

**VISVESVARAYA TECHNOLOGICAL UNIVERSITY**

**BELGAVI, KARNATAKA -590 018**

**A Minor Project Report on**

**“NOTEVERSE APP”**

***Submitted in partial fulfillment for the Mobile Application Development [18CSMP68] course of Sixth Semester of Bachelor of Engineering in Computer Science & Engineering during the academic year 2022-23.***

**By**

**Drupad S 4MN20CS015 Syed Zeeshan 4MN20CS050**

**|| Under the Guidance of ||**

**Dr. Ranjit K N**

**Associate Professor Dept. of CS&E MIT Thandavapura**



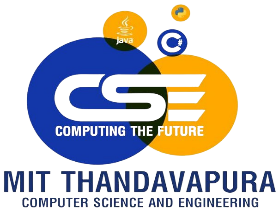
**2021-22**

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**

### MAHARAJA INSTITUTE OF TECHNOLOGY THANDAVAPURA

**NH 766, Nanjangud Taluk, Mysuru – 571302**

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING MAHARAJA INSTITUTE OF TECHNOLOGY THANDAVAPURA**



# ~~ CERTIFICATE ~~

*Certified that the minor project work entitled* ***“NoteVerse App”*** *is a bonafide work carried out by* ***Syed Zeeshan****(4MN20CS050) &* ***Drupad S****(4MN20CS015) for the course* ***Mobile Application Development*** *with course code* ***18CSMP68*** *of Sixth Semester in Computer Science & Engineering under Visvesvaraya Technological University, Belagavi during academic year* ***2022-23****.*

*It is certified that all corrections/suggestions indicated for Internal Assignment have been incorporated in the report. The report has been approved as it satisfies the course requirements.*

\_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ Signature of Lab Staff In-Charge

**Dr. Ranjit K N**

Associate Professor Dept. of CS&E MIT Thandavapura

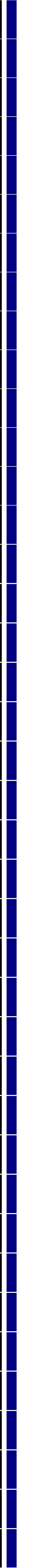
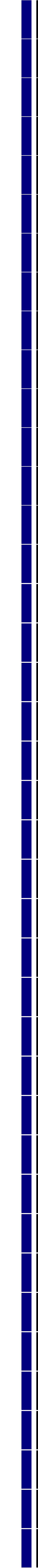
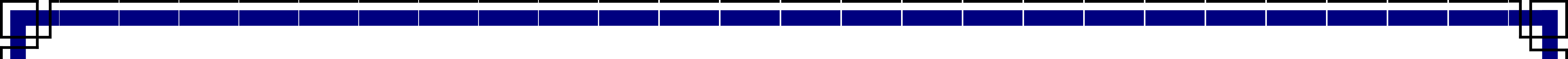
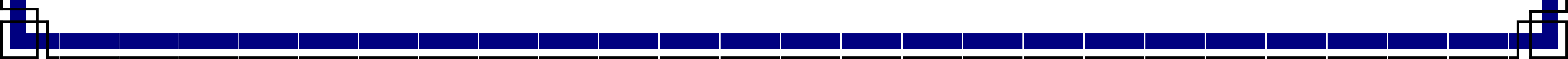
\_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_

Signature of the HoD

**Dr. Ranjit K N**

Associate Professor & Head Dept. of CS&E

MIT Thandavapura



**External viva**

**Name of the Examiners Signature with date**

1)…………………………………………………………………………………….

2)…………………………………………………………………………………….

# ~~~~ ACKNOWLEDGEMENT ~~~~

It is the time to acknowledge all those who have extended their guidance, inspiration and their whole hearted co-operation all along our project work.

We are grateful to **Dr. Y T Krishne Gowda**, Principal, MIT Thandavapura, **Dr. H K Chethan,** Professor and Mentor, CS&E, MIT Thandavapura and also **Dr. Ranjit K N**, Associate Professor and Head, CS&E, MIT Thandavapura for having provided us academic environment which nurtured our practical skills contributing to the success of our project.

We would like to sincerely thank our project guide **Dr. Ranjit K N**, Associate Professor, Dept. of CS&E, MIT Thandavapura for providing relevant information, valuable guidance and encouragement to complete this project.

We wish to place a deep sense of gratitude to all Teaching and Non-Teaching staffs of Computer Science and Engineering Department for whole-hearted guidance and constant support without which this endeavor would not have been possible.

Our gratitude will not be complete without thanking our parents and also our friends, who have been a constant source of support and aspirations.

