getContext does not always return a non-null value, as there are scenarios where a Fragment is not attached to an Activity. Thus, the return type of Fragment#getContext is nullable.

Since the Context returned from Fragment#getContext is nullable (and is annotated as @Nullable), you must treat it as a Context? in your Kotlin code. This means applying one of the previously-mentioned operators to address nullability before accessing its properties and functions. For some of these scenarios, Android contains alternative APIs that provide this convenience. Fragment#requireContext, for example, returns a non-null Context and throws an IllegalStateException if called when a Context would be null. This way, you can treat the resulting Context as non-null without the need for safe-call operators or workarounds.

Property initialization

Properties in Kotlin are not initialized by default. They must be initialized when their enclosing class is initialized.

You can initialize properties in a few different ways. The following example shows how to initialize an index variable by assigning a value to it in the class declaration

However, you might have some properties that can't be initialized during object construction. For example, you might want to reference a View from within a Fragment, which means that the layout must be inflated first. Inflation does not occur when a Fragment is constructed. Instead, it's inflated when calling Fragment#onCreateView.

Android Runtime:

This is the third section of the architecture and available on the second layer from the bottom. This section provides a key component called Dalvik Virtual Machine which is a kind of Java Virtual Machine specially designed and optimized for Android. The Dalvik VM makes use of Linux core features like memory management and multi-threading, which is intrinsic in the Java language. The Dalvik VM enables every Android application to run in its own process, with its own instance of the Dalvik virtual machine. The Android runtime also provides a set of core libraries which enable Android application developers to write Android applications using standard Java programming language.

Application Framework:

The Application Framework layer provides many higher-level services to applications in the form of Java classes. Application developers are allowed to make use of these services in their applications.

The Android framework includes the following key services −

•**Activity Manager** − Controls all aspects of the application lifecycle and activity stack.

•**Content Providers** − Allows applications to publish and share data with other applications.

•**Resource Manager** − Provides access to non-code embedded resources such as strings, colour settings and user interface layouts.

•**Notifications Manager** − Allows applications to display alerts and notifications to the user.

•**View System** − An extensible set of views used to create application user interfaces.

Android Activity: An activity represents a single screen with a user interface just like window or frame of Java. Android activity is the subclass of ContextThemeWrapper class. If you have worked with C, C++ or Java programming language then you must have seen that your program starts from main() function. Very similar way, Android system initiates its program with in an Activity starting with a call on onCreate() callback method. There is a sequence of callback methods that start up an activity and a sequence of callback methods that tear down an activity as shown in the below Activity life cycle diagram:

• **onCreate():** This is the first callback and called when the activity is first created.

• **onStart()**: This callback is called when the activity becomes visible to the user.

• **onResume()**: This is called when the user starts interacting with the application.

• **onPause()**: The paused activity does not receive user input and cannot execute any code and is called when the current activity is being paused and the previous activity is being resumed.

• **onStop():** This callback is called when the activity is no longer visible.

• **onDestroy():** This callback is called before the activity is destroyed by the system.

• **onRestart():** This callback is called when the activity restarts after stopping it.

Some Details about some Important Project Components:

Android Manifest.xml :-

A manifest file in computing is a file containing metadata for a group of accompanying files that are part of a set or coherent unit. For example, the files of a computer program may have a manifest describing the name, version number, license and the constituent files of the program**.**

Tab Adapter:-

Tabs are created using the newTab() method of TabLayout class. The title and icon of Tabs are set through setText(int) and setIcon(int) methods of TabListener interface respectively. Tabs of layout are attached over TabLayout using the method addTab(Tab) method**.**

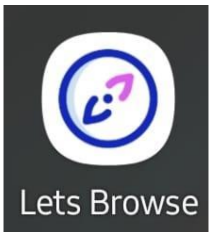
BookMark Adapter:-

The Android browser, a standard Google program, functions much like your Web browser on a computer. You have the ability to bookmark Web pages for quick access. Once you bookmark a page, you can bring it up with one touch of your finger.

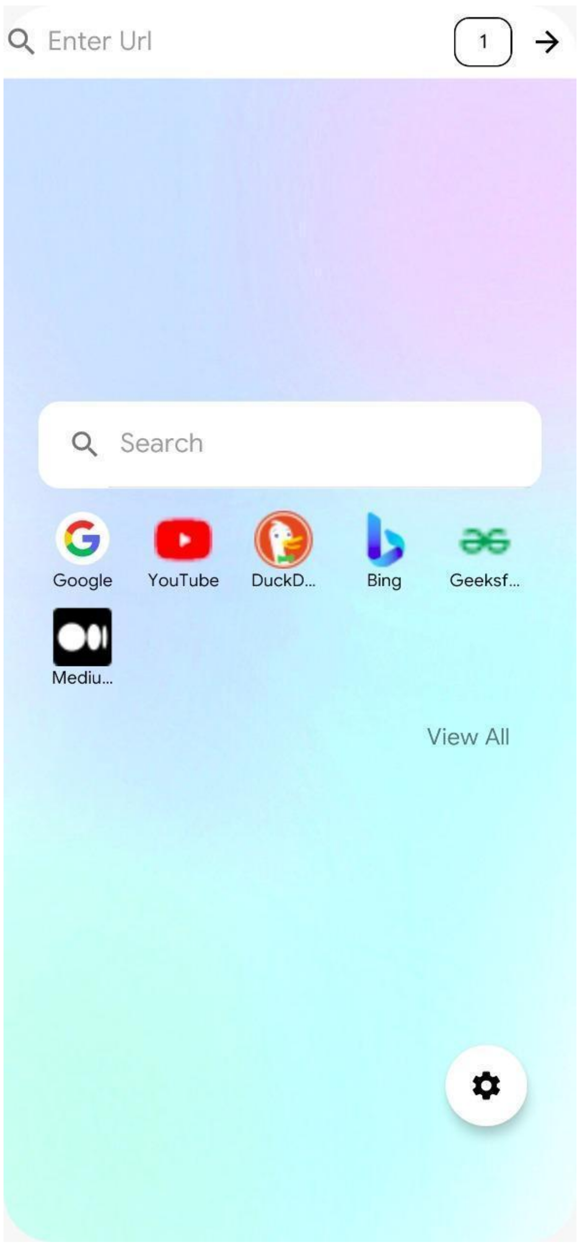
Gradle Script:-

Module-level build.gradle: Located in the project/module directory of the project this Gradle script is where all the dependencies are defined and where the SDK versions are declared.

