A

Project Report

on

Android App for Blocking other apps

Submitted to

BHILAI INSTITUTE OF TECHNOLOGY, DURG

an autonomous institute

affiliated to

CHHATTISGARH SWAMI VIVEKANAND TECHNICAL UNIVERSITY

BHILAI

in partial fulfillment

of

Bachelor of Technology

in

Computer Science and Engineering

by

Sanjana Gupta (6th sem) – 300102221305
Saniya khan (6th sem) – 300102221306
Sweety Pandre (6th sem) – 300102220098
Rishab Biswal (6th sem) - 300102221301

Under the Guidance of

Prof. Shiv Dutta Mishra
Assistant Professor

Prof. N. Chiranjeeva Rao
Assistant Professor

Department of Computer Science and Engineering
Bhilai Institute of Technology,
(An Autonomous Institute)
Bhilai House, GE Road, Durg, Chhattisgarh 491001

Session: 2022 – 2023

DECLARATION BY THE CANDIDATE(s)

I/we the undersigned solemnly declare that the report of the project work entitled **Android App for Blocking other apps**, is based on my own work carried out during the course of my study under the supervision of **Prof. N. Chiranjeeva Rao**

I assert that the statements made and conclusions drawn are an outcome of the project work. I further declare that to the best of my knowledge and belief that the report does not contain any part of any work which has been submitted for the award of any other degree/diploma/certificate in this University/ any other University of India or any other country.

Sanjana Gupta 300102221305 BG4715	Saniya Khan 300102221306 CA0686	Sweety Pandre 300102220098 BK4277	Rishab Biswal 300102221301 CA0687

CERTIFICATE

This is to Certify that the report of the project submitted is an outcome of the project work entitled **Android App for Blocking other apps** carried out by

Sanjana Gupta bearing Roll No 300102221305, Enrollment No BG4715;
Saniya Khan bearing Roll No 300102221306, Enrollment No CA0686;
Sweety Pandre bearing Roll No 300102220098, Enrollment No BK4277**;**
Rishab Biswal bearing Roll No 300102221301, Enrollment No CA0687;

Under my guidance and supervision in partial fulfillment of Bachelor of Technology in Computer Science from Bhilai Institute of Technology, Durg, an autonomous institute affiliated to Chhattisgarh Swami Vivekanand Technical University, Bhilai (C.G).

To the best of my knowledge and belief the project

i) Embodies the work of the candidate himself / herself,

ii) Has duly been completed,

iii) Fulfills the requirement of the Ordinance relating to the B. Tech. degree of the University,

iv) Is up to the desired standard for the purpose of which is submitted.

Signature of Coordinator	**Signature of Coordinator**	**Signature of Guide**
Prof. Dasari Siva Sankar	**Prof. Shiv Dutta Mishra**	**Prof. N. Chiranjeeva Rao**
Assistant Professor	**Assistant Professor**	**Assistant Professor**
Computer Sc. & Engg	**Computer Sc. & Engg.**	**Computer Sc. & Engg.**

The Project work as mentioned above is hereby being recommended and forwarded for examination and evaluation.

Dr. (Mrs,) Sunita Soni
Head of the Department
Computer Sc. & Engg.

CERTIFICATE BY THE EXAMINERS

This is to Certify that the project the entitled

"Android App for Blocking other apps",

Submitted by

Sanjana Gupta bearing Roll No 300102221305, Enrollment No BG4715;
Saniya Khan bearing Roll No 300102221306, Enrollment No CA0686;
Sweety Pandre bearing Roll No 300102220098, Enrollment No BK4277**;**
Rishab Biswal bearing Roll No 300102221301, Enrollment No CA0687;

Have been examined by the undersigned as a part of the examination for the award of Bachelor of Technology degree in Computer Science and Engineering from Bhilai Institute of Technology, Durg, an autonomous institute affiliated to Chhattisgarh Swami Vivekanand Technical University, Bhilai (C.G)

(Internal Examiner)	**(External Examiner)**
Name: Prof. Dasari Siva Sankar	**Name:**
Date: 02/06/2023	**Date:**

ACKNOWLEDGEMENT

I have great pleasure in the submission of this project report entitled **Android App for Blocking other apps** in partial fulfillment the degree of Bachelor of Engineering (CSE). While submitting this Project report, I take this opportunity to thank those directly or indirectly related to project work.

I would like to thank my guide **Prof. N. Chiranjeeva Rao** who has provided the opportunity and organizing project for me. Without his active co-operation and guidance, it would have become very difficult to complete task in time.

I would like to express sincere thanks and gratitude to Dr. Mohan Kumar Gupta, **Principal of the Institution**, Dr. (Mrs.) Sunita Soni, **Head of the Department** Computer Science & Engineering for their encouragement and cordial support.

While Submission of the project, I also like to thanks to Prof Shiv Dutta Mishra, and Prof. Dasari Siva Sankar **Project Coordinator,** faculties and all the staff of department of Computer Science & Engineering, **Bhilai Institute of Technology, Durg** for their continuous help and guidance throughout the course of project.

Acknowledgement is due to our parents, family members, friends and all those persons who have helped us directly or indirectly in the successful completion of the project work.

Sanjana Gupta
300102221305
BG4715

Saniya Khan
300102221306
CA0686

Sweety Pandre
300102220098
BK4277

Rishab Biswal
300102221301
CA0687

CONTENT

ABSTRACT (1-2)

1. INTODUCTION

1.1 OBJECTIVE (2-3)

1.2 PROJECT DESCRIPTION (3-5)

2. SYSTEM STUDY

2.1 EXISTING AND PROPOSED SYSTEM (5-6)

2.2 FEASIBILITY STUDY (7)

2.3 TOOLS AND TECHNOLOGIES USED (8-9)

2.4 HARDWARE AND SOFTWARE REQUIREMENTS (9-10)

3. SOFTWARE REQUIREMENTS SPECIFICATION

3.1 USERS (10-11)

3.2 FUNCTIONAL REQUIREMENTS (12-13)

3.3 NON-FUNCTIONAL REQUIREMENTS (13-14)

4. SYSTEM ANALYSIS AND DESIGN (various design diagrams according to project)

4.1 SYSTEM PERSPECTIVE (14-15)

4.2 DATABASE DESIGN (ER and/or Conceptual schema) (15-17)

4.3 CONTEXT DIAGRAM (DFD) (18-19)

4.4 USE CASE DIAGRAM (19-20)

4.5 SEQUENCE DIAGRAMS (21-22)

4.6 COLLABARATION DIAGRAMS (22)

4.7 ACTIVITY DIAGRAM (22)

5. IMPLEMENTATION (full code or code snippet may be included) **(**23-38)

5.1 SCREEN SHOTS (38-41)

6. SOFTWARE TESTING (Test cases etc.) (42-45)

7. CONCLUSION (45)

8. FUTURE ENHANCEMENTS (46-47)

BIBLIOGRAPHY (47-48)

List of figures

Figure No.	Name of Figure	Page No.
1	ER Diagram	17
2	DFD	18
3	Use Case Diagram	20
4	Sequence Diagram	22
5	Utils	24
6	Layout	33
7	Logo	38
8	Front	38
9	Add Apps	39
10	Select App	39
11	If Open App	40
12	Selected App	40
13	Set Time & Date	41
14	About	41

ABTRACT

App Block is an advanced Android application designed to help users regain control over their digital habits by effectively blocking access to distracting or unproductive applications. With the proliferation of smartphones and the increasing number of apps vying for our attention, it becomes essential to have a reliable tool that promotes focus, productivity, and overall digital well-being. This application offers a comprehensive set of features and a user-friendly interface to enable users to block or limit access to specific apps for designated periods. By setting boundaries and managing app usage, users can reduce distractions, improve time management, and enhance their productivity and concentration.

Key functionalities of App block include:

1. App Blocking: Users can select specific apps they want to block or limit access to during designated times, such as working hours, study sessions, or bedtime. This allows users to create a distraction-free environment and maintain focus on their tasks or personal well-being.

2. Scheduling and Timers: App Block provides flexible scheduling options and timers, allowing users to define when and for how long certain apps should be blocked. This feature encourages disciplined app usage and ensures that users adhere to their intended goals or routines.

3. App Usage Insights: The application provides detailed insights and reports on app usage patterns, allowing users to gain a better understanding of their digital habits. This information empowers users to make informed decisions about their app usage and take steps towards a healthier and more balanced smartphone experience.

4. Exception Management: App Block offers the flexibility to create exceptions for specific apps or categories, enabling users to maintain access to essential applications or those required for specific tasks. This ensures that users can strike a balance between necessary app usage and minimizing distractions.

5. Focus Mode: App Block includes a dedicated Focus Mode that allows users to temporarily block all apps except for essential ones, encouraging deep work or uninterrupted relaxation periods.

App Block is a powerful and customizable app blocker that promotes mindful app usage, productivity, and digital well-being. By empowering users to take control over their app habits and minimize distractions, it helps create a healthier relationship with technology and fosters a more balanced and fulfilling lifestyle

1.INTRODUCTION

1.1 Objective

The objective of an app blocker can be explored in greater depth across, allowing for a more comprehensive understanding of its significance and benefits. Here are some key objectives that can be:

1. Enhancing Productivity and Focus: One of the primary objectives of an app blocker is to help users enhance their productivity and maintain focus. By blocking or limiting access to distracting applications, users can concentrate on important tasks, work efficiently, and accomplish more within a given timeframe.

2. Managing Digital Well-being: In the modern digital age, managing digital well-being has become crucial. An app blocker aims to promote a healthier relationship with technology by allowing users to set boundaries and reduce excessive screen time. By managing app usage, individuals can strike a balance between their online and offline lives, leading to improved overall well-being.

3. Reducing Distractions and Increasing Efficiency: With the plethora of apps available, it is easy to get side tracked and lose valuable time. App blockers provide the means to minimize distractions and optimize efficiency by preventing access to non-essential or time-consuming applications. This objective focuses on empowering users to make conscious choices about their app usage and eliminate distractions that hinder their productivity.

4. Encouraging Mindful App Usage: App blockers aim to foster mindful app usage by raising awareness about individual app habits and their impact on well-being. By analyzing app usage patterns and providing insights, users can gain a deeper understanding of their digital behavior. This objective promotes self-reflection and empowers users to make informed decisions about their app usage.

5. Promoting Healthy Digital Habits: App blockers play a pivotal role in promoting healthy digital habits. By allowing users to create schedules, set timers, and establish app usage restrictions, individuals can develop disciplined app routines.

   This objective emphasizes the importance of establishing healthy boundaries and encourages users to adopt mindful and intentional app usage practices.

6. Supporting Digital Detox and Self-care: In today's constantly connected world, taking breaks from technology is essential for overall well-being. App blockers facilitate digital detox by providing features such as temporary app blocking or enforcing designated downtime. This objective emphasizes the importance of self-care, balance, and rejuvenation by disconnecting from digital distractions.

1.2 Project Description

The App Blocker project aims to develop an innovative Android application that empowers users to take control of their app usage and enhance their productivity, focus, and digital well-being. The application will provide a user-friendly interface and a range of features to effectively block or limit access to specific applications on the user's device.

Key Features:

1. App Blocking: The App Blocker will enable users to select and block specific applications that they find distracting or unproductive. By preventing access to these apps, users can minimize distractions and maintain focus on important tasks.

2. Schedule and Timer: Users will have the flexibility to set schedules or timers for blocking apps. This feature allows users to define specific timeframes during which access to certain applications will be restricted. For example, users can block social media apps during work hours or limit gaming apps during study sessions.

3. Customization: The App Blocker will offer customization options to meet individual user preferences and requirements. Users will be able to create personalized lists of blocked apps, adjust blocking settings, and define exceptions for certain apps or categories.

4. App Usage Insights: The application will provide users with valuable insights into their app usage patterns. Users will be able to view detailed reports and statistics on their app usage, helping them understand their digital habits and make informed decisions about managing their time more effectively.

5. Notification and Reminder: The App Blocker will send notifications and reminders to users, serving as gentle prompts to adhere to their designated app blocking schedules or timers. This feature will aid in building awareness and fostering self-discipline.

6. App Whitelist: The application will allow users to create an app whitelist, consisting of essential or productive applications that will not be blocked. This whitelist feature ensures that users can access necessary tools or apps required for work, communication, or other important tasks.

The App Blocker project will be developed for Android devices, leveraging the latest Android software development tools and best practices. The user interface will be designed to be intuitive, visually appealing, and easy to navigate.

By implementing the App Blocker, users will have the ability to manage their app usage effectively, reduce distractions, and optimize their productivity. The project's primary goal is to provide a solution that empowers users to regain control over their digital habits, promoting a healthier and more balanced relationship with technology.

In conclusion, the App Blocker project aims to develop an Android application that offers app blocking capabilities, customization options, app usage insights, and reminders to enhance productivity, focus, and digital well-being. By providing users with the tools to manage their app usage effectively, the App Blocker project seeks to support users in their journey towards a more mindful and productive digital lifestyle

2.SYSTEM STUDY

2.1 Existing System and Proposed System

Existing System:

The existing system lacks a comprehensive solution for effectively managing app usage and reducing distractions on Android devices. While some devices offer limited built-in app blocking or parental control features, they often lack customization options and flexibility. Users may resort to manual methods, such as uninstalling or disabling apps, which can be time-consuming and inconvenient. Overall, the existing system falls short in providing users with a robust and user-friendly app blocking solution that addresses their specific needs.