## Name Signature

**Syed Zeeshan**

**Drupad S**

# ~~~ ABSTRACT ~~~

In this project, the development of the user-friendly application exploited using the Android platform. Android is an open-source software stack created for mobile phones and other devices and is built on top of Linux kernel and GNU software. The software stack of the Android runs Java applications using Java core libraries. Each instance of Java application runs on its own Virtual Machine called Dalvik . Android is freely available to manufacturers for customization, there are no fixed hardware and software configurations

. However, Android itself supports features. The following listed are some of them which used in the project Uses SQLite, a lightweight relational database, for data storage. SQLite has been used to save the data read and images used by the application development Supports GSM/EDGE ,Global System for Mobile Communications - Enhanced Data rates for GSM Evolution ,IDEN,CDMA,EV-DO,UMTS, Bluetooth, LTE, WiMAX and WiFi which have be used for this application.

# ~~~~~ CONTENTS ~~~~~

|  |  |  |  |
| --- | --- | --- | --- |
| **SL. No.** | | **Index** | **Page No.** |
| **1** |  | **INTRODUCTION** | **1 - 2** |
|  | 1.1 | Existing Project | 1 |
|  | 1.2 | Overview of the Project | 2 |
| **2** |  | **DESIGN AND IMPLEMENTATION** | **3 - 4** |
|  | 2.1 | Functional Requirements | 3 |
|  | 2.2 | User Requirements | 3 |
|  | 2.3 | Software Requirements | 3 |
|  | 2.4 | Implementation | 4 |
| **3** |  | **RESULT ANALYSIS** | **5** |
|  | 3.1 | Snapshots | 5 |
| **4** |  | **CONCLUSION AND FUTURE WORK** | **6** |
|  | 4.1 | Conclusion | 6 |
|  | 4.2 | Future Enhancement | 6 |
|  |  | **REFERENCES** | **7** |
|  |  | **APPENDIX A – Source Code** | **A1 – A4** |

**~~~~~ LIST OF FIGURES ~~~~~**

|  |  |  |
| --- | --- | --- |
| **SL. No.** | **Index** | **Page No.** |
| **3.1** | Home screen without information | **5** |
| **3.2** | Screen Displaying Information | **5** |

**CHAPTER – 1**

# INTRODUCTION

Noteverse is an Android-based note-taking application designed to streamline the process of capturing and organizing your notes. With its user-friendly interface and a range of useful features, Noteverse offers a convenient solution for keeping track of your thoughts, ideas, and important information.

The app allows you to create text-based notes, complete with titles and descriptions, providing a structured approach to note-taking. Additionally, Noteverse supports multimedia attachments, enabling you to add images, audio recordings, and sketches to your notes, enhancing the versatility and visual appeal of your content.To keep your notes well-organized, Noteverse offers customizable categories, allowing you to create a personalized organizational system that suits your needs. You can also assign tags to your notes for further classification and easy searching based on specific keywords or topics.

Noteverse provides several tools to enhance your productivity and workflow. You can set reminders for important tasks or deadlines, ensuring that you stay on top of your commitments. The app also offers rich text editing capabilities, enabling you to format your notes with different styles, colors, and fonts.

Data security and accessibility are prioritized in Noteverse. You have the option to create local backups on your device, ensuring that your notes are protected in case of any unexpected events. Furthermore, Noteverse supports synchronization with popular cloud storage services such as Google Drive and Dropbox, allowing you to access your notes from multiple devices and ensuring seamless data continuity.

* 1. **Existing System**

The existing system of conventional note-making apps may have certain disadvantages when compared to the Noteverse app.

Limited Organization: Conventional note-making apps often provide limited options for organizing notes. They may lack customizable categories or tagging features, making it challenging to categorize and search for specific notes effectively. In contrast, Noteverse offers customizable categories and tagging options, allowing users to create a personalized organizational system for efficient note management

Lack of Multimedia Support: Some traditional note-taking apps may not support multimedia attachments such as images, audio recordings, or sketches. This limitation restricts the ability to capture and present information in a versatile and visually appealing manner. Noteverse, on the other hand, supports multimedia content, enabling users to include various media types within their notes for enhanced creativity and communication.

Limited Customization: Conventional note-making apps may offer limited customization options, such as limited font choices or themes. This can result in a monotonous note-taking experience that does not align with individual preferences. Noteverse addresses this by providing customization choices including themes, fonts, and sorting preferences, allowing users to personalize the app to suit their aesthetic and functional preferences.

Inadequate Synchronization and Backup: Some traditional note-taking apps may lack robust synchronization and backup options. This can result in data loss or limited accessibility when switching devices or encountering technical issues. Noteverse emphasizes data backup and synchronization, offering the flexibility to store notes locally or sync them with cloud storage services, ensuring data security and providing seamless access across devices.

* 1. **Proposed System**

The proposed system of the Noteverse app offers several advantages over traditional note-taking apps.

Enhanced Organization: The Noteverse app provides customizable categories and tagging options, allowing users to create a personalized organizational system. This feature makes it easier to categorize and search for specific notes, improving overall note management and productivity**.**

Multimedia Support: Unlike traditional note-taking apps, Noteverse supports multimedia attachments such as images, audio recordings, and sketches. This capability enables users to capture and incorporate various types of media within their notes, enhancing creativity and facilitating better communication

Customization Options: Noteverse offers a range of customization choices including themes, fonts, and sorting preferences. This allows users to personalize the app's appearance and functionality according to their individual preferences, creating a more enjoyable and tailored note-taking experience.