6 Output:

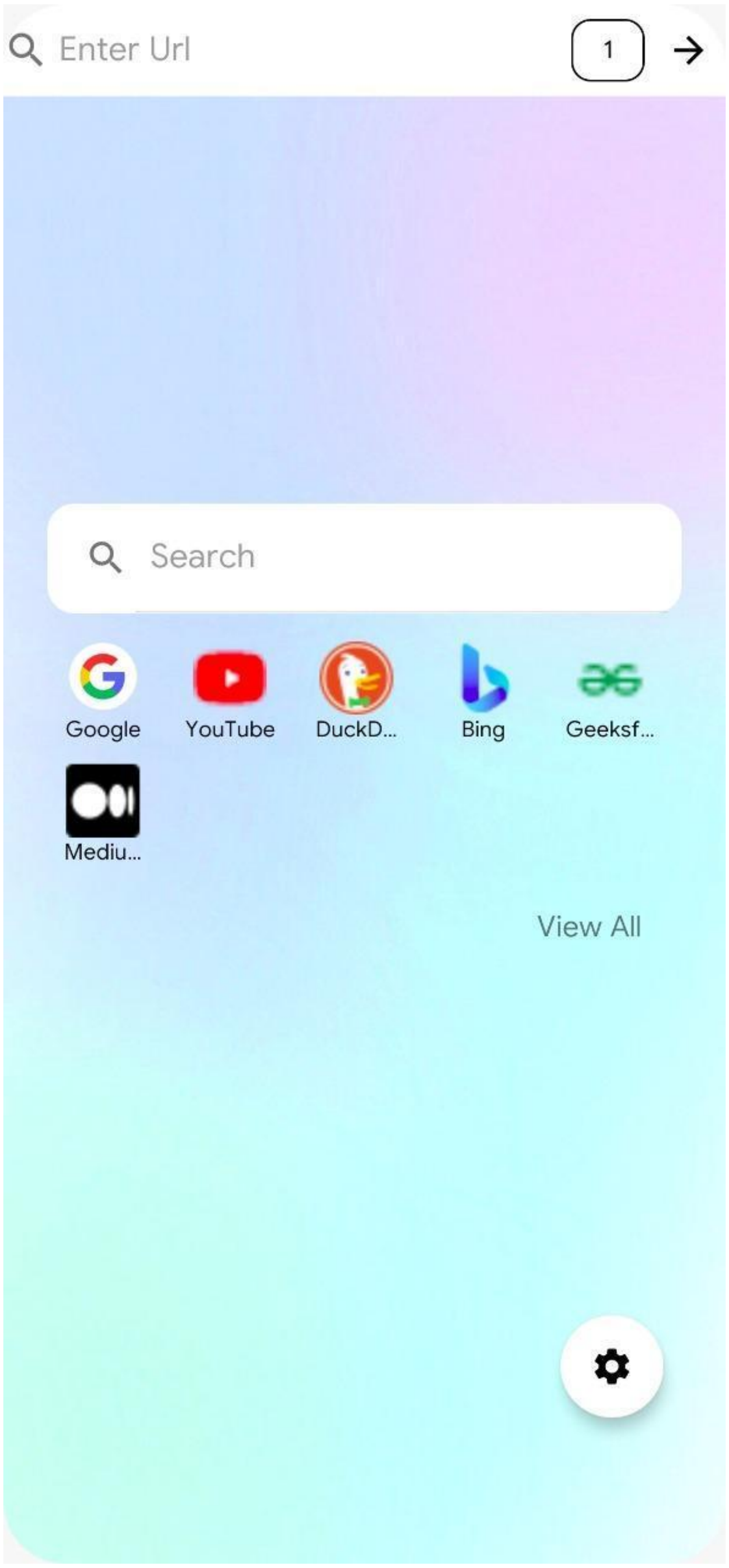


This is the logo of our project



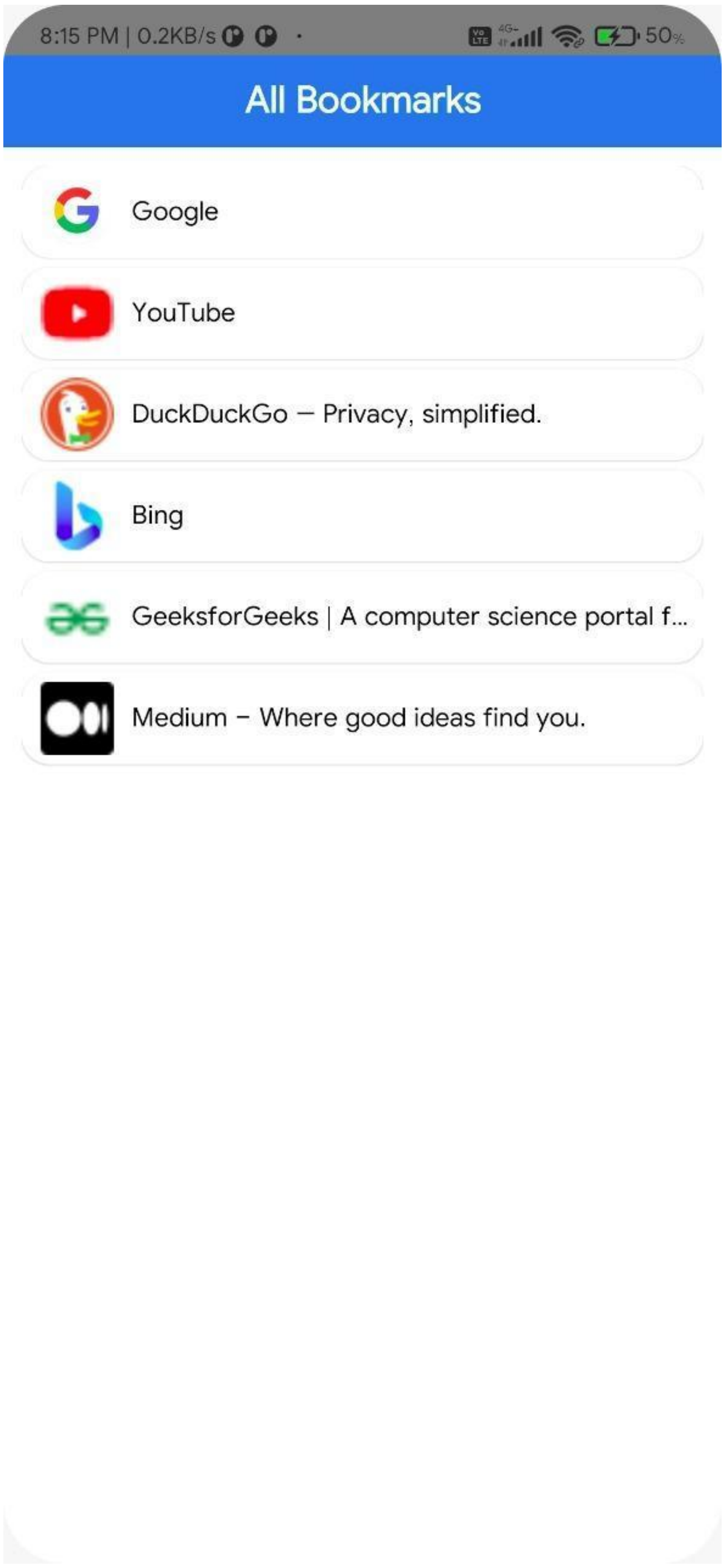
Bookmarks in web browser feature:

used to save a web site's URL address for future reference



Bookmarks in web browser feature:

used to save a web site's URL address for future reference.

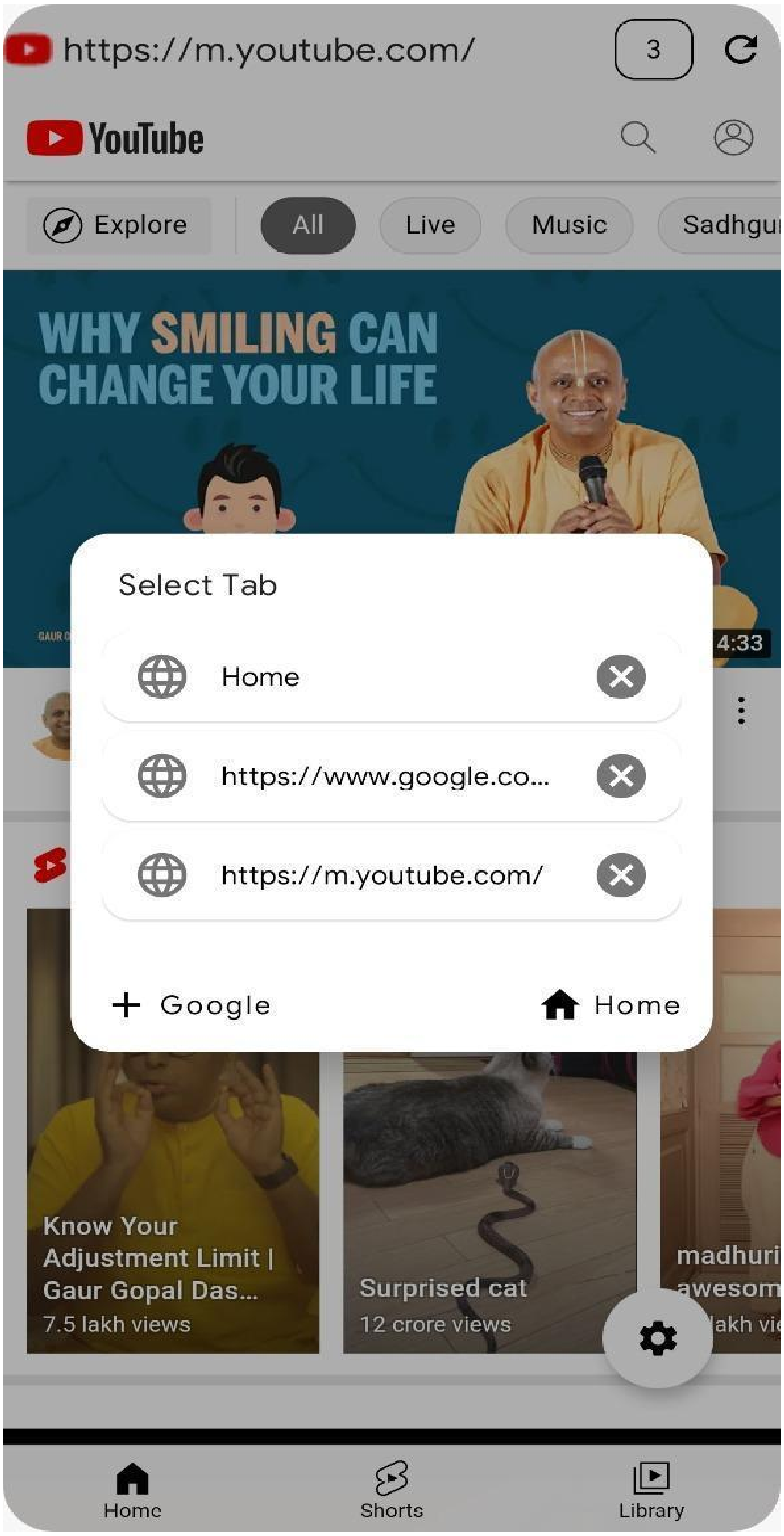

8:15 PM | 0.2KB/s

50%

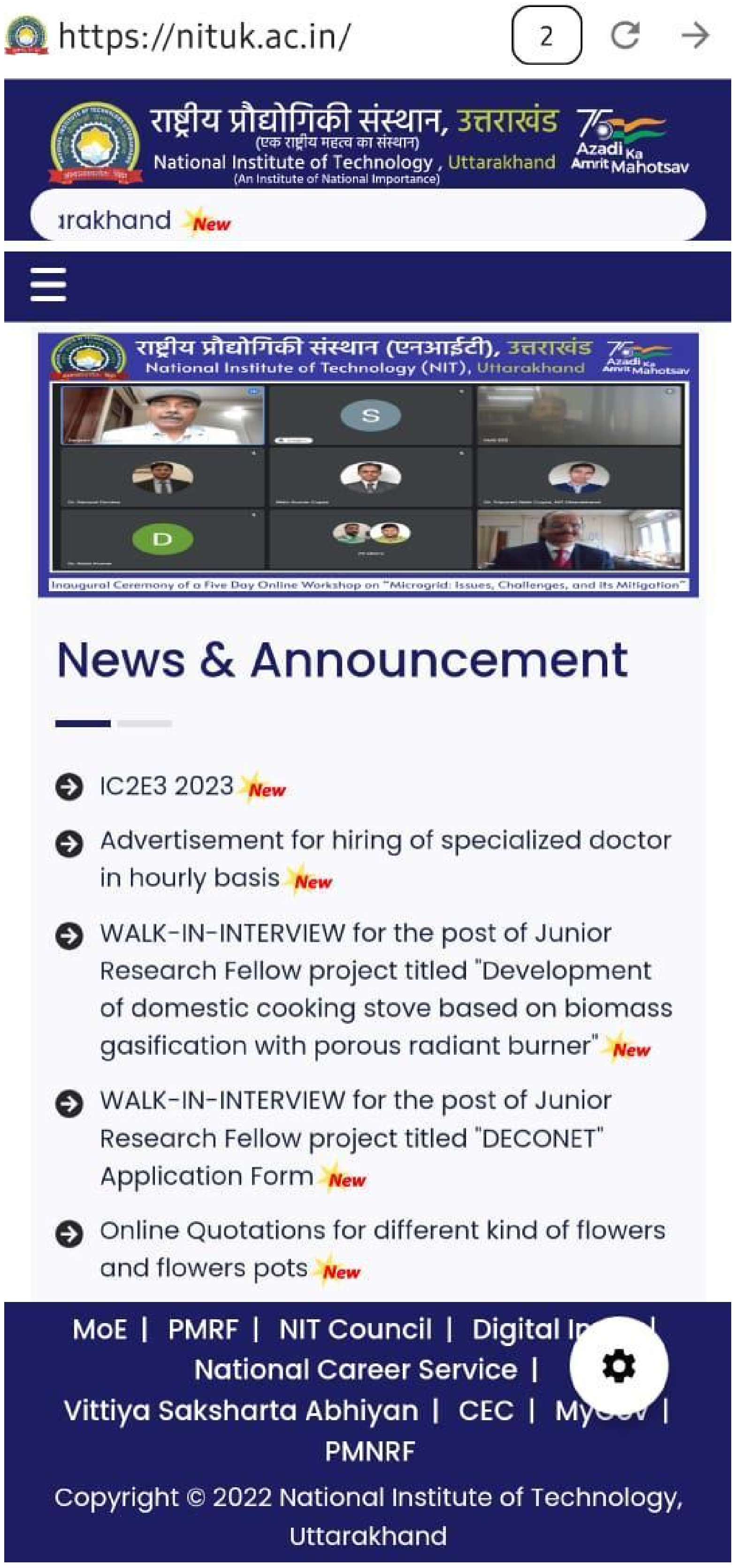
All Bookmarks

- Google
- YouTube
- DuckDuckGo — Privacy, simplified.
- Bing
- GeeksforGeeks | A computer science portal f...
- Medium – Where good ideas find you.

This Browser supports multiple tabs to be accessed in a Single Session. Users can switch between different tabs of his choice at different points of time .



Multi-tab support


https://nituk.ac.in/

राष्ट्रीय प्रौद्योगिकी संस्थान, उत्तराखंड
(एक राष्ट्रीय महत्व का संस्थान)
National Institute of Technology , Uttarakhand
(An Institute of National Importance)

ırakhand New

राष्ट्रीय प्रौद्योगिकी संस्थान (एनआईटी), उत्तराखंड
National Institute of Technology (NIT), Uttarakhand

Inaugural Ceremony of a Five Day Online Workshop on "Microgrid: Issues, Challenges, and Its Mitigation"