Proposed System:

The proposed system aims to address the limitations of the existing system by developing an advanced Android application, the App Blocker. This application will provide users with a comprehensive solution for managing their app usage, reducing distractions, and enhancing their productivity. The key features and improvements of the proposed system are as follows:

1. App Blocking and Limiting: The App Blocker will allow users to block or limit access to specific applications on their Android devices. Users can select the apps they find distracting or unproductive and set restrictions on their usage, helping them stay focused on important tasks.

2. Customization and Flexibility: Unlike the existing system, the App Blocker will offer a high degree of customization and flexibility. Users will have the ability to create personalized lists of blocked apps, adjust blocking settings based on their preferences, and define exceptions for certain apps or app categories. This level of customization ensures that users can tailor the app blocking system to their specific needs.

3. Scheduling and Timer: The App Blocker will provide scheduling and timer features, enabling users to set specific timeframes during which access to certain apps will be blocked or limited. This feature empowers users to establish dedicated periods for work, study, or relaxation, ensuring that they can effectively manage their time and reduce distractions.

4. App Usage Insights and Reports: The proposed system will offer detailed app usage insights and reports. Users will be able to access information about their app usage patterns, including total usage time, frequency, and duration. These insights will enable users to gain a better understanding of their digital habits and make informed decisions about managing their app usage effectively.

5. Notifications and Reminders: The App Blocker will send notifications and reminders to users, serving as gentle prompts to adhere to their designated app blocking schedules or timers. This feature will help users stay accountable to their goals, fostering self-discipline and promoting a healthier relationship with technology.

6. User-friendly Interface: The user interface of the App Blocker will be designed to be intuitive, visually appealing, and easy to navigate. Users will be able to access and configure app blocking settings effortlessly, ensuring a seamless user experience.

By implementing the proposed system, users will have a powerful and user-friendly app blocking solution that enhances their productivity, reduces distractions, and promotes a healthier digital lifestyle. The App Blocker fills the gaps left by the existing system, providing a comprehensive and customizable tool for managing app usage effectively.

In conclusion, the proposed system, the App Blocker, offers significant improvements over the existing system by providing enhanced customization, scheduling and timer features, app usage insights, notifications, and a user-friendly interface. These improvements aim to empower users to take control of their app usage, reduce distractions, and optimize their productivity. The proposed system presents a comprehensive and user-centric solution to effectively manage app usage on Android devices

2.2 Feasibility Study

The feasibility study of the App Blocker project aims to assess the viability and potential success of developing and implementing an app blocking application for Android devices. This study examines various aspects, including technical, economic, operational, and legal considerations, to determine the feasibility of the project.

Technical Feasibility: The technical feasibility assesses the project's compatibility with existing technology and infrastructure. In the case of the App Blocker, the Android platform provides a solid foundation for development. The necessary tools and frameworks, such as Android Studio and Java, are readily available, ensuring the project's technical feasibility. Additionally, the project can leverage existing APIs and SDKs to implement app blocking and scheduling features effectively.

Economic Feasibility: The economic feasibility analyzes the project's financial viability and potential return on investment. The App Blocker can be monetized through various strategies, such as offering a free version with limited features and a premium version with additional functionalities. Revenue streams can include in-app purchases, subscription models, or displaying targeted advertisements. A thorough market analysis is crucial to understanding the target audience, competition, and potential revenue generation, thus ensuring the economic feasibility of the project.

Operational Feasibility: The operational feasibility evaluates the project's practicality and ability to integrate with existing systems and processes. The App Blocker can be seamlessly integrated into Android devices without requiring significant changes or modifications to the user's operating system. User feedback and usability testing will be essential to refining the user interface and ensuring a smooth user experience. Additionally, continuous updates and support will be necessary to address emerging Android versions and maintain compatibility with new device models.

Legal and Ethical Feasibility: The legal and ethical feasibility examines the project's compliance with relevant laws, regulations, and ethical considerations. The App Blocker must respect user privacy by securely handling user data and complying with data protection regulations such as GDPR or CCPA. Clear and transparent privacy policies and terms of use should be provided to users. It is crucial to avoid infringing on app developers' intellectual property rights while implementing app blocking functionalities. Adhering to ethical guidelines and best practices in app development will further enhance the project's feasibility

2.3 Tools and Technology used

Developing an app blocker requires the utilization of specific tools and technologies that are commonly used in Android app development. This section focuses on the key tools and technologies used in building an app blocker using Java and XML.

1. **Android Studio:** Android Studio is the primary Integrated Development Environment (IDE) for Android app development. It provides a comprehensive set of tools and features for designing, coding, debugging, and testing Android applications. Android Studio supports Java and XML, making it the ideal choice for developing an app blocker using these technologies.

2. **Java Programming Language :** Java is the most widely used programming language for Android app development. It offers a vast ecosystem of libraries and frameworks specifically designed for building Android applications. Java provides the necessary functionality and flexibility to implement app-blocking features, handle app permissions, and manage app usage statistics in an app blocker.

3. **XML (Extensible Markup Language) :** XML is a markup language used for designing user interfaces in Android applications. It is extensively used in defining the layout and visual elements of app screens. XML allows developers to create a hierarchy of UI elements, set attributes, and specify the arrangement of UI components such as buttons, text fields, and lists. In an app blocker, XML is used to design the user interface and define the layout of screens for configuring app blocking settings.

4. **Android SDK (Software Development Kit**) : The Android Software Development Kit (SDK) is a collection of tools, libraries, and resources that enable developers to build Android applications. It provides essential components and APIs required for app development, including UI components, data storage, networking, and device functionalities. The Android SDK offers the necessary tools and APIs to implement app-blocking features, handle app permissions, and access app usage statistics in an app blocker.

5. **Android App Permissions :** App blockers often require specific permissions to access app usage statistics, monitor installed applications, and block or limit app usage.

In Android, app permissions are defined in the AndroidManifest.xml file. By specifying the required permissions, the app blocker can access the necessary information and functionalities to perform app blocking effectively.

6. **Android Material Design** : Material Design is a design language developed by Google that provides guidelines for creating visually appealing and consistent user interfaces. It focuses on using colour, typography, and motion to create a modern and intuitive user experience. Adhering to the principles of Material Design ensures that the app blocker has a visually pleasing and user-friendly interface.

Java and XML are fundamental tools and technologies used in developing an app blocker for Android devices. Android Studio provides a comprehensive development environment, while Java offers the necessary functionality and flexibility for implementing app-blocking features. XML is used to design the user interface and layout of app screens. The Android SDK, Android app permissions, and Material Design guidelines are additional tools and technologies that support the development process. By leveraging these tools and technologies, developers can create a robust and user-friendly app blocker that effectively manages app usage and enhances productivity on Android devices.

2.4 Hardware and Software Requirements

App Blocker is an Android application that helps users manage their app usage and reduce distractions on their devices. To run the app blocker effectively, certain hardware and software requirements need to be met. This section outlines the hardware and software requirements for running the app blocker smoothly on an Android device.

Hardware Requirements:

1. Android Device: The primary hardware requirement is an Android device, such as a smartphone or tablet, running Android OS version 5.0 (Lollipop) or above. The device should have sufficient processing power, memory, and storage capacity to handle the app blocker and other installed apps.

Software Requirements:

1. Operating System: The Android device must have Android OS version 5.0 or above. The app blocker may utilize features and APIs introduced in later versions of Android, so it is recommended to have the latest compatible Android OS version installed.

2. Permissions: The app blocker may require certain permissions to access app usage statistics, monitor installed applications, and block or limit app usage. The user must grant the necessary permissions to the app blocker for it to function correctly. These permissions can be managed in the device settings under "App Permissions" or a similar section.

3. App Compatibility: The app blocker should be compatible with the installed apps on the device. It should be able to recognize and block popular social media apps, games, messaging apps, and other productivity-distracting applications. Compatibility with a wide range of apps ensures that users can effectively manage their app usage across different categories.

4. Storage: Sufficient storage space is required on the device to install and run the app blocker. The exact storage requirements depend on the app blocker's size, but typically a few megabytes of free storage should be available for the installation.

5. Network Connectivity: The app blocker may require an active internet connection for certain features, such as syncing app usage statistics or downloading updates. Wi-Fi or mobile data connectivity is recommended to ensure the app blocker can function optimally.

To run the app blocker effectively, an Android device with compatible hardware and software is required. This includes an Android device running Android OS version 5.0 or above, sufficient storage space, and a stable internet connection. Granting the necessary permissions for the app blocker ensures it can access app usage statistics and monitor app activities accurately. By meeting these hardware and software requirements, users can successfully install and run the app blocker, allowing them to manage their app usage and improve productivity on their Android devices.

3.SOFTWARE REQUIREMENTS SPECIFICATION

Here are some common requirements users might have for an app blocker app:

1. Platform Compatibility: Users may expect the app blocker to be available on multiple platforms, such as Android, iOS, Windows, and macOS, to ensure it can be used on their preferred devices.

2. App Selection: Users may want the ability to select specific apps to block or restrict, allowing them to focus on specific tasks or limit distractions.

3. Scheduling: Users may require the ability to schedule specific time periods during which certain apps should be blocked automatically. For example, they may want to block social media apps during work hours or during bedtime.

4. Customizable Block Rules: Users may desire the flexibility to create customized blocking rules. This could include setting time limits for app usage, defining exceptions for specific contacts or urgent notifications, or categorizing apps into different groups for different blocking configurations.

5. Whitelisting: Users may need the ability to whitelist certain apps that should never be blocked, even during scheduled blocking periods. These could include essential productivity tools, communication apps, or emergency services.

6. Usage Insights: Users might appreciate access to usage statistics and insights, such as how much time they spend on different apps or categories of apps. This information can help them understand their app usage patterns and make informed decisions about blocking or limiting certain apps.

7. Password Protection: Users may expect the app blocker to offer a secure password or PIN protection to prevent unauthorized access and changes to the app blocking settings.

8. Intuitive User Interface: Users would appreciate a user-friendly interface that is easy to navigate and configure. Clear instructions and visual cues can enhance the overall user experience.

9. Performance and Battery Efficiency: Users may expect the app blocker app to have minimal impact on device performance and battery life, ensuring that their devices remain responsive and usable.

10. Regular Updates and Support: Users may desire regular updates to the app blocker app to ensure compatibility with new operating system versions and to address any security or performance issues. They may also appreciate responsive customer support for addressing their queries or troubleshooting problems.

Remember, these requirements can vary depending on the target audience and the specific use case for the app blocker app. It is important to conduct user research and gather feedback to refine the requirements further and meet the specific needs of your target users.

3.2 Functional Requirements

Blocking apps

- A list of the installed apps on the user's device is viewable through the app.
- The user is allowed to pick particular apps from the list to block.
- The app provides a list of installed apps on the user's device.
- The user is able to select specific apps to block.
- Once an app is blocked, it is prevented from launching or running on the device.
- The app will provide an option to unblock previously blocked apps.
- When an app is blocked, the device is not allow it to start or execute.
- Different blocking modes, such as blocking during particular time periods or depending on user-defined triggers, is able to be configured in the app.
- Both banning system apps and third-party apps supported by the app.

User Authentication

- To make certain that only authorized users can access and edit the app blocking settings, the app has to include a user authentication mechanism.
- A username and password, biometric authentication, or other safe authentication techniques can all be used to implement user authentication.
- The app supports user authentication to ensure only authorized users can access and modify the app blocking settings.
- User authentication can be implemented using a username and password or other authentication mechanisms.

App Monitoring

- The device is watched by the app constantly for any attempts to open banned apps.
- When a blocked app is found, the app tells the user via notifications or alerts about the blocked app's attempted activation.
- The user have the option to alter the notification settings, including whether or not notifications are enabled or disabled, the notification style, and the priority levels.
- The app continuously monitor the device for any attempts to launch blocked apps.
- When a blocked app is detected, the app should provide notifications or alerts to the user, informing them of the blocked app's attempted launch.

Blocking Profiles and Scheduling

- With various sets of restricted apps and settings, users of the app can build multiple blocking profiles.
- Specific blocking profiles can be activated or deactivated by users depending on their preferences or specific conditions.
- The blocking profiles can be scheduled using the app, allowing users to specify the times when the chosen profile will be active.

Settings and Configuration

- The app provides a settings section where users can configure various aspects of the app.
- Settings include options to enable/disable app blocking, customize notifications, set up authentication, etc.
- The app ensures the security and privacy of user data, including information on prohibited apps and authentication credentials.
- Sensitive information and user passwords are maintained securely using encryption methods.
- If necessary, proper protocols and encryption are used to safeguard communication with any back-end services.

3.3 Non-Functional Requirements

User Interface

The app's user interface is sleek and simple, and it adheres to the Android Material Design standards. The user interface is pleasing to the eye, responsive, and offers a smooth user experience.