Synchronization and Backup: Noteverse emphasizes data synchronization and backup options. Users can store their notes locally or sync them with cloud storage services like Google Drive or Dropbox. This ensures that notes are securely backed up and accessible across multiple devices, providing convenience and peace of mind.

Synchronization and Backup: Noteverse emphasizes data synchronization and backup options. Users can store their notes locally or sync them with cloud storage services like Google Drive or Dropbox. This ensures that notes are securely backed up and accessible across multiple devices, providing convenience and peace of mind.

**CHAPTER – 2**

# DESIGN AND IMPLEMENTATION

## Functional Requirements

The functional requirements are the statement of services the system should provide, how system reacts to particular inputs and how system should behave in particular situation. It describes the functionality that the system provides.

Our app requires:

* The user should have the appropriate version of windows.
* The application should be installed on the system.
* Active internet connection.

## User Requirements

User requires active internet connection to use the app.

## Software Requirements

* Operating System: Windows 10
* Android SDK
* Android Studio

## A. Android SDK

The Android SDK provides you the API libraries and developer tools necessary to build, test, and debug apps for Android. The ADT bundle includes the essential Android SDK components and a version of the Eclipse IDE with built-in Android Developer Tools to streamline the Android app development. ADT bundle consists of following components for developing the application II. Eclipse ADT plugin.

* + Android SDK Tools
  + Android Platform-tools
  + The latest Android platform
  + The latest Android system image for the emulator

## B. Android Studio

Android is a mobile operating system based on a modified version of the Linux kernel and other open-source software, designed primarily for touch screen mobile devices

such as smart phones and tablets. Android is developed by a consortium of developers known as the Open Handset Alliance, with the main contributor and commercial marketer being Google. Initially developed by Android Inc., which Google bought in 2005, Android was unveiled in 2007, with the first commercial Android device launched in September 2008. The current stable version is Android 11, released on September 8, 2020.

## Implementation

We are here designing an app to digitize the process of note taking. The user can obtain both light mode and dark mode.

## Solution Approach

We are here using xml and java for the front end and SQLite database for the backend as a server.

## Storage

The files like images, audio, video etc can be stored in the app. The data stored is highly secured and is robust in nature means it resumes from the last point if any network error occurs.

## Database and Android App

Saving data to a database is ideal for repeating or structured data, such as contact information. If you are familiar with SQL databases in general, it helps you get started with SQLite databases on Android. SQLite is very good for testing. It doesn’t need any complex setup to store the data. When you build native applications with java, it comes integrated with platform

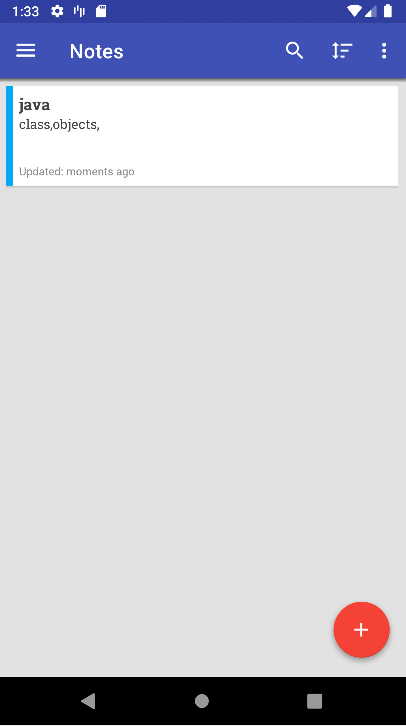
## Java

There are several ways to create apps for Android devices, but the recommended method for most developers is to write native apps using Java and the Android SDK. Java for Android apps is both similar and quite different from other types of Java applications. If you have experience with Java (or a similar language) then you’ll probably feel comfortable diving right into the code and learning how to use the Android SDK to make your app run.

