Snack Squad: A Customizable Snack Ordering and Delivery App

1. INTRODUCTION

1.1 Description

A project that demonstrates the use of Android Jetpack Compose to build a UI for a snack squad app. Snack Squad is a sample project built using the Android Compose UI toolkit. It demonstrates how to create a simple e-commerce app for snacks usingthe Compose libraries. The user can see a list of snacks, and by tapping on asnack, and by tapping on the "Add to Cart" button, the snack will be added to the cart.The user can also see the list of items in the cart and can proceed to checkout to make the purchase.

Online snack ordering is the process of ordering snack, for delivery or pickup, from a website or other application. The product can be either ready-to-eat food (e.g., direct from a home-kitchen, restaurant, or a virtual restaurant) or food that has not been specially prepared for direct consumption (e.g., vegetables direct from a farm/garden, fruits, frozen meats. etc).Customers browse a digital menu, either on an app or website and place and pay for their order online.

An online food ordering system allows your business to accept and manage orders placed online for delivery or takeaway. Customers browse a digital menu, either on an app or website and place and pay for their order online. Venues will then receive the order details via their chosen online food ordering system and produce the order ready for delivery or customer pickup.

While online food ordering systems have been around for several years, demand for this technology took off during the pandemic. With extended lockdowns and restrictions, hospitality venues quickly pivoted their businesses to offer online ordering solutions so their customers could continue to enjoy restaurant-quality food at home.

1.2 Purpose:

The purpose of an online ordering system is to make itself beneficial for the customer and the business so that they can stay afloat while also serving customers their favorite dishes. With Food Aggregators increasing their commission every quarter, it is unsustainable for the restaurant to manage their restaurant while depending on food delivery orders from swiggy and zomato. The online ordering market is expanding and, especially the online food ordering segment is growing at a very rapid pace. Food Delivery is the preferred way for customers to enjoy food these days. This change has been roughly owed to the pandemic where customers prefer to order food online instead of dining out.

Objectives

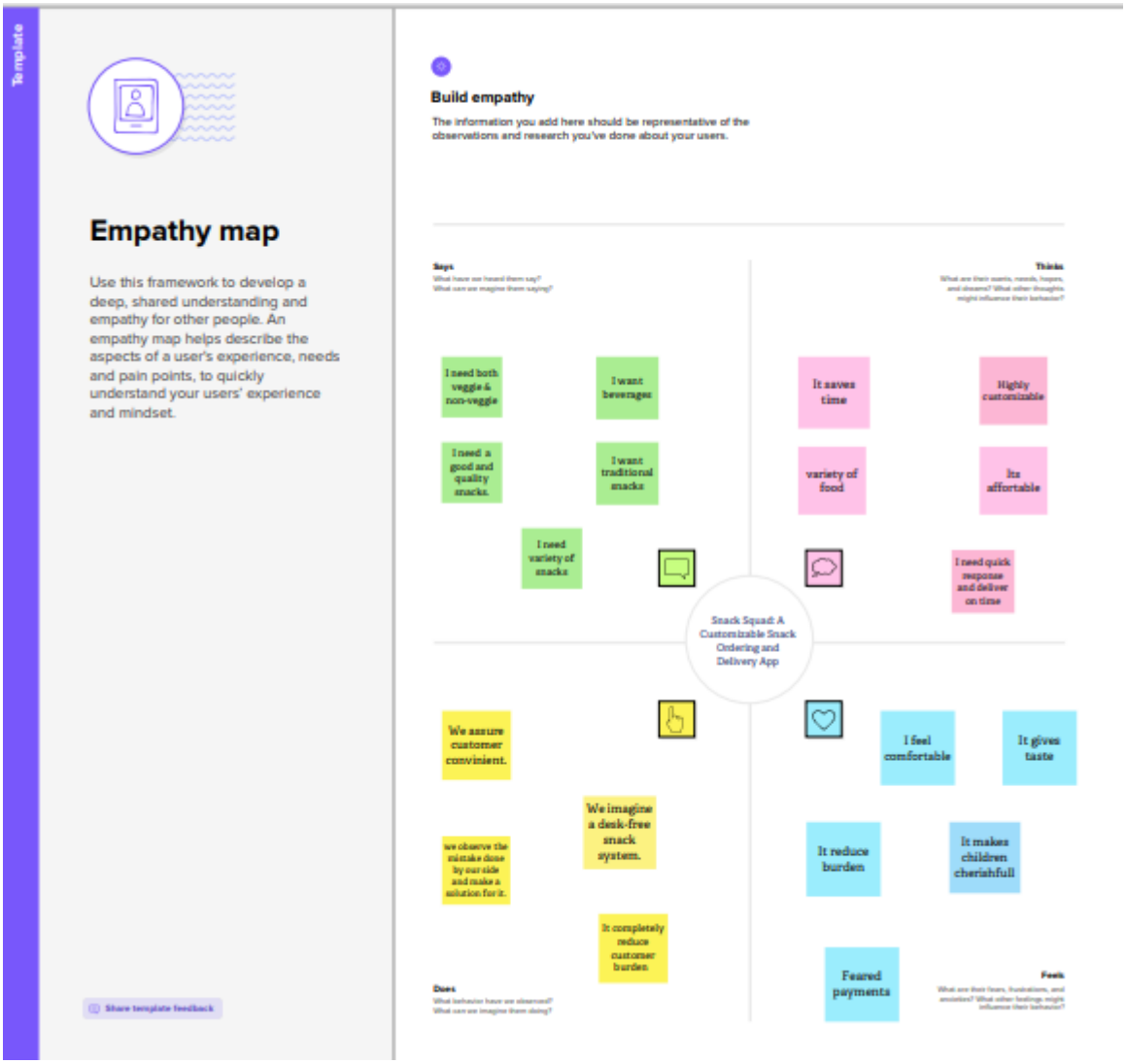
- To develop a system that will surely satisfy the customer service.
- To design a system able to accommodate huge number of orders at a time.
- To evaluate its performance and acceptability in terms of security, user-friendliness, accuracy and reliability.
- To improve the communication between the client and the server and minimize the time of ordering.