Performance

Even while blocking many apps at once, the user experience is made to be as seamless as possible thanks to performance and responsiveness optimizations made to the app.

The monitoring features of the software work effectively in the background with minimal negative effects on the device's performance or battery life.

Privacy and security

The app ensures the security and privacy of user data, including information on prohibited apps and authentication credentials.

Encryption techniques are used to safely store user passwords and sensitive information.
Using the proper protocols and encryption, communications with any back-end services, if applicable, are protected.

Compatibility

The software supports a variety of Android devices, including those with various screen sizes, resolutions, and operating systems.

Languages and Frameworks for Programming

Java is the programming language used to create the app for the Android operating system.
the development environment was Android Studio.

4. SYSTEM PROSPECTIVE

4.1 SYSTEM PERSPECTIVE

App blocker app is designed to help users limit or manage their time spent on social media platforms and other digital distractions. From a system perspective, this app can be analyzed and described in several key aspects. Here's an overview of the system perspective :

- System Purpose
  The purpose of the app blocker app is to help users control and manage their app usage on their devices. It allows users to set restrictions, block specific apps or categories of apps, and promote healthy digital habits.

- Users and Roles:
  Role of app user in the system is to install and interact with the app, to select the apps they want to block.

Here's an overview of the key components and functionalities of the app, it aims to capture the main components :

USER INTERFACE

- A user-friendly interface that allows users to easily navigate and interact with the app. Providing intuitive controls for configuring app blocking rules, managing blocked apps, and adjusting settings.
- Implementing visual indicators to display the status of blocked apps and time restrictions.

Blocking Engine:

- The blocking engine is the core component responsible for monitoring and restricting access to specified apps according to user defined rules.
- A mechanism to define and enforce app blocking rules.

- Allowing users to specify which apps should be blocked and when (e.g., specific times, duration limits).

3.Configuration and Preferences:

- The app allow users to define their blocking preferences. This includes setting specific time intervals and week days and choosing which apps to block.

4. Notifications :

- The app provides notifications to users when a blocked app is accessed. These notifications can serve as reminders to stay focused or to discourage excessive usage.

5. Cross-Platform Support:

- Ensuring compatibility with different devices.

Above all the functionalities and specifications are explained in Software Requirement Specification of this report.

4.2 ER MODEL

4.2.1 Entities and its attributes

- **USER** :

The "User" entity represents the app users.
Attributes : user_id, name
Primary key - user_id

- **APPS**

The "Apps" entity represents apps in users phone.
Attributes : app_id, app_name

- **BLOCKLIST**

The "BlockList" entity represents apps in user's blocklist.
Attributes : blocklist_id, user_id, app_id, app_name
Primary key - blocklist_id
Foreign key - user_id (references the user entity),
app_id (references apps attribute)

- **BLOCKED APPS** :

The "BlockedApp" entity represents an app that is blocked within a blocklist.
Attributes : blockedapp_id , blocklist_id , app_name (the name of the blocked app)
Primary key - blockedapp_id
Foreign key - blocklist_id(references the BlockList entity)

4.2.2 Types of attributes are specified below :

- user_id - single-valued, Stored, Simple
- name - Composite attribute, Single-valued
- app_id - multi-valued, Stored, Simple
- app_name - multi-valued, Simple
- blocklist_id - Multivalued, Simple
- blockedapp_id - Multivalued, Simple

4.2.3 Schema overview

- USER(user_id,name)
- APPS(app_id,app_name)
- BLOCKLIST(blockedlist_id, user_id, app_id, app_name)
- BLOCKEDAPP(blockedapp_id, blockedlist_id,app_name)

4.2.4 Relationship analysis

- Between user and Blocked lists : ***One to one***

Each user can have one blocklist and each blocklist with scheduled rules is associated with only one user.

- Between user and apps : ***Many to many***

Each user can have multiple apps and each app can be installed by multiple users.

- Between user and Blocked apps : **Many *to many***

Each user can have multiple Blocked apps and each blocked app can be blocked by multiple users.

- Between Blocked list and apps : ***Many to many***

Each Blocked list can have multiple apps.

- Between Blocked list and blocked apps : ***Many to many***

Each Blocked list can have multiple blocked apps and each blocked app can be a part of different blocked lists.

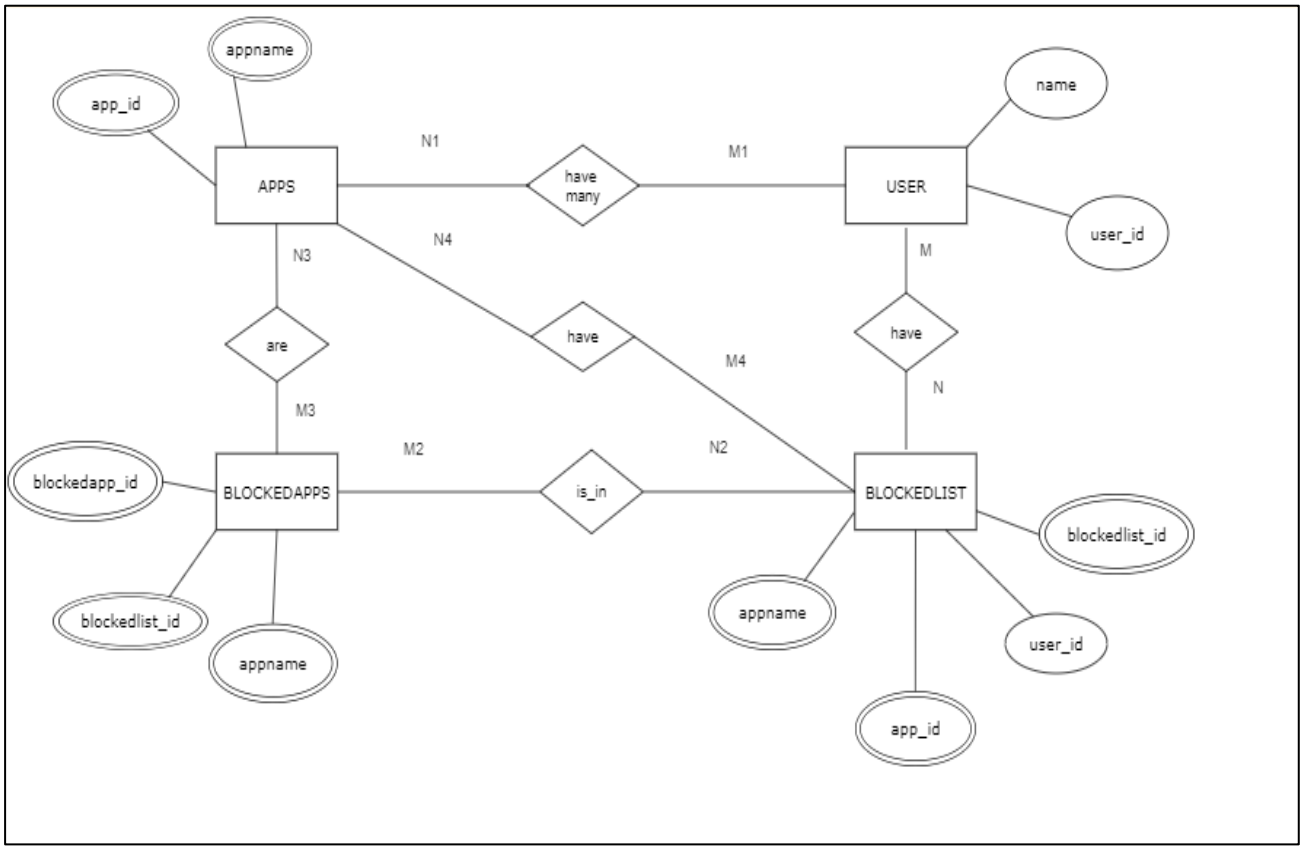

Fig 1. ER Diagram

4.3 Context Diagram

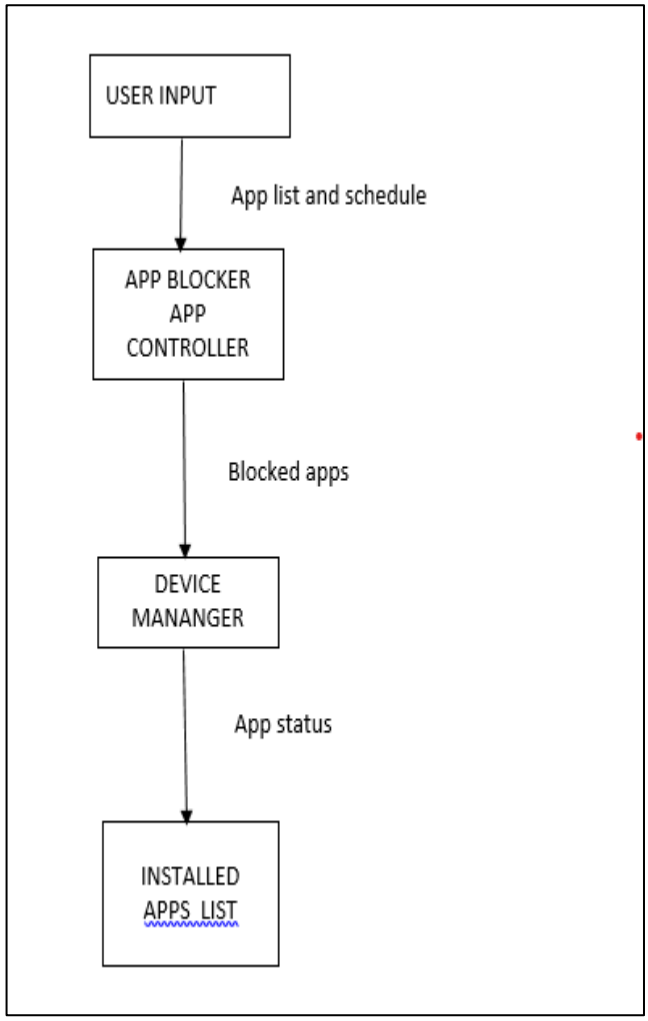

Fig 2. DFD

In this DFD, the main components are:

User : Represents the user interaction with the app, such as selecting apps to block or unblock.

App Blocker App Controller: This component is responsible for managing the app blocking functionality. It receives the user's input and interacts with other components accordingly.

App List: Represents the list of installed apps on the device.

Blocked Apps: Represents the list of apps that are currently blocked.

Device Manager: Handles the communication with the device and system-level operations, such as retrieving the list of installed apps and blocking/unblocking apps.

The arrows in the diagram represent the flow of data between the components. Here's a brief explanation of the data flow:

- The user provides input by selecting apps to block or unblock through the User Input component.
- The selected apps' information is passed to the App Blocker App Controller.
- The App Blocker App Controller communicates with the Device Manager to block or unblock the selected apps.
- The Device Manager interacts with the App Database to update the app status.
- The App Database stores the information about the installed apps and their status.
- The Device Manager retrieves the updated app status from the App Database.
- The App Blocker App Controller receives the updated information and can display it to the user if needed.

This DFD provides a high-level overview of the data flow in the app blocker app project, emphasizing the interactions between the user, the app controller, the device manager, and the storage.

4.4 USE CASE DIAGRAM

App Blocker App: Represents the main app component that provides the functionality to block and unblock apps. It interacts with the User and supports multiple use cases.

User (App User): Represents the user of the app, who interacts with the App Blocker App to perform various actions.

Use Case 1: Block App: This use case represents the action of blocking a specific app. The user selects an app from the app list and triggers the block action.

Use Case 2: Unblock App: This use case represents the action of unblocking a previously blocked app. The user selects a blocked app from the blocked apps list and triggers the unblock action.

Use Case 3: View Blocked Apps: This use case represents the action of viewing the list of currently blocked apps. The user can access this feature to see which apps are currently blocked.

Use Case 4: View App List: This use case represents the action of viewing the list of installed apps. The user can access this feature to see all the apps available on their device.

The use case diagram provides a high-level view of the interactions between the App Blocker App and the user. It helps to identify the main actions that the user can perform and how they relate to the functionality of the app.