**CHAPTER – 3**

# 3.1 Snapshots

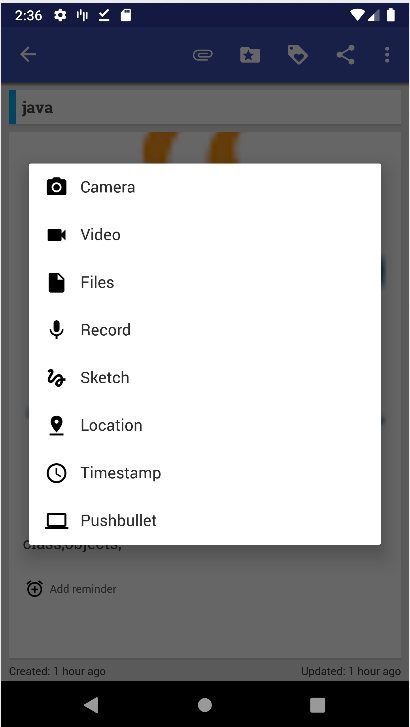
**RESULT ANALYSIS**

****

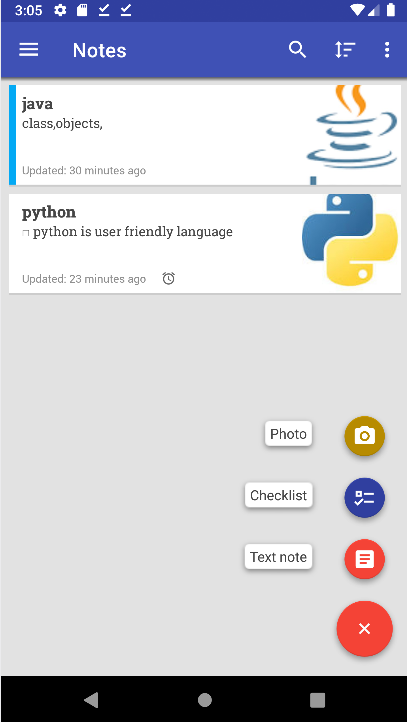
**Fig 3.1**: Home Screen Interface



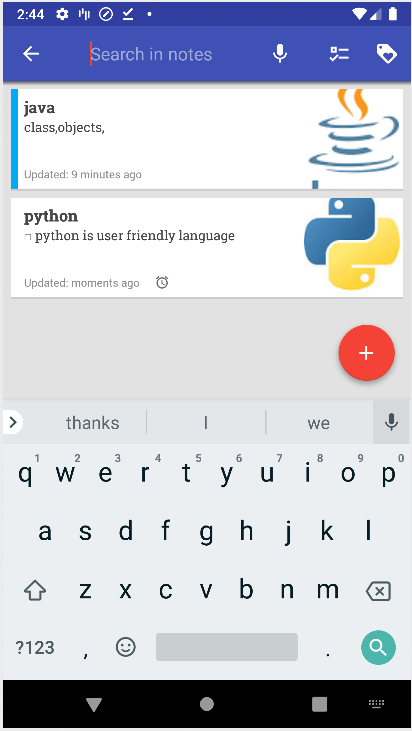
**Fig 3.2**: Displaying Contents of the notes



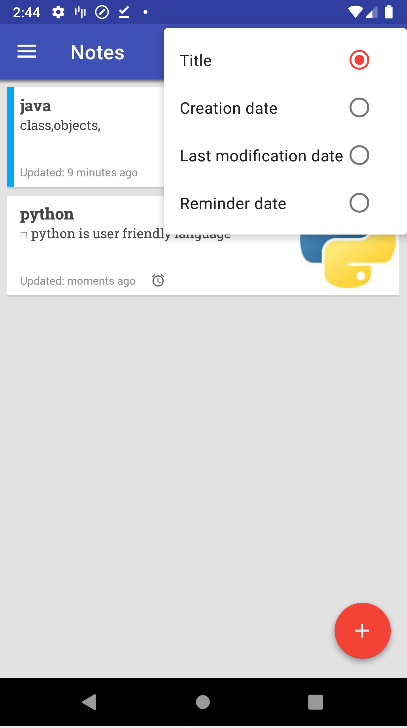
**Fig 3.3**: Adding Multimedia to the Content



**Fig 3.4**: Multimedia Support



**Fig 3.5**: Content Search Feature



**Fig 3.6**: Organizing Features

**CHAPTER – 4**

# CONCLUSION AND FUTURE WORK

* 1. **Conclusion**

In conclusion, Noteverse is an Android-based note-taking app that provides a user-friendly and efficient solution for capturing, organizing, and managing notes. With its customizable organization system, support for multimedia attachments, and robust synchronization options, Noteverse offers a comprehensive set of features to enhance the note-taking experience. The app's focus on simplicity, customization, and data security makes it a valuable tool for users who want to stay organized and productive.

## Future Work

While Noteverse already offers a range of useful features, there are several areas where future enhancements could further improve the app. Introducing collaboration features would allow users to share and work on notes with others, enabling seamless teamwork and enhancing productivity in group settings. Incorporating artificial intelligence (AI) capabilities could enhance the app's functionality. For example, AI-based text recognition could enable users to search for specific words or phrases within their handwritten notes or scanned documents. Developing web and desktop versions of Noteverse would allow users to access and manage their notes across multiple platforms, providing seamless synchronization and a consistent user experience.

# REFERENCES

1. Varsha Chavan, Priya Jadhav, Snehal Korade, Priyanka Teli, “Implementing Customizable Online Food Ordering System Using Web Based Application”,

International.