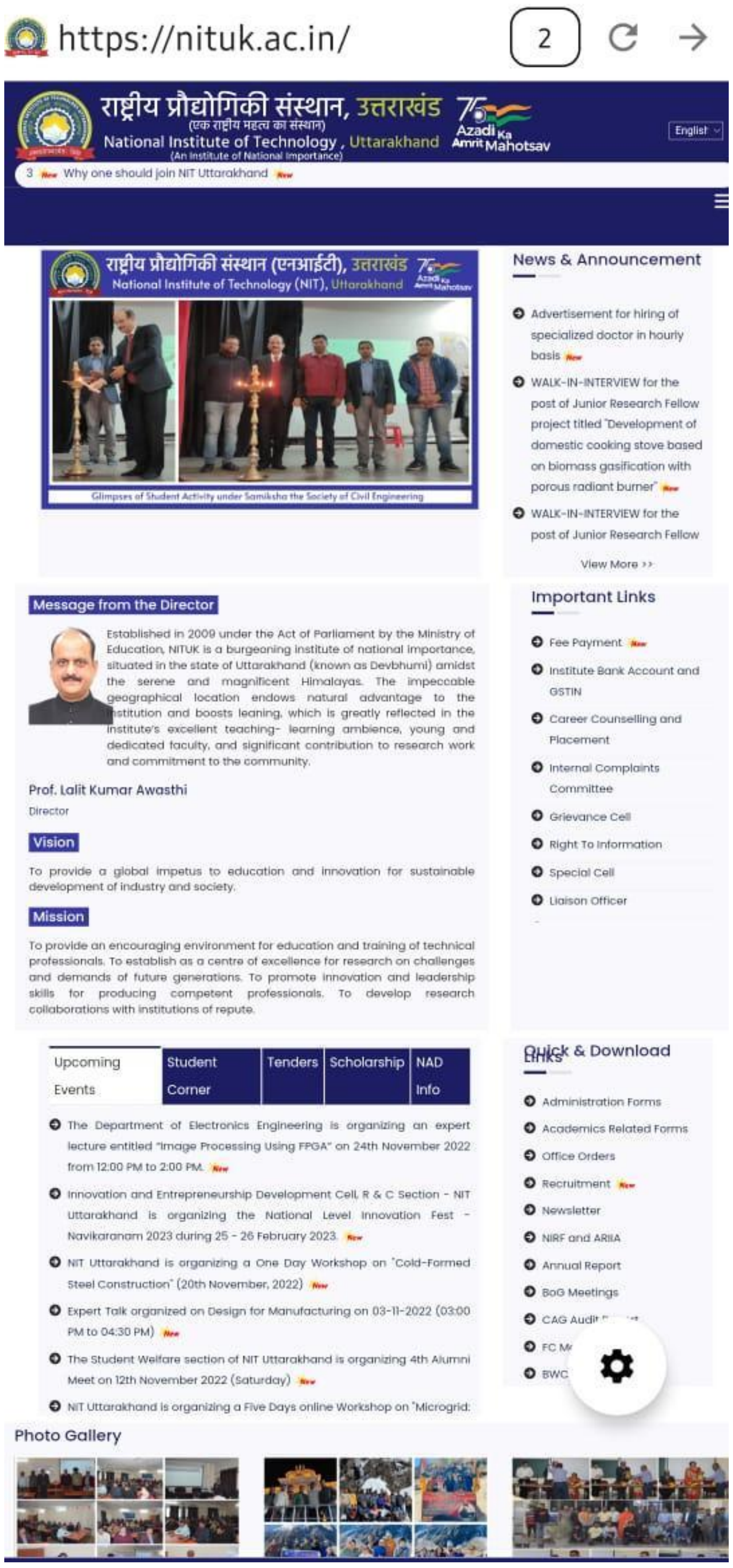
News & Announcement

- IC2E3 2023 New
- Advertisement for hiring of specialized doctor in hourly basis New
- WALK-IN-INTERVIEW for the post of Junior Research Fellow project titled "Development of domestic cooking stove based on biomass gasification with porous radiant burner" New
- WALK-IN-INTERVIEW for the post of Junior Research Fellow project titled "DECONET" Application Form New
- Online Quotations for different kind of flowers and flowers pots New

MoE | PMRF | NIT Council | Digital In | National Career Service | Vittiya Saksharta Abhiyan | CEC | MyGov | PMNRF

Copyright © 2022 National Institute of Technology, Uttarakhand

This is preview of Full screen Enabled View


https://nituk.ac.in/ 2

राष्ट्रीय प्रौद्योगिकी संस्थान, उत्तराखंड
(एक राष्ट्रीय महत्व का संस्थान)
National Institute of Technology, Uttarakhand
(An Institute of National Importance)

3 New Why one should join NIT Uttarakhand New

राष्ट्रीय प्रौद्योगिकी संस्थान (एनआईटी), उत्तराखंड
National Institute of Technology (NIT), Uttarakhand

Glimpses of Student Activity under Samiksha the Society of Civil Engineering

News & Announcement

- Advertisement for hiring of specialized doctor in hourly basis New
- WALK-IN-INTERVIEW for the post of Junior Research Fellow project titled "Development of domestic cooking stove based on biomass gasification with porous radiant burner" New
- WALK-IN-INTERVIEW for the post of Junior Research Fellow

View More ››

Message from the Director

Established in 2009 under the Act of Parliament by the Ministry of Education, NITUK is a burgeoning institute of national importance, situated in the state of Uttarakhand (known as Devbhumi) amidst the serene and magnificent Himalayas. The impeccable geographical location endows natural advantage to the institution and boosts leaning, which is greatly reflected in the institute's excellent teaching- learning ambience, young and dedicated faculty, and significant contribution to research work and commitment to the community.