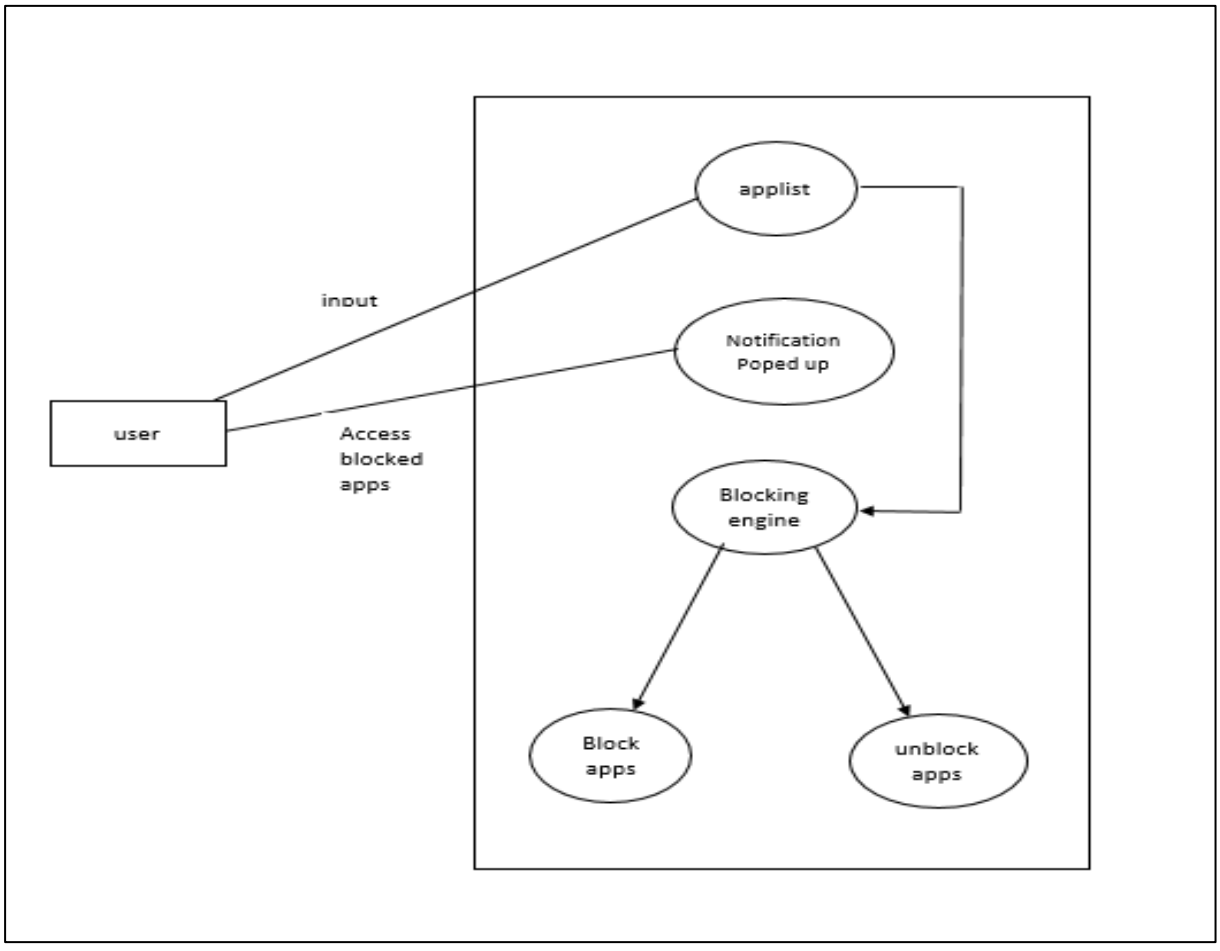

Fig 3 : Use Case Diagram

4.5 Sequence Diagram

In this sequence diagram, the app follows these steps:

1. The user interacts with the User Interface to open the App Blocker app.

2. The User Interface calls the `openAppBlocker( )` function in the AppListController.

3. The AppListController communicates with the DeviceManager to retrieve a list of installed apps.

5. The DeviceManager returns the installed apps to the AppListController.

6. The AppListController displays the installed apps through the User Interface.

7. The user selects the apps they want to block through the User Interface and schedule them (schedule.java class).

8. The AppListController gets the selected apps from the User Interface.

9. The AppListController communicates with the DeviceManager to block the selected apps.

10. The DeviceManager interacts with the AppDatabase to update the app status as blocked.

Updated screen is shown by getInstalledApps.java class

11. The DeviceManager returns the list of blocked apps to the AppListController.

12. The AppListController displays the blocked apps through the User Interface (showBlockingInfo( )).

This sequence diagram illustrates the interaction between the various components involved in the app blocking process. Keep in mind that this is a simplified example, and the actual implementation may involve additional steps and interactions based on the specific requirements of the app blocker app.

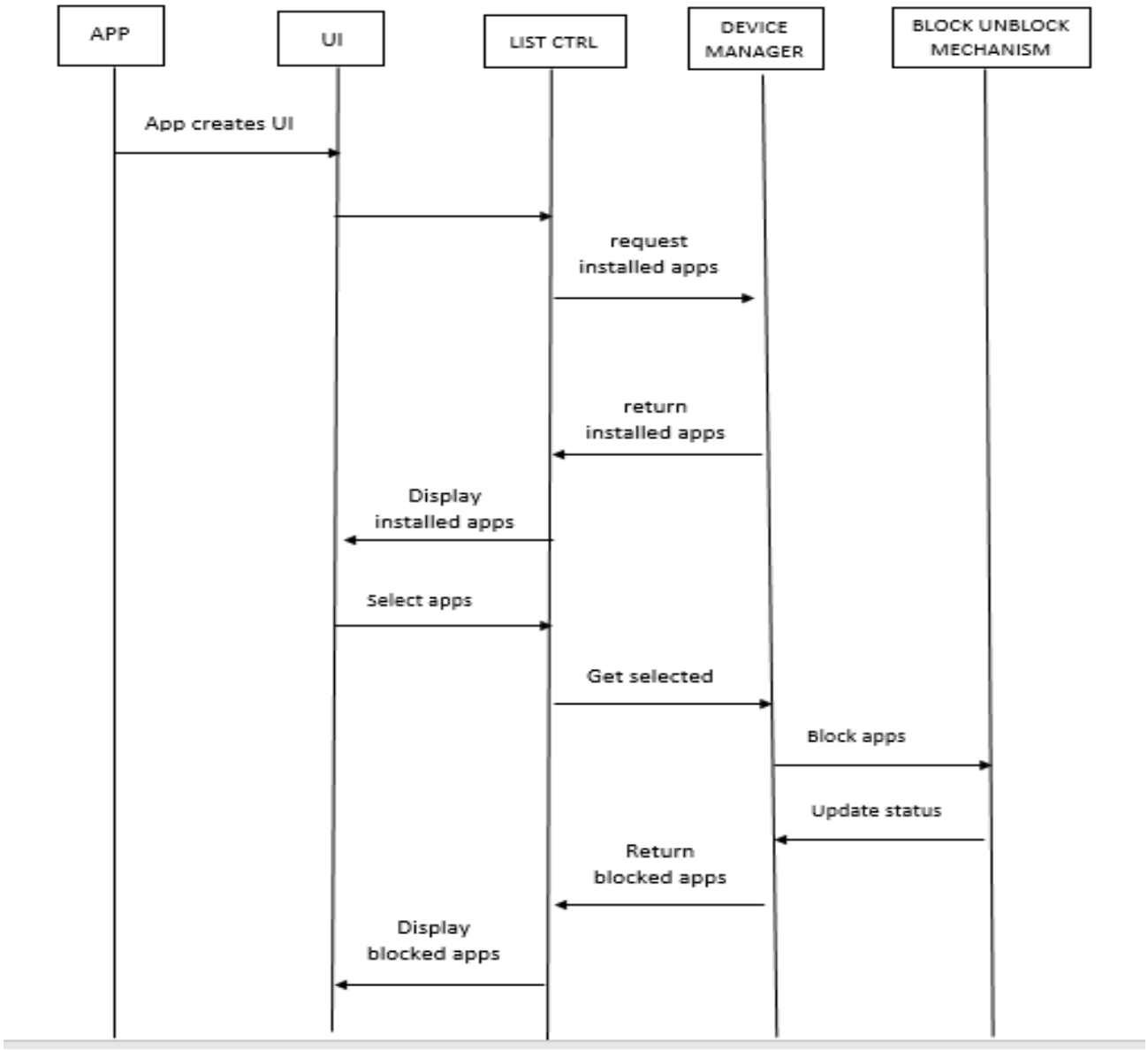

Fig 4 Sequence Diagram

5.IMPLEMENTATION

AndroideManifest.xml

The provided code is an XML file that represents the AndroidManifest.xml file for an app blocker Android application. The AndroidManifest.xml file is a crucial configuration file that provides essential information about the app to the Android operating system.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    package="com.example.appblockr">

    <uses-permission android:name="android.permission.DISABLE_KEYGUARD" />
    <uses-permission android:name="android.permission.GET_TASKS" />
    <uses-permission android:name="android.permission.FOREGROUND_SERVICE" />
    <uses-permission android:name="android.permission.KILL_BACKGROUND_PROCESSES" />
    <uses-permission android:name="android.permission.REORDER_TASKS" />
    <uses-permission android:name="android.permission.WAKE_LOCK" />
    <uses-permission android:name="android.permission.REQUEST_IGNORE_BATTERY_OPTIMIZATIONS" />
    <uses-permission android:name="android.permission.SYSTEM_ALERT_WINDOW" />
    <uses-permission
        android:name="android.permission.PACKAGE_USAGE_STATS"
        tools:ignore="ProtectedPermissions" />
    <uses-permission
        android:name="android.permission.QUERY_ALL_PACKAGES"
        tools:ignore="QueryAllPackagesPermission" />

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:supportsRtl="true"
        android:theme="@style/Theme.Appsift">
        <activity android:name=".Schedule"></activity>
        <activity android:name=".IntroScreen" />
        <activity android:name=".MainActivity" />
        <activity android:name=".About" />
        <activity android:name=".ScreenBlocker" />
        <activity android:name=".SplashScreen">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity android:name=".LockedApps" />
        <activity android:name=".ShowAllApps" />

        <receiver
            android:name=".broadcast.ReceiverApplock"
            android:enabled="true"
            android:exported="true" />
        <receiver
            android:name=".broadcast.RestartServiceWhenStopped"
            android:enabled="true"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.BOOT_COMPLETED" />
            </intent-filter>
        </receiver>

        <service
            android:name=".services.ServiceApplock"
            android:enabled="true"
            android:exported="false" />
```

About.java

```
<service
        android:name=".services.ServiceApplockJobIntent"
        android:exported="true"
        android:permission="android.permission.BIND_JOB_SERVICE" />
    </application>

</manifest>


ackage com.example.appblockr;


import android.app.AlertDialog;
import android.content.DialogInterface;
import android.content.Intent;
import android.content.pm.PackageInfo;
import android.content.pm.PackageManager;
import android.os.Bundle;
import android.util.Log;
import android.view.MenuItem;
import android.view.View;
import android.widget.LinearLayout;
import android.widget.TextView;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;

import com.google.android.material.bottomnavigation.BottomNavigationView;

import mehdi.sakout.aboutpage.AboutPage;


public class About extends AppCompatActivity {
    int verCode;
    String versionName;
    TextView termsAndCondition;
    TextView privacyPolicy;


    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        addIconToBar();
        setTitle(" About");
        versionElement();
        String pp_string = getResources().getString(R.string.privacy_policy);
        String tm_string = getResources().getString(R.string.terms_conditions);

        View aboutPage = new AboutPage(this)
                .isRTL(false)
                .setImage(R.drawable.ic_launcher_foreground)

                .setDescription("AppBlockr \n Version: " + versionName + verCode)

                .addGroup("Connect with us")
                .addEmail("sanjanagpt471@gmail.com")
                .addWebsite("url here")

                .create();

        setContentView(R.layout.activity_about);
        LinearLayout linearLayout = findViewById(R.id.info);
        linearLayout.addView(aboutPage);
```

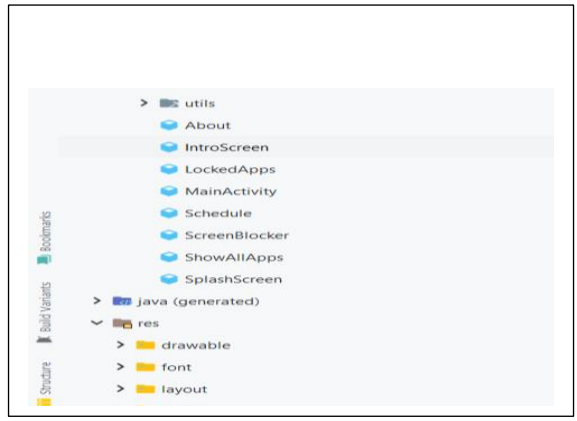

Fig 5 - Utils

```
    termsAndCondition = findViewById(R.id.terms_conditions);
        privacyPolicy = findViewById(R.id.privacy_policy);

        termsAndCondition.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                popupMessage(tm_string, "Terms & Conditions");
            }
        });
        privacyPolicy.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                popupMessage(pp_string, "Privacy Policy");
            }
        });

        BottomNavigationView bottomNavigationView = findViewById(R.id.bottom_navigation);
        bottomNavigationView.setSelectedItemId(R.id.nav_settings);
        bottomNavigationView.setOnNavigationItemSelectedListener(new
BottomNavigationView.OnNavigationItemSelectedListener() {
            @Override
            public boolean onNavigationItemSelected(@NonNull MenuItem item) {
                switch (item.getItemId()) {
                    case R.id.nav_locked_apps:
                        startActivity(new Intent(getApplicationContext(),
                                MainActivity.class));
                        overridePendingTransition(0, 0);
                        return true;
                    case R.id.nav_all_apps:
                        startActivity(new Intent(getApplicationContext(),
                                ShowAllApps.class));
                        overridePendingTransition(0, 0);
                        return true;
                    case R.id.nav_settings:

                        return true;
                }
                return false;
            }
        });
    }

    public void popupMessage(String message, String title) {
        AlertDialog.Builder alertDialogBuilder = new AlertDialog.Builder(this);
        alertDialogBuilder.setMessage(message);
        alertDialogBuilder.setIcon(R.drawable.ic_launcher_foreground);
        alertDialogBuilder.setTitle(title);
        alertDialogBuilder.setNegativeButton("ok", new DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialogInterface, int i) {
                Log.d("internet", "Ok btn pressed");
            }
        });
        AlertDialog alertDialog = alertDialogBuilder.create();
        alertDialog.show();
    }

    private void addIconToBar() {
        getSupportActionBar().setDisplayShowHomeEnabled(true);
        getSupportActionBar().setLogo(R.drawable.ic_launcher_foreground);
        getSupportActionBar().setDisplayUseLogoEnabled(true);

        setContentView(R.layout.activity_about);
    }
```

```
  boolean versionElement() {
      try {
         PackageInfo pInfo =
getApplicationContext().getPackageManager().getPackageInfo(getApplicationContext().getPackageName(), 0);
         versionName = pInfo.versionName;
         verCode = pInfo.versionCode;
      } catch (PackageManager.NameNotFoundException e) {
         e.printStackTrace();
         return false;
      }
      return true;

   }
}
```

IntroScreen.java

```
package com.example.appblockr;

import android.app.ActionBar;
import android.app.AlertDialog;
import android.content.DialogInterface;
import android.content.Intent;
import android.os.Bundle;
import android.text.Html;
import android.view.View;
import android.view.Window;
import android.widget.Button;
import android.widget.LinearLayout;
import android.widget.TextView;

import androidx.appcompat.app.AppCompatActivity;
import androidx.viewpager.widget.ViewPager;

import com.example.appblockr.adapter.SliderAdapter;

public class IntroScreen extends AppCompatActivity {
   String pp_string;
   private ViewPager mSlideViewPager;
   private LinearLayout mDotLayout;
   private SliderAdapter sliderAdapter;
   private TextView[] mDots;
   private Button nextBtn;
   private Button backBtn;
   private int currentPage;
   ViewPager.OnPageChangeListener viewListener = new ViewPager.OnPageChangeListener() {
      @Override
      public void onPageScrolled(int position, float positionOffset, int positionOffsetPixels) {

      }

      @Override
      public void onPageSelected(int position) {
         addDotsIndicator(position);
         currentPage = position;
         sliderAdapter.hidePrivacyPopup();
         if (position == 0) {
            nextBtn.setEnabled(true);
            backBtn.setEnabled(false);
            backBtn.setVisibility(View.INVISIBLE);
            nextBtn.setText("Next");
            backBtn.setText("");
```

```
} else if (position == 1) {
            nextBtn.setEnabled(true);
            backBtn.setEnabled(true);
            backBtn.setVisibility(View.VISIBLE);
            nextBtn.setText("Next");
            backBtn.setText("Back");
        } else {
            sliderAdapter.showPrivacyPopup();
            nextBtn.setEnabled(true);
            backBtn.setEnabled(false);
            backBtn.setVisibility(View.INVISIBLE);
            nextBtn.setText("Accept");
            backBtn.setText("");
            sliderAdapter.privacyPopup.setOnClickListener(new View.OnClickListener() {
                @Override
                public void onClick(View v) {
                    popupMessage(pp_string, "Privacy Policy");
                }
            });
        }
    }

    @Override
    public void onPageScrollStateChanged(int state) {

    }
};

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    requestWindowFeature(Window.FEATURE_NO_TITLE);
    getWindow().setLayout(ActionBar.LayoutParams.MATCH_PARENT, ActionBar.LayoutParams.MATCH_PARENT);
    setContentView(R.layout.activity_intro_screen);
    pp_string = getResources().getString(R.string.privacy_policy);
    getSupportActionBar().hide();
    mSlideViewPager = findViewById(R.id.slideViewPager);
    mDotLayout = findViewById(R.id.layout_dots);
    nextBtn = findViewById(R.id.nextBtn);
    backBtn = findViewById(R.id.backBtn);
    sliderAdapter = new SliderAdapter(this);
    mSlideViewPager.setAdapter(sliderAdapter);
    addDotsIndicator(0);
    mSlideViewPager.addOnPageChangeListener(viewListener);
    nextBtn.setEnabled(true);
    backBtn.setEnabled(false);
    backBtn.setVisibility(View.INVISIBLE);
    nextBtn.setText("Next");
    backBtn.setText("");


    nextBtn.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            if (currentPage == 0) {
                sliderAdapter.hidePrivacyPopup();
            }
            if (currentPage == 1) {
                sliderAdapter.hidePrivacyPopup();
            }
            if (currentPage == 2) {
                Intent myIntent = new Intent(IntroScreen.this, MainActivity.class);
                IntroScreen.this.startActivity(myIntent);
                sliderAdapter.showPrivacyPopup();
            }
            mSlideViewPager.setCurrentItem(currentPage + 1);

        }
    });
```

```
    backBtn.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                mSlideViewPager.setCurrentItem(currentPage - 1);

            }
        });


    }

    public void popupMessage(String message, String title) {
        AlertDialog.Builder alertDialogBuilder = new AlertDialog.Builder(this);
        alertDialogBuilder.setMessage(message);
        alertDialogBuilder.setIcon(R.drawable.ic_launcher_foreground);
        alertDialogBuilder.setTitle(title);
        alertDialogBuilder.setNegativeButton("ok", new DialogInterface.OnClickListener() {

            @Override
            public void onClick(DialogInterface dialogInterface, int i) {

            }
        });
        AlertDialog alertDialog = alertDialogBuilder.create();
        alertDialog.show();
    }

    public void addDotsIndicator(int position) {
        mDots = new TextView[3];
        mDotLayout.removeAllViews();
        for (int i = 0; i < mDots.length; i++) {
            mDots[i] = new TextView(this);
            mDots[i].setText(Html.fromHtml("&#8226;"));
            mDots[i].setTextSize(35);
            mDots[i].setTextColor(getResources().getColor(R.color.lightBlue));
            mDotLayout.addView(mDots[i]);
        }

        if (mDots.length > 0) {
            mDots[position].setTextColor(getResources().getColor(R.color.white));
        }
    }
}
```

LockedApps.java

```
ackage com.example.appblockr;

import android.app.ProgressDialog;
import android.content.Context;
import android.content.DialogInterface;
import android.content.pm.ApplicationInfo;
import android.graphics.drawable.Drawable;
import android.os.Bundle;

import androidx.appcompat.app.AppCompatActivity;
import androidx.recyclerview.widget.LinearLayoutManager;
import androidx.recyclerview.widget.RecyclerView;

import com.example.appblockr.adapter.LockedAppAdapter;
import com.example.appblockr.model.AppModel;
import com.example.appblockr.shared.SharedPrefUtil;

import java.util.ArrayList;
import java.util.List;
```

```
public class LockedApps extends AppCompatActivity {

    RecyclerView recyclerView;
    List<AppModel> apps = new ArrayList<>();
    LockedAppAdapter adapter;
    ProgressDialog progressDialog;
    Context ctx;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_locked_apps);
        recyclerView = findViewById(R.id.lockedAppsListt);
        adapter = new LockedAppAdapter(apps, this);
        recyclerView.setLayoutManager(new LinearLayoutManager(this));
        recyclerView.setAdapter(adapter);
        progressDialog = new ProgressDialog(this);
        progressDialog.setOnShowListener(new DialogInterface.OnShowListener() {
            @Override
            public void onShow(DialogInterface dialog) {
                getLockedApps();
            }
        });
    }

    @Override
    protected void onResume() {
        super.onResume();
        progressDialog.setTitle("Fetching Apps");
        progressDialog.setMessage("Loading");
        progressDialog.show();
    }

    public void getLockedApps() {
        List<String> prefAppList = SharedPrefUtil.getInstance(this).getLockedAppsList();
        List<ApplicationInfo> packageInfos = getPackageManager().getInstalledApplications(0);
        for (int i = 0; i < packageInfos.size(); i++) {
            if (packageInfos.get(i).icon > 0) {
                String name = packageInfos.get(i).loadLabel(getPackageManager()).toString();
                Drawable icon = packageInfos.get(i).loadIcon(getPackageManager());
                String packageName = packageInfos.get(i).packageName;

                if (prefAppList.contains(packageName)) {
                    apps.add(new AppModel(name, icon, 1, packageName));
                } else {
                    continue;
                }

            }

        }
        adapter.notifyDataSetChanged();
        progressDialog.dismiss();
    }
}
```

ScreenBlocker.java

```
package com.example.appblockr;

import android.content.pm.ApplicationInfo;
import android.content.pm.PackageManager;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.ImageView;
```

```
import android.widget.TextView;



import androidx.annotation.Nullable;
import androidx.appcompat.app.AppCompatActivity;

import com.example.appblockr.shared.SharedPrefUtil;

public class ScreenBlocker extends AppCompatActivity {
    Button close_btn;

    @Override
    protected void onCreate(@Nullable Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        getSupportActionBar().hide();
        setContentView(R.layout.activity_screen_blocker);
        initIconApp();

        close_btn = findViewById(R.id.close_block_screen_btn);
        close_btn.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                onBackPressed();
            }
        });

    }

    private void initIconApp() {
        if (getIntent().getStringExtra("broadcast_receiver") != null) {
            ImageView icon = findViewById(R.id.app_icon);
            TextView blockInfo = findViewById(R.id.empty_blocked_list_text);
            String current_app = new SharedPrefUtil(this).getLastApp();
            ApplicationInfo applicationInfo = null;
            try {
                applicationInfo = getPackageManager().getApplicationInfo(current_app, 0);
            } catch (PackageManager.NameNotFoundException e) {
                e.printStackTrace();
            }
            icon.setImageDrawable(applicationInfo.loadIcon(getPackageManager()));
            blockInfo.setText("OOPS!!!! You have to work. "+
applicationInfo.loadLabel(getPackageManager()).toString().toUpperCase() + " is blocked by AppBlockr. FOCUS FOCUS FOCUS.");
        }
    }


    @Override
    protected void onPause() {
        super.onPause();
    }

    @Override
    public void onBackPressed() {
        super.onBackPressed();
    }


}
```

ShowAllApps,java

```
package com.example.appblockr;

import android.app.ProgressDialog;
```

```
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.content.pm.ActivityInfo;



import android.content.pm.PackageManager;
import android.content.pm.ResolveInfo;
import android.graphics.drawable.Drawable;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;
import android.view.inputmethod.EditorInfo;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;
import androidx.appcompat.widget.SearchView;
import androidx.recyclerview.widget.GridLayoutManager;
import androidx.recyclerview.widget.RecyclerView;

import com.example.appblockr.adapter.AllAppAdapter;
import com.example.appblockr.model.AppModel;
import com.example.appblockr.shared.SharedPrefUtil;
import com.google.android.material.bottomnavigation.BottomNavigationView;

import java.util.ArrayList;
import java.util.List;

public class ShowAllApps extends AppCompatActivity {
    RecyclerView recyclerView;
    List<AppModel> apps = new ArrayList<>();
    AllAppAdapter adapter;
    ProgressDialog progressDialog;
    Context ctx;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_show_all_apps);
        setTheme(R.style.Theme_Appsift);
        addIconToBar();
        setTitle(" Installed Apps");
        BottomNavigationView bottomNavigationView = findViewById(R.id.bottom_navigation);
        bottomNavigationView.setSelectedItemId(R.id.nav_all_apps);
        bottomNavigationView.setOnNavigationItemSelectedListener(new
BottomNavigationView.OnNavigationItemSelectedListener() {
            @Override
            public boolean onNavigationItemSelected(@NonNull MenuItem item) {
                switch (item.getItemId()) {
                    case R.id.nav_locked_apps:
                        startActivity(new Intent(getApplicationContext(),
                                MainActivity.class));
                        overridePendingTransition(0, 0);
                        return true;
                    case R.id.nav_all_apps:
                     /*  startActivity(new Intent(getApplicationContext(),
                                ShowAllApps.class));
                        overridePendingTransition(0,0);*/
                        return true;
                    case R.id.nav_settings:
                        startActivity(new Intent(getApplicationContext(),
                                About.class));
                        overridePendingTransition(0, 0);
                        return true;
                }
                return false;
            }
```

```
        });

        recyclerView = findViewById(R.id.recycleview);
        adapter = new AllAppAdapter(apps, this);
        recyclerView.setLayoutManager(new GridLayoutManager(this, 5));



recyclerView.setAdapter(adapter);
        progressDialog = new ProgressDialog(this);
        progressDialog.setOnShowListener(new DialogInterface.OnShowListener() {
            @Override
            public void onShow(DialogInterface dialog) {
                getInstalledApps();
            }
        });
    }