1. Ashutosh Bhargave, Niranjan Jadhav, Apurva Joshi, PrachiOke, S.R. Lahane, “Digital

Customizable notes app Using Android”, International Journal of Scientific and Research Publications 201

# APPENDIX – A

# SOURCE CODE

## JAVA CODE

import static it.feio.android.omninotes.OmniNotes.isDebugBuild;

importstaticit.feio.android.omninotes.helpers.AppVersionHelper.isAppUpdated;

importstaticit.feio.android.omninotes.helpers.AppVersionHelper.updateAppVersionInPreferences;

importstaticit.feio.android.omninotes.helpers.ChangelogHelper.showChangelog;

importstaticit.feio.android.omninotes.utils.ConstantsBase.ACTION\_NOTIFICATION\_CLICK;

importstaticit.feio.android.omninotes.utils.ConstantsBase.ACTION\_RESTART\_APP; import static it.feio.android.omninotes.utils.ConstantsBase.ACTION\_SEND\_AND\_EXIT; import static it.feio.android.omninotes.utils.ConstantsBase.ACTION\_SHORTCUT; import static it.feio.android.omninotes.utils.ConstantsBase.ACTION\_SHORTCUT\_WIDGET; import static it.feio.android.omninotes.utils.ConstantsBase.ACTION\_START\_APP; import static it.feio.android.omninotes.utils.ConstantsBase.ACTION\_WIDGET; import static it.feio.android.omninotes.utils.ConstantsBase.ACTION\_WIDGET\_TAKE\_PHOTO; import static it.feio.android.omninotes.utils.ConstantsBase.INTENT\_GOOGLE\_NOW;

import static it.feio.android.omninotes.utils.ConstantsBase.INTENT\_KEY; import static it.feio.android.omninotes.utils.ConstantsBase.INTENT\_NOTE; import staticit.feio.android.omninotes.utils.ConstantsBase.PREF\_PASSWORD;

**import it.feio.android.omninotes.helpers.LogDelegate;**

**import it.feio.android.omninotes.helpers.NotesHelper;**

**import it.feio.android.omninotes.intro.IntroActivity;**

**import it.feio.android.omninotes.models.Attachment;**

**import it.feio.android.omninotes.models.Category;**

**import it.feio.android.omninotes.models.Note;**

**import it.feio.android.omninotes.models.ONStyle;**

**import it.feio.android.omninotes.utils.FileProviderHelper;**

**import it.feio.android.omninotes.utils.PasswordHelper;**

**import it.feio.android.omninotes.utils.SystemHelper;**

**import it.feio.android.pixlui.links.UrlCompleter;**

**import java.io.FileNotFoundException;**

**import java.util.ArrayList;**

**import java.util.Collections;**

**import java.util.HashMap;**

**import lombok.Getter;**

**import lombok.Setter;**

**public class MainActivity extends BaseActivity implements**

**SharedPreferences.OnSharedPreferenceChangeListener {**

**private boolean isPasswordAccepted = false;**

**public static final String FRAGMENT\_DRAWER\_TAG = "fragment\_drawer";**

**public static final String FRAGMENT\_LIST\_TAG = "fragment\_list";**

**public static final String FRAGMENT\_DETAIL\_TAG = "fragment\_detail";**

**public static final String FRAGMENT\_SKETCH\_TAG = "fragment\_sketch";**

**@Getter @Setter**

**private Uri sketchUri;**

**boolean prefsChanged = false;**

**private FragmentManager mFragmentManager;**

**ActivityMainBinding binding;**

**@Override**

**protected void onCreate(Bundle savedInstanceState) {**

**super.onCreate(savedInstanceState);**

**setTheme(R.style.OmniNotesTheme\_ApiSpec);**

**binding = ActivityMainBinding.inflate(getLayoutInflater());**

**View view = binding.getRoot();**

**setContentView(view);**

**EventBus.getDefault().register(this);**

**Prefs.getPreferences().registerOnSharedPreferenceChangeListener(this);**

**initUI();**

**}**

**@Override**

**protected void onPostCreate(@Nullable Bundle savedInstanceState) {**

**super.onPostCreate(savedInstanceState);**

**if (!launchIntroIfRequired() && isAppUpdated(getApplicationContext()) && !isDebugBuild()) {**

**showChangelogAndUpdateCurrentVersion();**

**}**

**}**

**private void showChangelogAndUpdateCurrentVersion() {**

**showChangelog(this);**

**updateAppVersionInPreferences(getApplicationContext());**

**}**

**private boolean launchIntroIfRequired() {**

**if (IntroActivity.mustRun()) {**

**startActivity(new Intent(getApplicationContext(), IntroActivity.class));**

**return true;**

**}**

**return false;**

**}**

**@Override**

**protected void onResume() {**

**super.onResume();**

**if (isPasswordAccepted) {**

**init();**

**} else {**

**checkPassword();**

**}**

**}**

**@Override**

**protected void onStop() {**

**super.onStop();**

**EventBus.getDefault().unregister(this);**

**}**

**private void initUI() {**

**setSupportActionBar(binding.toolbar.toolbar);**

**getSupportActionBar().setDisplayHomeAsUpEnabled(true);**

**getSupportActionBar().setHomeButtonEnabled(true);**

**}**

**/\*\***

**\* This method starts the bootstrap chain.**

**\*/**

**private void checkPassword() {**

**if (Prefs.getString(PREF\_PASSWORD, null) != null**

**&& Prefs.getBoolean("settings\_password\_access", false)) {**

**PasswordHelper.requestPassword(this**, passwordConfirmed -> {

switch (passwordConfirmed) {

case SUCCEED:

init();

break;

case FAIL:

finish();

break;

case RESTORE:

PasswordHelper.resetPassword(this);

}

});

} else {

init();

}

}

public void onEvent(PasswordRemovedEvent passwordRemovedEvent) {

showMessage(R.string.password\_successfully\_removed, ONStyle.ALERT);

init();

}

private void init() {

isPasswordAccepted = true;

getFragmentManagerInstance();

NavigationDrawerFragment mNavigationDrawerFragment = (NavigationDrawerFragment) getFragmentManagerInstance()

.findFragmentById(R.id.navigation\_drawer);

if (mNavigationDrawerFragment == null) {

FragmentTransaction fragmentTransaction = getFragmentManagerInstance().beginTransaction();

fragmentTransaction.replace(R.id.navigation\_drawer, new NavigationDrawerFragment(),

FRAGMENT\_DRAWER\_TAG).commit();

}

if (getFragmentManagerInstance().findFragmentByTag(FRAGMENT\_LIST\_TAG) == null) {

FragmentTransaction fragmentTransaction = getFragmentManagerInstance().beginTransaction();

fragmentTransaction.add(R.id.fragment\_container, new ListFragment(), FRAGMENT\_LIST\_TAG)

.commit();

}

handleIntents();

}

private FragmentManager getFragmentManagerInstance() {

if (mFragmentManager == null) {

mFragmentManager = getSupportFragmentManager();

}

return mFragmentManager;

}

@Override

protected void onNewIntent(Intent intent) {

if (intent.getAction() == null) {

intent.setAction(ACTION\_START\_APP);

}

super.onNewIntent(intent);

setIntent(intent);

handleIntents();

LogDelegate.d("onNewIntent");

}

public MenuItem getSearchMenuItem() {

Fragment f = checkFragmentInstance(R.id.fragment\_container, ListFragment.class);

if (f != null) {

return ((ListFragment) f).getSearchMenuItem();

} else {

return null;

}

}

public void editTag(Category tag) {

Fragment f = checkFragmentInstance(R.id.fragment\_container, ListFragment.class);

if (f != null) {

((ListFragment) f).editCategory(tag);

}

}

public void initNotesList(Intent intent) {

if (intent != null) {

Fragment searchTagFragment = startSearchView();

new Handler(getMainLooper()).post(() -> ((ListFragment) searchTagFragment).initNotesList(intent));

}

}

public Fragment startSearchView() {

FragmentTransaction transaction = getFragmentManagerInstance().beginTransaction();

animateTransition(transaction, TRANSITION\_HORIZONTAL);

ListFragment mListFragment = new ListFragment();

transaction.replace(R.id.fragment\_container, mListFragment, FRAGMENT\_LIST\_TAG).addToBackStack

(FRAGMENT\_DETAIL\_TAG).commit();

Bundle args = new Bundle();

args.putBoolean("setSearchFocus", true);

mListFragment.setArguments(args);

return mListFragment;

}

public void commitPending() {

Fragment f = checkFragmentInstance(R.id.fragment\_container, ListFragment.class);

if (f != null) {

((ListFragment) f).commitPending();

}

}

/\*\*

\* Checks if allocated fragment is of the required type and then returns it or returns null

\*/

private Fragment checkFragmentInstance(int id, Object instanceClass) {

Fragment result = null;

Fragment fragment = getFragmentManagerInstance().findFragmentById(id);

if (fragment != null && instanceClass.equals(fragment.getClass())) {

result = fragment;

}

return result;

}

@Override

public void onBackPressed() {

// SketchFragment

Fragment f = checkFragmentInstance(R.id.fragment\_container, SketchFragment.class);

if (f != null) {

((SketchFragment) f).save();

// Removes forced portrait orientation for this fragment

setRequestedOrientation(

ActivityInfo.SCREEN\_ORIENTATION\_UNSPECIFIED);

getFragmentManagerInstance().popBackStack();

return;

}

// DetailFragment

f = checkFragmentInstance(R.id.fragment\_container, DetailFragment.class);

if (f != null) {

((DetailFragment) f).goBack = true;

((DetailFragment) f).saveAndExit((DetailFragment) f);

return;

}

// ListFragment

f = checkFragmentInstance(R.id.fragment\_container, ListFragment.class);

if (f != null) {

// Before exiting from app the navigation drawer is opened

if (Prefs.getBoolean("settings\_navdrawer\_on\_exit", false) && getDrawerLayout() != null &&