Prof. Lalit Kumar Awasthi
Director

Vision

To provide a global impetus to education and innovation for sustainable development of industry and society.

Mission

To provide an encouraging environment for education and training of technical professionals. To establish as a centre of excellence for research on challenges and demands of future generations. To promote innovation and leadership skills for producing competent professionals. To develop research collaborations with institutions of repute.

Important Links

- Fee Payment New
- Institute Bank Account and GSTIN
- Career Counselling and Placement
- Internal Complaints Committee
- Grievance Cell
- Right To Information
- Special Cell
- Liaison Officer

Upcoming Events	Student Corner	Tenders	Scholarship	NAD Info

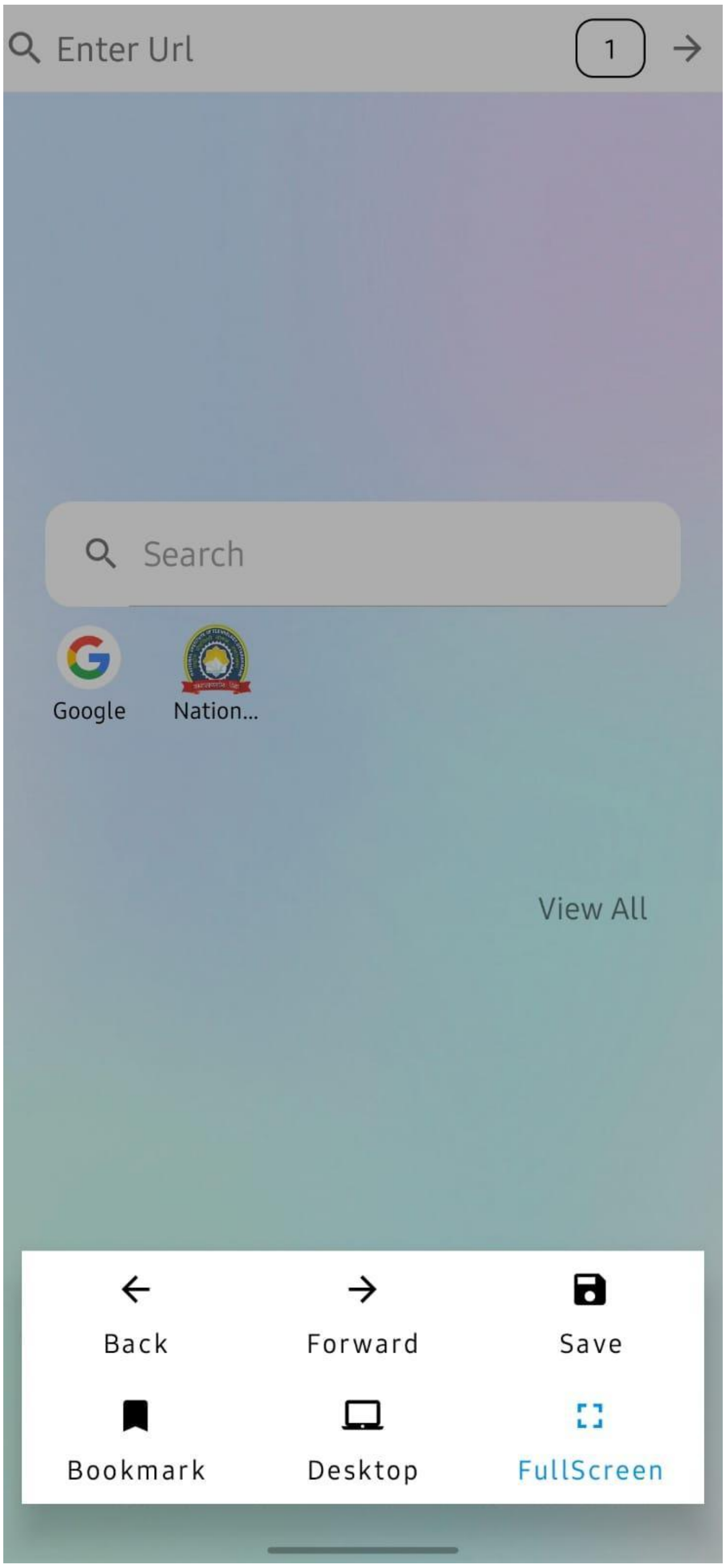
- The Department of Electronics Engineering is organizing an expert lecture entitled "Image Processing Using FPGA" on 24th November 2022 from 12:00 PM to 2:00 PM. New
- Innovation and Entrepreneurship Development Cell, R & C Section - NIT Uttarakhand is organizing the National Level Innovation Fest - Navikaranam 2023 during 25 - 26 February 2023. New
- NIT Uttarakhand is organizing a One Day Workshop on "Cold-Formed Steel Construction" (20th November, 2022) New
- Expert Talk organized on Design for Manufacturing on 03-11-2022 (03:00 PM to 04:30 PM) New
- The Student Welfare section of NIT Uttarakhand is organizing 4th Alumni Meet on 12th November 2022 (Saturday) New
- NIT Uttarakhand is organizing a Five Days online Workshop on "Microgrid:

Quick & Download Links

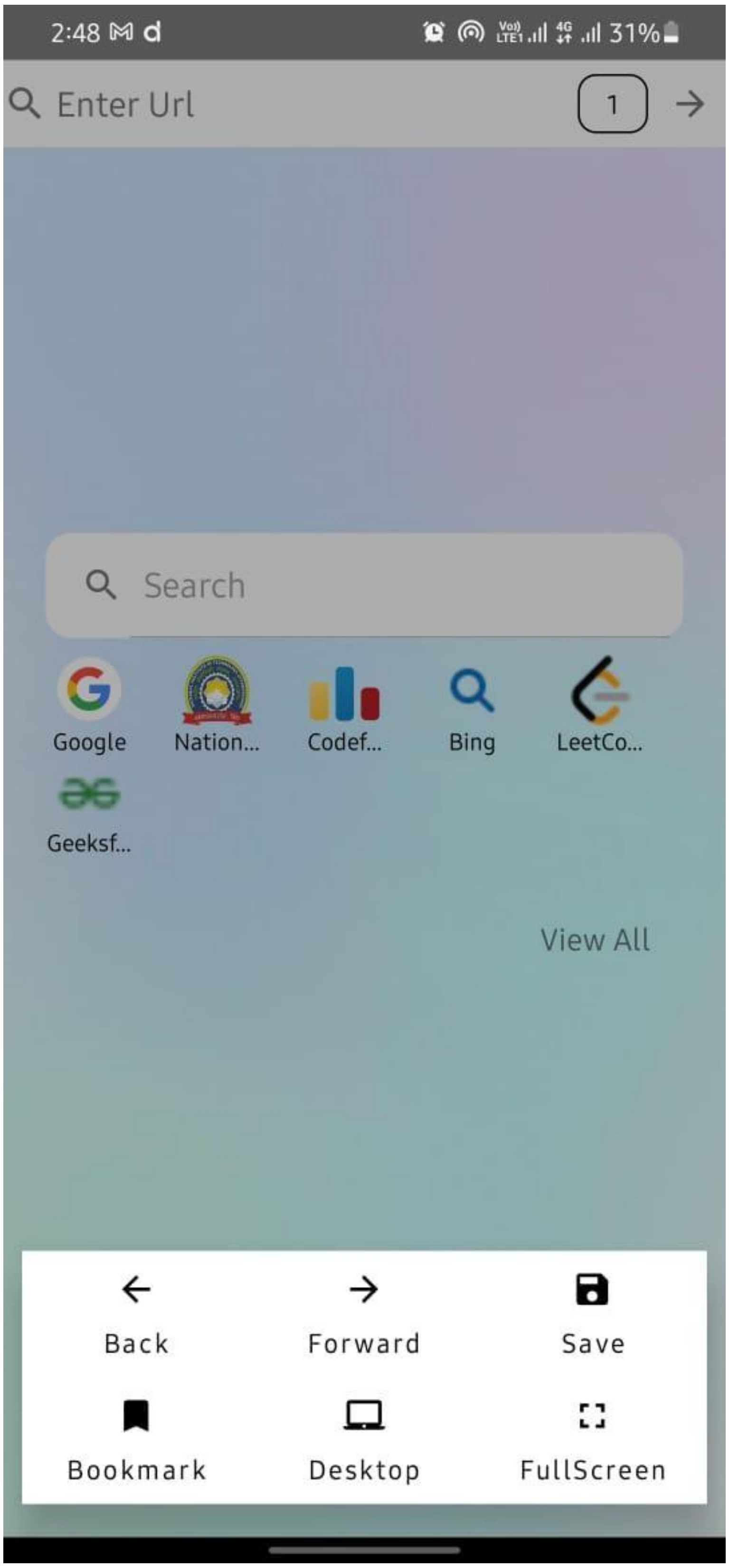
- Administration Forms
- Academics Related Forms
- Office Orders
- Recruitment New
- Newsletter
- NIRF and ARIIA
- Annual Report
- BoG Meetings
- CAG Audit
- FC M
- BWC