    @Override
    protected void onResume() {
        super.onResume();
        progressDialog.setTitle("Fetching Apps");
        progressDialog.setMessage("Loading");
        progressDialog.show();
    }

    private void addIconToBar() {
        getSupportActionBar().setDisplayShowHomeEnabled(true);
        getSupportActionBar().setLogo(R.drawable.ic_launcher_foreground);
        getSupportActionBar().setDisplayUseLogoEnabled(true);

        setContentView(R.layout.activity_show_all_apps);
    }

    public void getInstalledApps() {
        List<String> prefLockedAppList = SharedPrefUtil.getInstance(this).getLockedAppsList();

        PackageManager pk = getPackageManager();
        Intent intent = new Intent(Intent.ACTION_MAIN, null);
        intent.addCategory(Intent.CATEGORY_LAUNCHER);
        List<ResolveInfo> resolveInfoList = pk.queryIntentActivities(intent, 0);
        for (ResolveInfo resolveInfo : resolveInfoList) {
            ActivityInfo activityInfo = resolveInfo.activityInfo;
            String name = activityInfo.loadLabel(getPackageManager()).toString();
            Drawable icon = activityInfo.loadIcon(getPackageManager());
            String packageName = activityInfo.packageName;
            if (!packageName.matches("com.robocora.appsift|com.android.settings")) {
                if (!prefLockedAppList.isEmpty()) {
                    //check if apps is locked
                    if (prefLockedAppList.contains(packageName)) {
                        apps.add(new AppModel(name, icon, 1, packageName));
                    } else {
                        apps.add(new AppModel(name, icon, 0, packageName));
                    }
                } else {
                    apps.add(new AppModel(name, icon, 0, packageName));
                }
            } else {
                //do not add settings to app list
            }

        }
        adapter.notifyDataSetChanged();
        progressDialog.dismiss();

    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
```

```
        MenuInflater inflater = getMenuInflater();
        inflater.inflate(R.menu.search_menu, menu);
        MenuItem searchItem = menu.findItem(R.id.action_search);
        SearchView searchView = (SearchView) searchItem.getActionView();
        searchView.setQueryHint("Search...");
        searchView.setImeOptions(EditorInfo.IME_ACTION_DONE);
        searchView.setOnQueryTextListener(new SearchView.OnQueryTextListener() {


    @Override
            public boolean onQueryTextSubmit(String query) {
                return false;
            }

            @Override
            public boolean onQueryTextChange(String newText) {
                String userInput = newText.toLowerCase();
                ArrayList<AppModel> newList = new ArrayList<>();
                for (AppModel app : apps) {
                    if (app.getAppName().toLowerCase().contains(userInput)) {
                        newList.add(app);
                    }
                }
                adapter.updateList(newList);
                return false;
            }
        });
        return true;
    }

}
```

Activity_about.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".More">

    <LinearLayout
        xmlns:android="http://schemas.android.com/apk/res/android"
        android:layout_width="wrap_content"
        android:id="@+id/info"
        android:layout_height="wrap_content"
        android:orientation="vertical"
        tools:ignore="MissingConstraints">
    </LinearLayout>
```

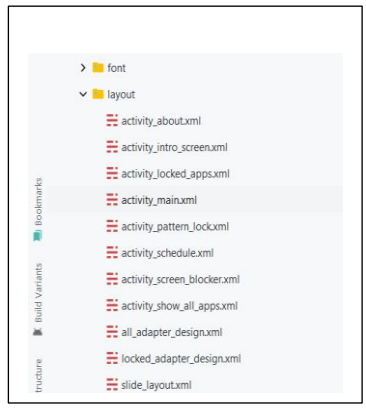

Fig 6: layout

```
    <TextView
        android:paddingStart="8dp"
        android:id="@+id/privacy_policy"
        android:drawableLeft="@drawable/ic_baseline_format_list"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:paddingVertical="10dp"
        android:textSize="18dp"
        android:text="Privacy Policy"
        app:layout_constraintTop_toBottomOf="@+id/info"
        tools:ignore="MissingConstraints" />
    <View
        android:layout_width="match_parent"
```

```
        android:layout_height="1dp"
        app:layout_constraintTop_toBottomOf="@+id/privacy_policy"
        android:background="@color/lightGrey"

        tools:ignore="MissingConstraints" />




 <TextView
        android:drawableLeft="@drawable/ic_baseline_format_list"
        android:textSize="18dp"
        android:id="@+id/terms_conditions"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:paddingStart="8dp"
        android:paddingVertical="10dp"
        android:text="Terms and Conditions"

        app:layout_constraintTop_toBottomOf="@+id/privacy_policy"
        tools:ignore="MissingConstraints" />
    <View
        android:layout_width="match_parent"
        android:layout_height="1dp"
        app:layout_constraintTop_toBottomOf="@+id/terms_conditions"
        android:background="@color/lightGrey"
        tools:ignore="MissingConstraints" />

    <com.google.android.material.bottomnavigation.BottomNavigationView
        android:id="@+id/bottom_navigation"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"

        android:layout_alignParentBottom="true"
        app:itemBackground="@color/darkBlue"
        app:itemIconTint="@drawable/selector"
        app:itemTextColor="@drawable/selector"
        app:layout_constraintBottom_toBottomOf="parent"
        app:menu="@menu/menu_navigation" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

Activity_Intro_screen.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@drawable/main_background"
    tools:context=".IntroScreen">

    <androidx.viewpager.widget.ViewPager
        android:id="@+id/slideViewPager"
        android:layout_width="match_parent"

        android:layout_height="658dp"
        android:layout_above="@+id/layout_dots"
        android:layout_marginBottom="24dp" />s

    <LinearLayout
        android:id="@+id/layout_dots"
        android:layout_alignParentBottom="true"
        android:layout_width="match_parent"
        android:gravity="center"
        android:layout_height="50dp"
```

```
        android:orientation="horizontal" >


    </LinearLayout>




  <Button
        android:id="@+id/backBtn"
        android:layout_width="wrap_content"
        android:layout_height="match_parent"
        android:layout_marginLeft="8dp"
        android:layout_gravity="left"
        android:layout_below="@+id/slideViewPager"
        android:layout_weight="1"
        android:text="Back" />

    <Button
        android:id="@+id/nextBtn"
        android:layout_width="wrap_content"
        android:layout_height="match_parent"
        android:layout_marginRight="8dp"
        android:layout_below="@+id/slideViewPager"
        android:layout_alignParentEnd="true"
        android:layout_weight="1"
        android:text="Next" />

</RelativeLayout>
```

Activity_locked_Apps.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".LockedApps">


    <androidx.recyclerview.widget.RecyclerView
        android:id="@+id/lockedAppsListt"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent">

    </androidx.recyclerview.widget.RecyclerView>


</RelativeLayout>
```

Activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/main_activity"
    android:layout_width="match_parent"
```

```
    android:layout_height="match_parent"
    android:background="@color/black"

    tools:context=".MainActivity">

  <androidx.recyclerview.widget.RecyclerView
      android:id="@+id/lockedAppsListt"
      android:layout_width="match_parent"
      android:layout_height="match_parent"



     android:layout_marginTop="98dp"
      app:layout_constraintBottom_toBottomOf="parent"
      app:layout_constraintEnd_toEndOf="parent"
      app:layout_constraintStart_toStartOf="parent"
      app:layout_constraintTop_toTopOf="parent" />

  <LinearLayout
      android:id="@+id/emptyLockListInfo"
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:background="@color/grey"
      android:gravity="center"
      android:orientation="vertical">
```

Activity_pattern_lock.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".PatternLockAct">

    <TextView
        android:id="@+id/lock_screen"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="App locked"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

Activity_schedule.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@color/grey"
    android:orientation="vertical"
    android:paddingTop="60dp"
    tools:context=".Schedule">


    <ca.antonious.materialdaypicker.MaterialDayPicker
        android:id="@+id/day_picker"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_margin="20dp"
        android:orientation="vertical" />
```

```
    <LinearLayout
        android:layout_width="wrap_content"
        android:layout_height="102dp"
        android:layout_gravity="center"
        android:orientation="horizontal">




<Button
            android:id="@+id/fromTimeBtn"
            android:layout_width="match_parent"
            android:layout_height="70dp"
            android:layout_marginHorizontal="26dp"
            android:layout_marginTop="36dp"
            android:text="FROM"
            android:textSize="15dp"
            app:backgroundTint="@color/darkblue" />

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="center"
            android:layout_marginTop="18dp"
            android:text="To"
            android:textStyle="bold" />

        <Button
            android:id="@+id/untilTimeBtn"
            android:layout_width="match_parent"
            android:layout_height="70dp"
            android:layout_marginHorizontal="26dp"
            android:layout_marginTop="36dp"
            android:text="UNTIL"
            android:textSize="15dp"
            app:backgroundTint="@color/darkblue" />
    </LinearLayout>

    <LinearLayout
        android:layout_width="wrap_content"
        android:layout_height="112dp"
        android:orientation="horizontal">

        <Button
            android:id="@+id/cancelScheduleBtn"
            android:layout_width="155dp"
            android:layout_height="70dp"
            android:layout_marginHorizontal="26dp"
            android:layout_marginTop="26dp"
            android:backgroundTint="@color/alertRed"
            android:drawableLeft="@drawable/ic_baseline_delete_forever_24"
            android:text="Cancel"
            android:textSize="12dp" />

        <Button
            android:id="@+id/confirmScheduleBtn"
            android:layout_width="155dp"
            android:layout_height="70dp"
            android:layout_marginHorizontal="26dp"
            android:layout_marginTop="26dp"
            android:background="@color/black"
            android:backgroundTint="@color/green"
            android:drawableLeft="@drawable/ic_baseline_check_circle_24"
            android:text="Confirm"
            android:textSize="12dp" />
```

```
    </LinearLayout>


</LinearLayout>
```

Build.gradle

```
// Top-level build file where you can add configuration options common to all sub-projects/modules.
buildscript {
    repositories {
        google()
        jcenter()
    }
    dependencies {
        classpath 'com.android.tools.build:gradle:7.4.1'

        // NOTE: Do not place your application dependencies here; they belong
        // in the individual module build.gradle files
    }
}

allprojects {
    repositories {
        google()
        jcenter()
    }
}

task clean(type: Delete) {
    delete rootProject.buildDir
}
```

Build.gridle(Module:app)

```
plugins {
    id 'com.android.application'
}

android {
    compileSdkVersion 30
    buildToolsVersion "30.0.3"

    defaultConfig {
        applicationId "com.robocora.appblockr"
        minSdkVersion 19
        targetSdkVersion 30
        versionCode 1
        versionName "1.01 "
        testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"
    }

    buildFeatures {
        viewBinding true
    }

    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android-optimize.txt'), 'proguard-rules.pro'
        }
    }
    compileOptions {
        sourceCompatibility JavaVersion.VERSION_1_8
        targetCompatibility JavaVersion.VERSION_1_8
    }
}
```

```
dependencies {

    implementation 'androidx.appcompat:appcompat:1.2.0'
    implementation 'com.google.android.material:material:1.3.0'
    implementation 'androidx.constraintlayout:constraintlayout:2.0.4'
    implementation 'androidx.legacy:legacy-support-v4:1.0.0'
    implementation 'androidx.cardview:cardview:1.0.0'
    implementation ''androidx.cardview:cardview:1.0.0''
    implementation 'ca.antonious:materialdaypicker:0.7.4'
    implementation 'com.google.android.material:material:1.3.0'
    testImplementation 'junit:junit:4.+'
    androidTestImplementation 'androidx.test.ext:junit:1.1.2'
    androidTestImplementation 'androidx.test.espresso:espresso-core:3.3.0'
    implementation 'com.etebarian:meow-bottom-navigation-java:1.2.0'
    implementation 'io.github.medyo:android-about-page:2.0.0'
    implementation 'gr.pantrif:easy-android-splash-screen:0.0.1'