!getDrawerLayout().isDrawerOpen(GravityCompat.START)) {

getDrawerLayout().openDrawer(GravityCompat.START);

} else if (!Prefs.getBoolean("settings\_navdrawer\_on\_exit", false) && getDrawerLayout() != null

&&

getDrawerLayout().isDrawerOpen(GravityCompat.START)) {

getDrawerLayout().closeDrawer(GravityCompat.START);

} else {

if (!((ListFragment) f).closeFab()) {

isPasswordAccepted = false;

super.onBackPressed();

}

}

return;

}

super.onBackPressed();

}

@Override

public void onSaveInstanceState(Bundle outState) {

super.onSaveInstanceState(outState);

outState.putString("navigationTmp", navigationTmp);

}

@Override

protected void onPause() {

super.onPause();

Crouton.cancelAllCroutons();

}

` public DrawerLayout getDrawerLayout() {

return binding.drawerLayout;

}

public ActionBarDrawerToggle getDrawerToggle() {

if (getFragmentManagerInstance().findFragmentById(R.id.navigation\_drawer) != null) {

return ((NavigationDrawerFragment) getFragmentManagerInstance().findFragmentById(

R.id.navigation\_drawer)).mDrawerToggle;

} else {

return null;

}

}

/\*\*

\* Finishes multiselection mode started by ListFragment

\*/

public void finishActionMode() {

ListFragment fragment = (ListFragment) getFragmentManagerInstance()

.findFragmentByTag(FRAGMENT\_LIST\_TAG);

if (fragment != null) {

fragment.finishActionMode();

}

}

Toolbar getToolbar() {

return binding.toolbar.toolbar;

}

private void handleIntents() {

Intent i = getIntent();

if (i.getAction() == null) {

return;

}

if (ACTION\_RESTART\_APP.equals(i.getAction())) {

SystemHelper.restartApp();

}

if (receivedIntent(i)) {

Note note = i.getParcelableExtra(INTENT\_NOTE);

if (note == null) {

note = DbHelper.getInstance().getNote(i.getIntExtra(INTENT\_KEY, 0));

}

// Checks if the same note is already opened to avoid to open again

if (note != null && noteAlreadyOpened(note)) {

return;

}

// Empty note instantiation

if (note == null) {

note = new Note();

}

switchToDetail(note);

return;

}

if (ACTION\_SEND\_AND\_EXIT.equals(i.getAction())) {

saveAndExit(i);

return;

}

// Tag search

if (Intent.ACTION\_VIEW.equals(i.getAction()) && i.getDataString()

.startsWith(UrlCompleter.HASHTAG\_SCHEME)) {

switchToList();

return;

}

// Home launcher shortcut widget

if (Intent.ACTION\_VIEW.equals(i.getAction()) && i.getData() != null) {

Long id = Long.valueOf(Uri.parse(i.getDataString()).getQueryParameter("id"));

Note note = DbHelper.getInstance().getNote(id);

if (note == null) {

showMessage(R.string.note\_doesnt\_exist, ONStyle.ALERT);

return;

}

switchToDetail(note);

return;

}

// Home launcher "new note" shortcut widget

if (ACTION\_SHORTCUT\_WIDGET.equals(i.getAction())) {

switchToDetail(new Note());

return;

}

}

/\*\*

\* Used to perform a quick text-only note saving (eg. Tasker+Pushbullet)

\*/

private void saveAndExit(Intent i) {

Note note = new Note();

note.setTitle(i.getStringExtra(Intent.EXTRA\_SUBJECT));

note.setContent(i.getStringExtra(Intent.EXTRA\_TEXT));

DbHelper.getInstance().updateNote(note, true);

showToast(getString(R.string.note\_updated), Toast.LENGTH\_SHORT);

finish();

}

private boolean receivedIntent(Intent i) {

return ACTION\_SHORTCUT.equals(i.getAction())

|| ACTION\_NOTIFICATION\_CLICK.equals(i.getAction())

|| ACTION\_WIDGET.equals(i.getAction())

|| ACTION\_WIDGET\_TAKE\_PHOTO.equals(i.getAction())

|| ((Intent.ACTION\_SEND.equals(i.getAction())

|| Intent.ACTION\_SEND\_MULTIPLE.equals(i.getAction())

|| INTENT\_GOOGLE\_NOW.equals(i.getAction()))

&& i.getType() != null)

|| i.getAction().contains(ACTION\_NOTIFICATION\_CLICK);

}

private boolean noteAlreadyOpened(Note note) {

DetailFragment detailFragment = (DetailFragment) getFragmentManagerInstance().findFragmentByTag(

FRAGMENT\_DETAIL\_TAG);

return detailFragment != null && NotesHelper.haveSameId(note, detailFragment.getCurrentNote());

}

public void switchToList() {

FragmentTransaction transaction = getFragmentManagerInstance().beginTransaction();

animateTransition(transaction, TRANSITION\_HORIZONTAL);