Photo Gallery

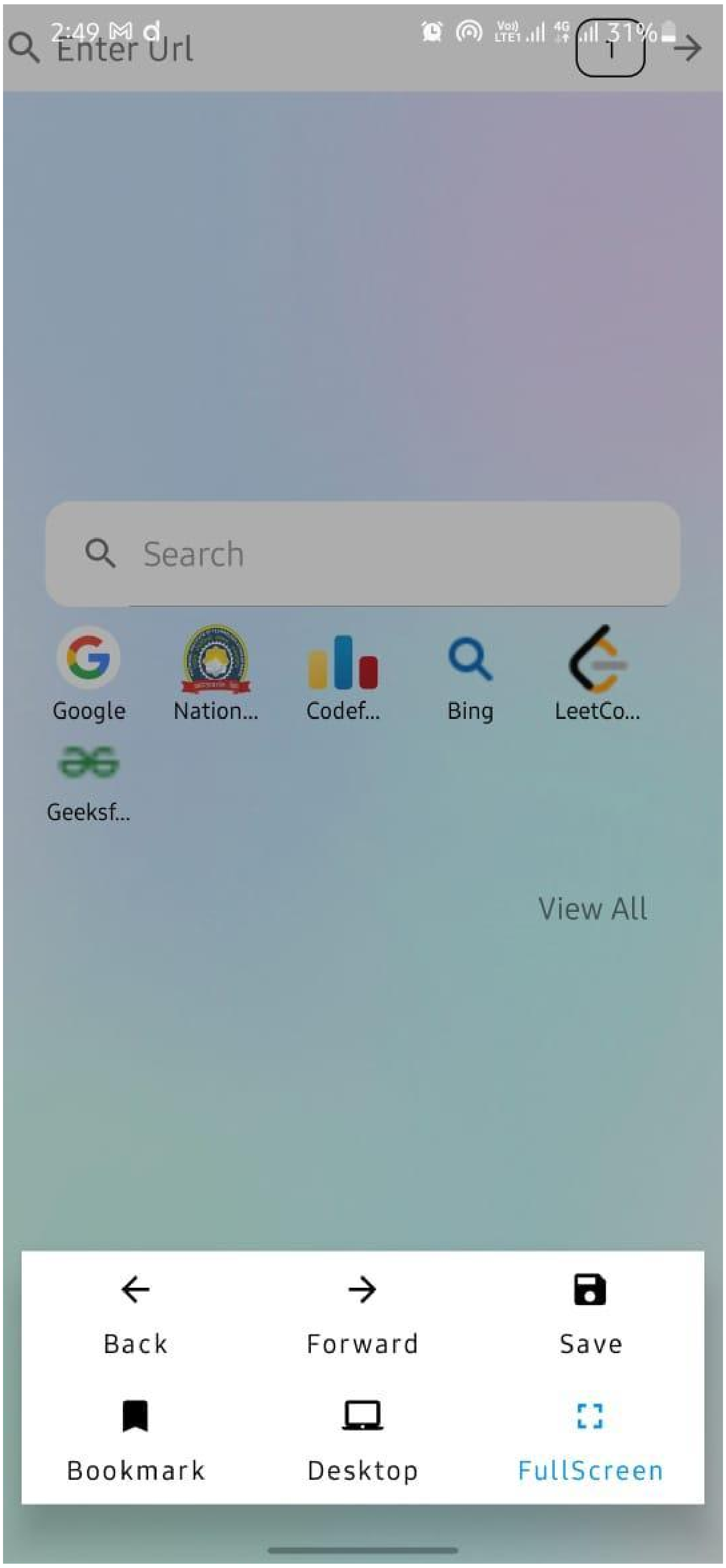
Our NIT Uttarakhand website in Desktop Mode



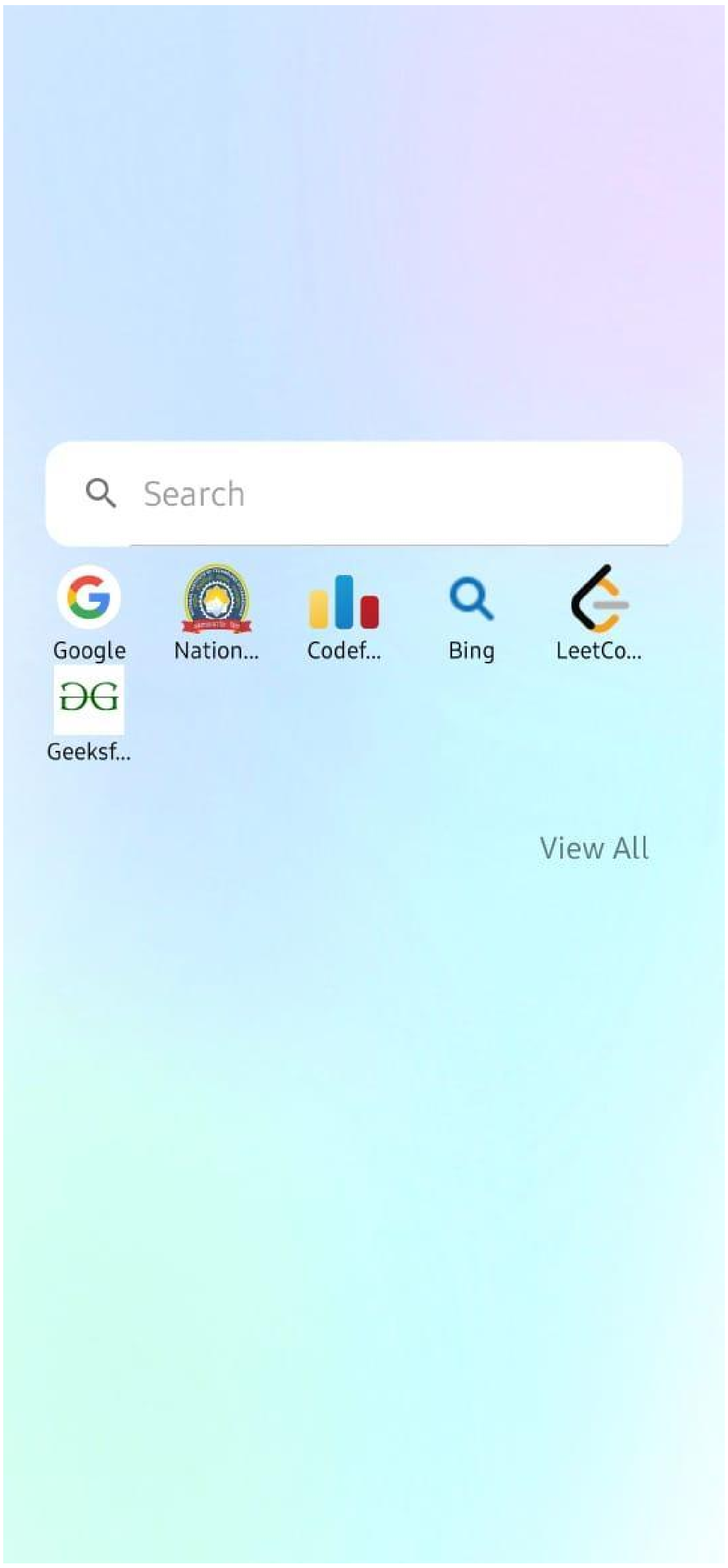
Controls present in Floating Action Button(FAB)