}
```

5.1 Screen Shots

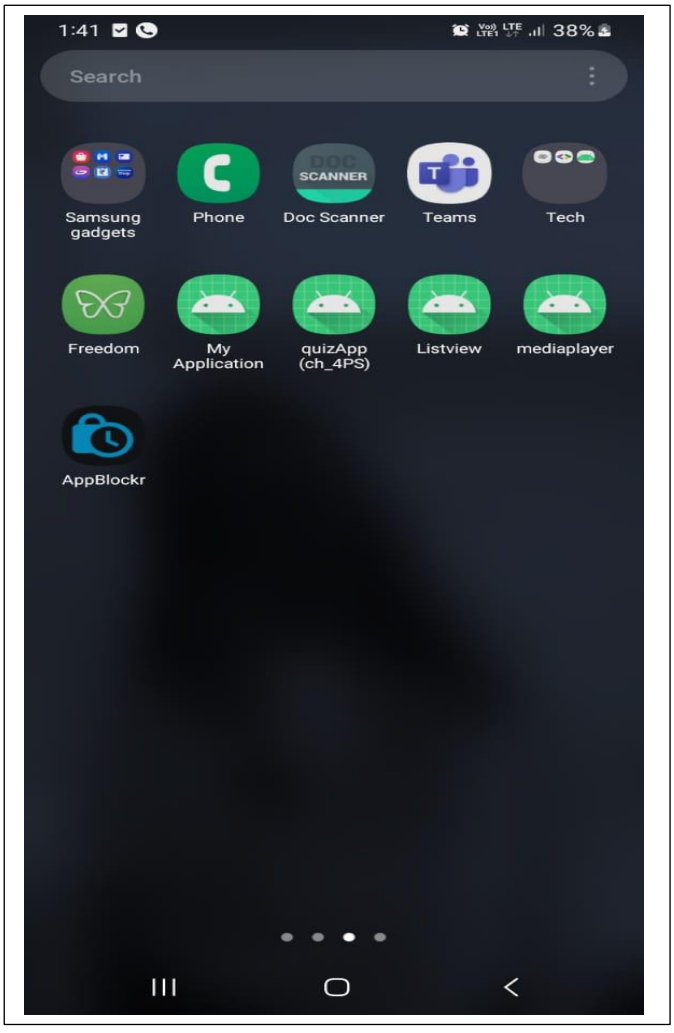

Fig 7: Logo

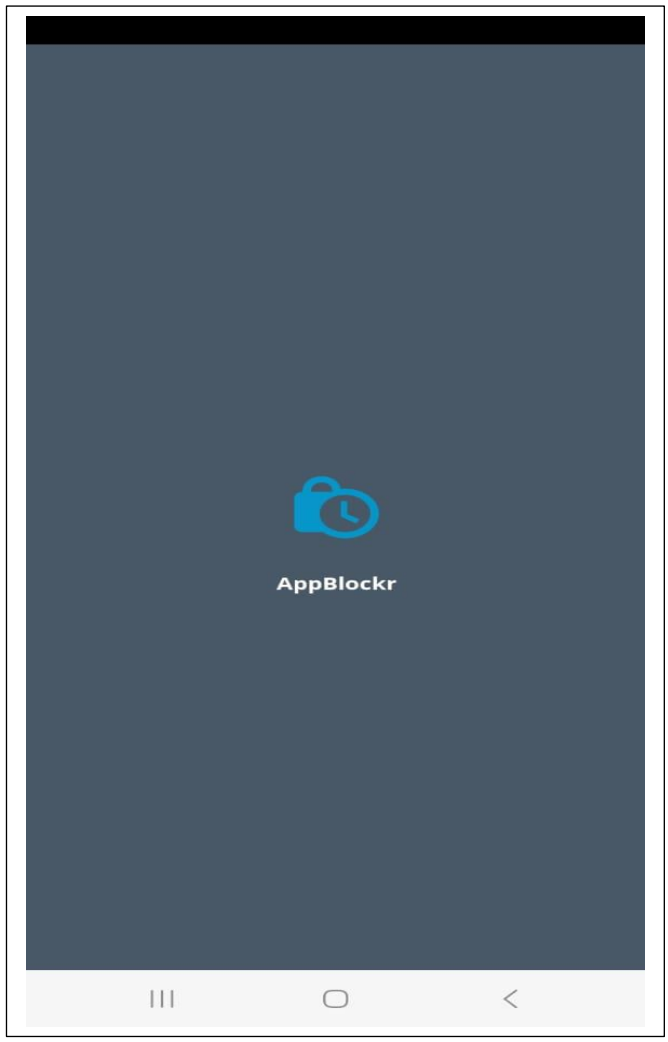

Fig 8: Front Page

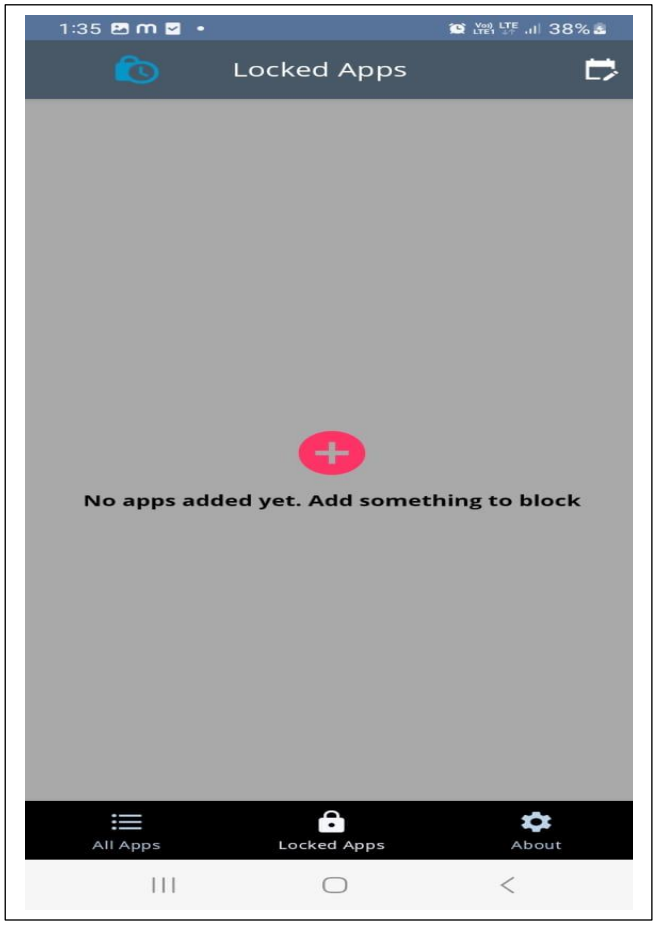

Fig 9: Add Apps

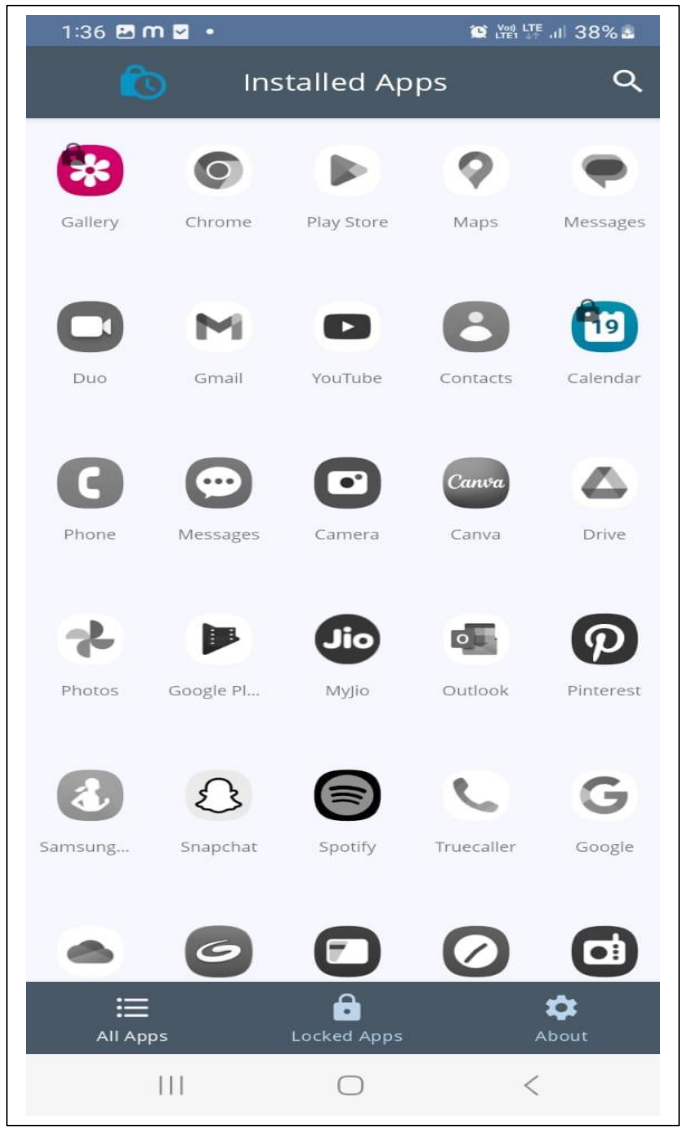

Fig 10: Select Apps

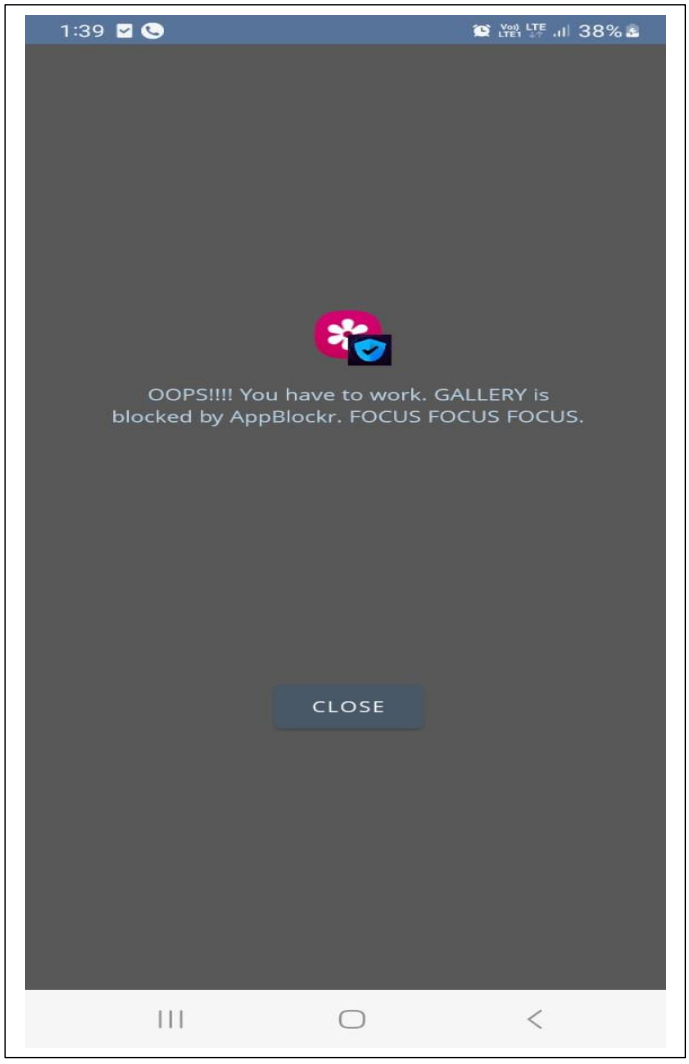

Fig 11: If open App

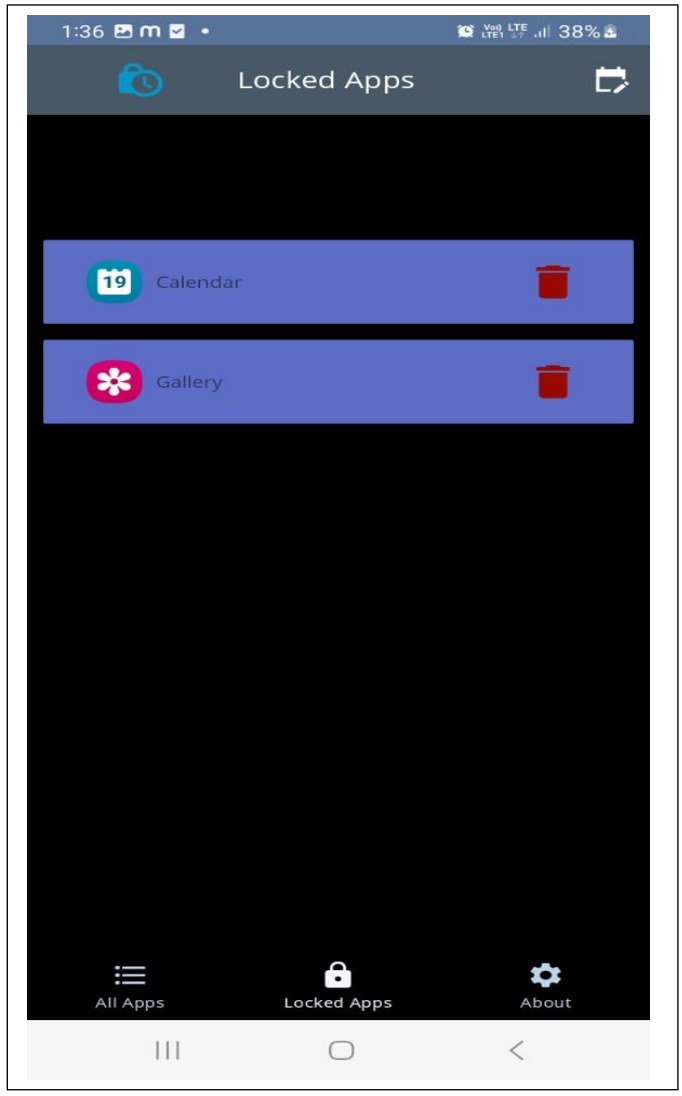

Fig 12: Selected App

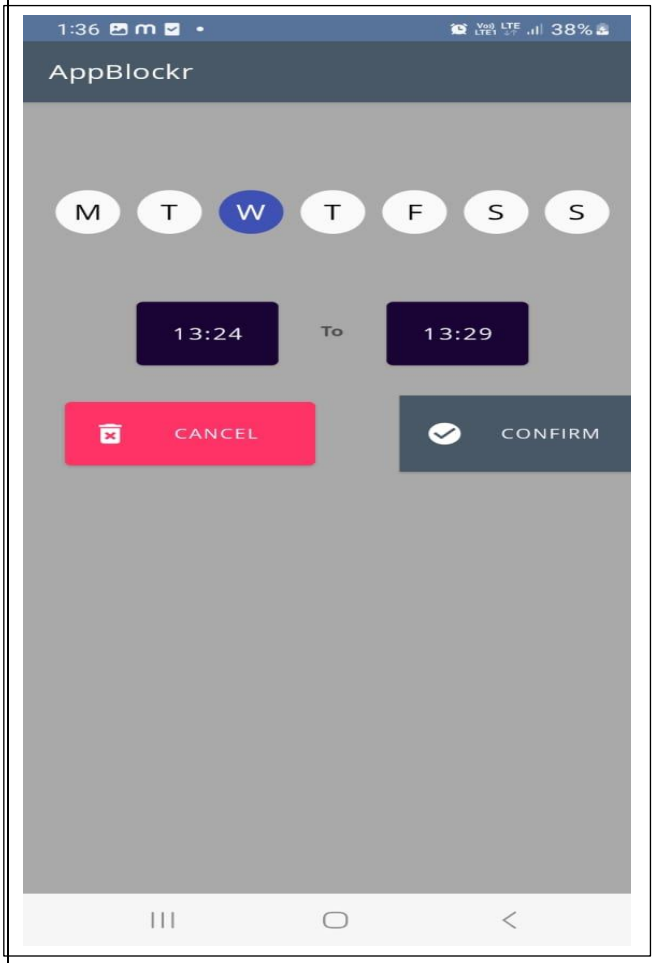

Fig 13: Set Time & Date

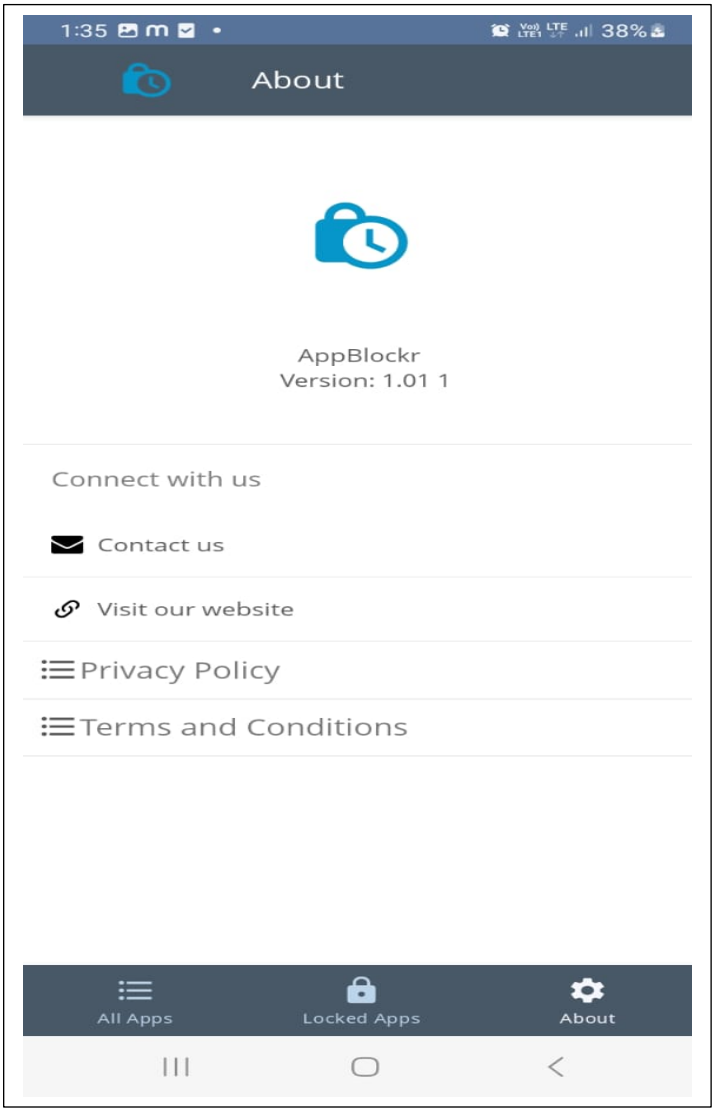

Fig 14: About

6.SOFTWARE TESTING

TEST CASES

Test Case 1: Verify App Blocker App Installation

Test Case Description: This test case verifies that the App Blocker App is installed successfully on the mobile device.

Test Steps:

- Open the mobile device's app store.
- Search for the App Blocker App.
- Verify that the App Blocker App is listed in the search results.
- Click on the App Blocker App listing.
- Verify that the app description, screenshots, and ratings are displayed.
- Click on the "Install" or "Download" button.
- Wait for the installation process to complete.
- Verify that the App Blocker App is successfully installed on the mobile device.

Expected Results:

- The App Blocker App is available for download in the app store.
- The app can be installed without any errors or issues.
- The app appears in the mobile device's app list after installation.

Test Case 2: Block Single app

Test Case Description: This test case verifies the functionality of the App Blocker App to block a single application on the mobile device.

Preconditions:

- The App Blocker App is installed on the mobile device.
- The mobile device has at least one target application to be blocked.

Test Steps:

- Launch the App Blocker App.
- Verify that the main screen of the App Blocker App is displayed.

- Tap on the ‘+’ icon on the top right side of the screen.
- Verify that the list of installed applications is displayed.
- Select one target application from the list.
- Tap on the "Block" button.
- Verify that a confirmation message is displayed, indicating that the selected application has been blocked.
- Launch the blocked application.
- Verify that a blocking screen is displayed, preventing access to the blocked application.
- Close the blocked application.
- Remove the application from the blocked list.
- Verify that the application is successfully removed from the blocked list.

Expected Results:

- The App Blocker App allows selecting and blocking a specific application.
- The blocked application is inaccessible and displays a blocking screen when launched.
- The blocked application becomes accessible again after being removed from the blocked list.

Test Case 3: Block Multiple Applications

Test Case Description: This test case verifies the functionality of the App Blocker App to block multiple applications on the mobile device.

Preconditions:

- The App Blocker App is installed on the mobile device.
- The mobile device has multiple target applications to be blocked.

Test Steps:

- Launch the App Blocker App.
- Verify that the main screen of the App Blocker App is displayed.
- Tap on the ‘+’ symbol shown at the center of the screen.
- Verify that the list of installed applications is displayed.
- Select multiple target applications from the list.
- Tap on the "Block" button.

- Verify that a confirmation message is displayed, indicating that the selected applications have been blocked.
- Launch each blocked application.
- Verify that a blocking screen is displayed for each blocked application, preventing access.
- Close each blocked application.
- Remove the applications from the blocked list.
- Verify that the applications are successfully removed from the blocked list.

Expected Results:

- The App Blocker App allows selecting and blocking multiple applications simultaneously.
- All blocked applications are inaccessible and display a blocking screen when launched.
- All blocked applications become accessible again after being removed from the blocked list.

Test Case 4 : Schedule Application Blocking

Test Case Description: This test case verifies the functionality of the App Blocker App to schedule application blocking at specific times.

Preconditions:

- The App Blocker App is installed on the mobile device.
- The mobile device has at least one target application to be blocked.

Test Steps:

- Launch the App Blocker App.
- Verify that the main screen of the App Blocker App is displayed.
- Tap on the calendar icon on the top right side of the screen.
- Set a specific time range for application blocking (e.g. : 9:00 - 12:00).
- Tap on the "Confirm" to save the schedule.
- Wait for the scheduled time range to start.
- Launch the blocked application within the scheduled time range.
- Verify that a blocking screen is displayed, preventing access to the blocked application and a message is popped up on the screen.
- Close the blocked application.

- Wait for the scheduled time range to end.
- Launch the blocked application after the scheduled time range.
- Verify that the application is accessible without any blocking screen.

Expected Results:

- The App Blocker App allows scheduling specific time ranges for application blocking.
- During the scheduled time range, launching the blocked application should display a blocking screen.
- Outside the scheduled time range, the blocked application should be accessible without any blocking screen.

7.CONCLUSION

In conclusion, an Android app blocker is a valuable tool for individuals who want to enhance their productivity, manage their time effectively, or reduce distractions. It provides the ability to block or limit access to specific applications on an Android device, allowing users to stay focused on important tasks, studies, or responsibilities.