ListFragment mListFragment = new ListFragment();

transaction.replace(R.id.fragment\_container, mListFragment, FRAGMENT\_LIST\_TAG).addToBackStack

(FRAGMENT\_DETAIL\_TAG).commitAllowingStateLoss();

if (getDrawerToggle() != null) {

getDrawerToggle().setDrawerIndicatorEnabled(false);

}

getFragmentManagerInstance().getFragments();

EventBus.getDefault().post(new SwitchFragmentEvent(SwitchFragmentEvent.Direction.PARENT));

}

public void switchToDetail(Note note) {

FragmentTransaction transaction = getFragmentManagerInstance().beginTransaction();

animateTransition(transaction, TRANSITION\_HORIZONTAL);

DetailFragment mDetailFragment = new DetailFragment();

Bundle b = new Bundle();

b.putParcelable(INTENT\_NOTE, note);

mDetailFragment.setArguments(b);

if (getFragmentManagerInstance().findFragmentByTag(FRAGMENT\_DETAIL\_TAG) == null) {

transaction.replace(R.id.fragment\_container, mDetailFragment, FRAGMENT\_DETAIL\_TAG)

.addToBackStack(FRAGMENT\_LIST\_TAG)

.commitAllowingStateLoss();

} else {

getFragmentManagerInstance().popBackStackImmediate();

transaction.replace(R.id.fragment\_container, mDetailFragment, FRAGMENT\_DETAIL\_TAG)

.addToBackStack(FRAGMENT\_DETAIL\_TAG)

.commitAllowingStateLoss();

}

}

public void shareNote(Note note) {

String titleText = note.getTitle();

String contentText = titleText

+ System.getProperty("line.separator")

+ note.getContent();

Intent shareIntent = new Intent();

// Prepare sharing intent with only text

if (note.getAttachmentsList().isEmpty()) {

shareIntent.setAction(Intent.ACTION\_SEND);

shareIntent.setType("text/plain");

// Intent with single image attachment

} else if (note.getAttachmentsList().size() == 1) {

Attachment attachment = note.getAttachmentsList().get(0);

Uri shareableAttachmentUri = getShareableAttachmentUri(attachment);

if (shareableAttachmentUri != null) {

shareIntent.setAction(Intent.ACTION\_SEND);

shareIntent.setType(attachment.getMime\_type());

shareIntent.putExtra(Intent.EXTRA\_STREAM, shareableAttachmentUri);

}

// Intent with multiple images

} else if (note.getAttachmentsList().size() > 1) {

shareIntent.setAction(Intent.ACTION\_SEND\_MULTIPLE);

ArrayList<Uri> uris = new ArrayList<>();

// A check to decide the mime type of attachments to share is done here

HashMap<String, Boolean> mimeTypes = new HashMap<>();

for (Attachment attachment : note.getAttachmentsList()) {

Uri shareableAttachmentUri = getShareableAttachmentUri(attachment);

if (shareableAttachmentUri != null) {

uris.add(shareableAttachmentUri);

mimeTypes.put(attachment.getMime\_type(), true);

}

}

// If many mime types are present a general type is assigned to intent

if (mimeTypes.size() > 1) {

shareIntent.setType("\*/\*");

} else {

shareIntent.setType((String) mimeTypes.keySet().toArray()[0]);

}

Sha reIntent.putParcelableArrayListExtra(Intent.EXTRA\_STREAM, uris);

}

shareIntent.putExtra(Intent.EXTRA\_SUBJECT, titleText);

shareIntent.putExtra(Intent.EXTRA\_TEXT, contentText);

startActivity(Intent.createChooser(shareIntent, getResources().getString(R.string.share\_message\_chooser)));

}

public @Nullable Uri getShareableAttachmentUri(Attachment attachment) {

try {

return FileProviderHelper.getShareableUri(attachment);

} catch (FileNotFoundException e) {

LogDelegate.e(e.getMessage());

Toast.makeText(this, R.string.attachment\_not\_found, Toast.LENGTH\_SHORT).show();

return null;

}

}

/\*\*

\* Single note permanent deletion

\*

\* @param note Note to be deleted

\*/

public void deleteNote(Note note) {

new NoteProcessorDelete(Collections.singletonList(note)).process();

BaseActivity.notifyAppWidgets(this);

LogDelegate.d("Deleted permanently note with ID '" + note.get\_id() + "'");

}

public void updateWidgets() {

new UpdateWidgetsTask(getApplicationContext())

.executeOnExecutor(AsyncTask.THREAD\_POOL\_EXECUTOR);

}

public void showMessage(int messageId, Style style) {

showMessage(getString(messageId), style);

}

public void showMessage(String message, Style style) {

// ViewGroup used to show Crouton keeping compatibility with the new Toolbar

runOnUiThread(

() -> Crouton.makeText(this, message, style, binding.croutonHandle.croutonHandle).show());

}

@Override

public void onSharedPreferenceChanged(SharedPreferences sharedPreferences, String key) {

prefsChanged = t