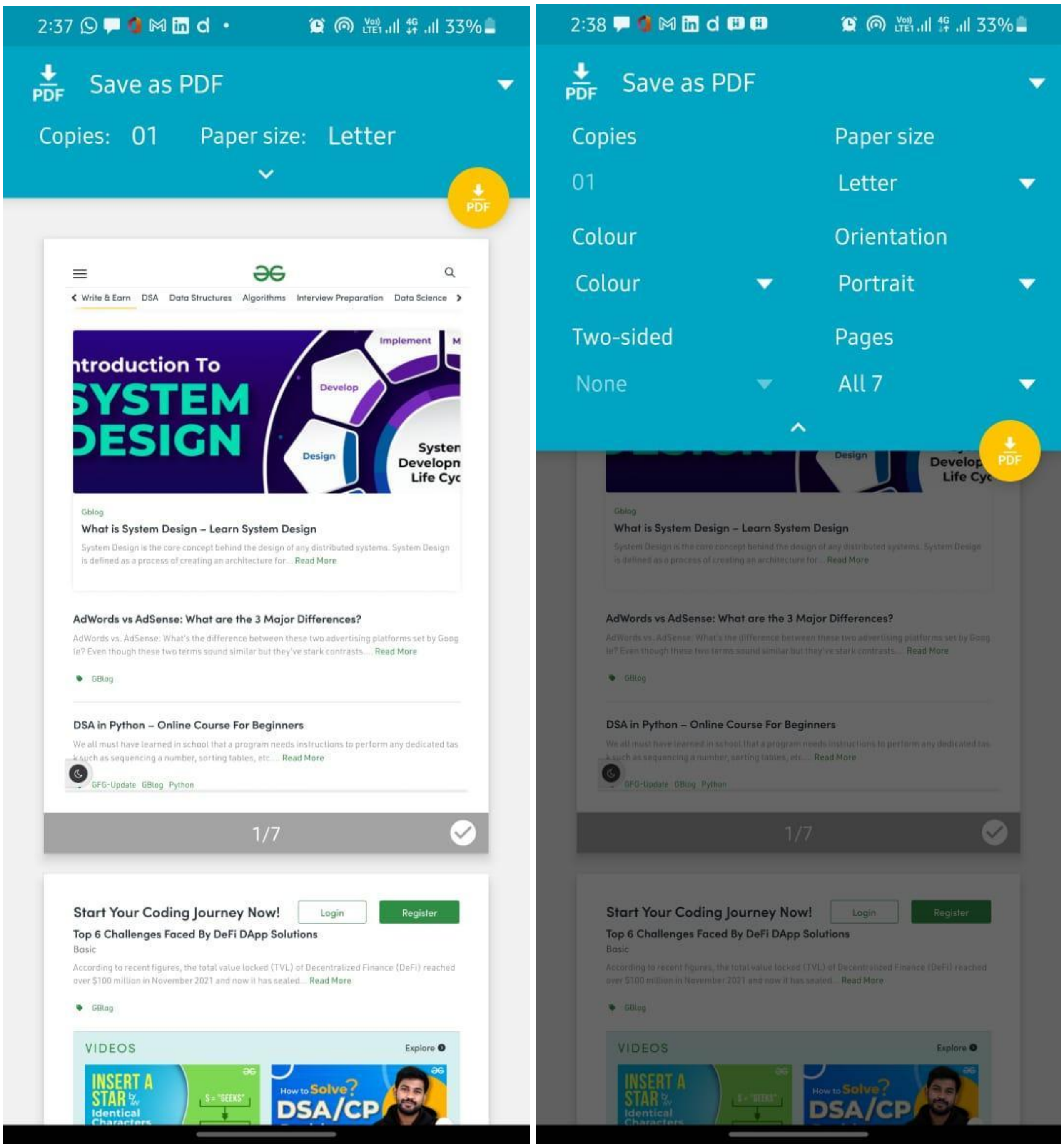
Full Screen Turned Off



Full Screen Turned On



Hidden Address Bar



Copies: 01 Paper size: Letter

Copies 01

Paper size Letter

Colour Colour

Orientation Portrait

Two-sided None

Pages All 7

Gblog

What is System Design – Learn System Design

System Design is the core concept behind the design of any distributed systems. System Design is defined as a process of creating an architecture for... Read More

AdWords vs AdSense: What are the 3 Major Differences?

AdWords vs. AdSense: What's the difference between these two advertising platforms set by Google? Even though these two terms sound similar but they've stark contrasts.... Read More

GBlog

DSA in Python – Online Course For Beginners

We all must have learned in school that a program needs instructions to perform any dedicated task such as sequencing a number, sorting tables, etc.... Read More

GFG-Update GBlog Python

Start Your Coding Journey Now!

Top 6 Challenges Faced By DeFi DApp Solutions

Basic

According to recent figures, the total value locked (TVL) of Decentralized Finance (DeFi) reached over $100 million in November 2021 and now it has sealed... Read More

GBlog

Save as PDF Functionality and Expanded Options in Save as PDF

8 References:

[1]https://developer.android.com/
[2]https://developer.android.com/kotlin?gclid=Cj0KCQiA1ZGcBhCoARIsAGQ0kkrt_zyVsulTJvOoD7V5VdmlGHDzxj5iaUqmjligIM5UuA0suSrTCRcaAq6TEALw_wcB&gclsrc=aw.ds
[3]https://developer.android.com/studio/write/java8-support?gclid=Cj0KCQiA1ZGcBhCoARIsAGQ0kkrdR7lwZWw_POgCKz2us5LrCBxErVr0Hz37FLdLrzlcFgAL-EDuypcaApkUEALw_wcB&gclsrc=aw.ds
[4]https://developer.android.com/studio?gclid=Cj0KCQiA1ZGcBhCoARIsAGQ0kkqfwUNJLNDpEDUYnCY2AYKJJPpSxPV2qLvDjKZQJzXTE6YEV9YkxL8aAt_6EALw_wcB&gclsrc=aw.ds
[5]https://www.geeksforgeeks.org/android-studio-tutorial/
[6]https://www.geeksforgeeks.org/a-complete-guide-to-learn-xml-for-android-app-development/
[7]https://developer.android.com/develop/ui
[8]https://github.com/KnoxArrow/LetsBrowse
[9]https://kotlinlang.org/docs/comparison-to-java.html#some-java-issues-addressed-in-kotlin
[10]https://www.geeksforgeeks.org/android-ui-layouts/#:~:text=Android%20Layout%20is%20used%20to,combination%20of%20View%20and%20ViewGroup.