By using an app blocker, users can customize their device usage according to their needs. They can choose which apps to block or set specific time limits for app usage, creating a healthier balance between work and leisure activities. This can be particularly beneficial for students, professionals, or anyone who struggles with excessive smartphone usage.

Additionally, an Android app blocker can also be used as a parental control tool. Parents can restrict access to certain apps or set usage limits for their children, ensuring they maintain a healthy digital lifestyle and prioritize other activities such as studying or spending time with family.

Overall, an Android app blocker empowers users to take control of their digital habits and focus on what truly matters to them. It promotes productivity, time management, and a healthy balance between technology and other aspects of life. However, it's important to note that while an app blocker can be a helpful tool, it is ultimately up to the user to practice self-discipline and make conscious decisions about their smartphone usage.

8.FUTURE ENHANCEMENT

There are a few potential areas for future improvements and extra capabilities as we work on the Android app for blocking other apps. Take into account the following ideas for further development iterations:

- App Usage Statistics: Offer users comprehensive usage data for both blocked and unblocked apps, including information on how long users spent using each app and how frequently they launched it. Users can use this information to better understand their app usage habits and decide whether or not to restrict certain apps.

- Advanced Blocking Rules: Apply more complex blocking regulations in accordance with user choices. For instance, let users create rules to ban applications during particular times or activities (like during study hours, bedtime, or while driving), or let users block apps based on specified terms in the app titles or descriptions.

- Remote Management: Include a feature that enables users to block or unblock apps on their device from a different device or via a web-based interface. Parents who wish to monitor app usage on their child's smartphone or users who want to remotely modify their app blocking settings can both benefit from this functionality.

- Blacklisting and Whitelisting: Give people the option to make their own blacklists and whitelists. Blacklists can be used to restrict specific apps regardless of other blocking settings, while whitelists can be used to select a range of apps that are always allowed.

- Blocking based on Geo location can be accomplished by integrating Geo location services. When users wish to automatically prohibit particular apps when they are in particular settings (like the workplace or school), this can be helpful.

- Blocking of social media apps and websites should be added to the capabilities of app blocking. Users who desire to increase productivity or cut back on their social media use may find this tool useful.

- Support for numerous user profiles on a single device should be implemented, enabling various users to have their own preferences and app banning settings.

- App recommendations: Offer customers customized app suggestions based on their app usage and preference habits. Users can utilize this tool to find new apps that fit their interests while promoting safe app usage.

- Integration with Digital Well-being: To improve the overall well-being and balance of app usage for consumers, integrate the app with Android's Digital Well-being features or other comparable platforms.
- Implement a system for backing up preferences and settings for app blocking to the cloud so that users can restore them or sync them across other devices.

BIBLIOGRAPHY

1. "AppLock: Android Application for Blocking Unwanted Applications," 2017 International Conference on Data Management, Analytics and Innovation (ICDMAI), Pune, India, pp. 1-4. Authors R. Pandey and S. Joshi.
2. "Android Application for App Locking with Multiple Authentication Techniques," 2017 International Conference on Intelligent Sustainable Systems (ICISS), Palladam, India, S. Rastogi, N. Sengar, V. Mittal, and R. Bansal, pp. 525-529.
3. "Android Application for App Locking and Parental Control," International Conference on Power, Signals, Control and Computation (EPSCICON), Kannur, India, 2018, pp. 1–5. N. Tyagi, P. Tyagi, and A. Kumar.
4. "Design and Implementation of App-locker for Android Mobiles," 2016 International Conference on Information Processing (ICIP), Singapore, pp. 1–5, by S. Gupta, D. Patel, R. Shastri, and A. Shastri.
5. "App Locking Mechanism for Android Smartphones," 2016 International Conference on Computing, Analytics and Security Trends (CAST), Chennai, India, pp. 79–83. J. Kaur and A. Kaur.
6. "AppLocker: Application for Android Smartphone," 2018 IEEE 8th International Conference on Advanced Computing (IACC), Jaipur, India, pp. 610-615. L. K. Patel and P. Chauhan

   .

7. "Design and Implementation of an App Locker for Android Mobiles," 2016 International Conference on Inventive Computation Technologies (ICICT), Coimbatore, India, pp. 1–5. Authors A. Kaur and G. Kaur.
8. "Android Application Permission Analysis and Classification," 2019 International Conference on Artificial Intelligence and Big Data (ICAIBD), Chongqing, China, pp. 295-298. C. B. Zhang, H. Wang, C. Hu, and Q. M. Zhu.
9. J. Wang, W. Huang, and Z. Li, "Design and Implementation of AppLocker System Based on Android," 5th International Conference on Systems and Informatics (ICSAI), Nanjing, China, 2018, pp. 1-4.