Project Workflow:

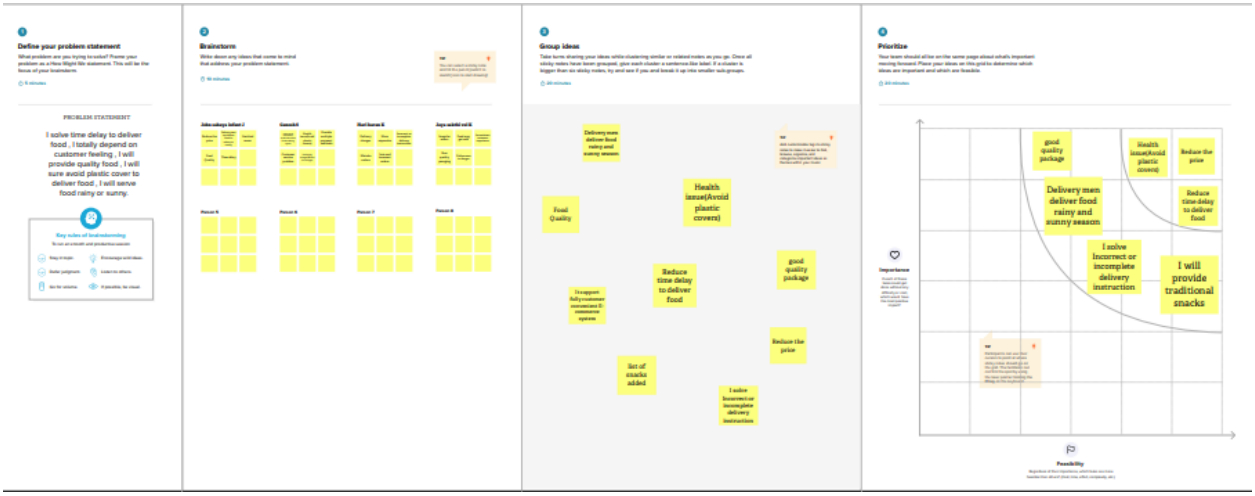
- Users register into the application.
- After registration, user logins into the application.
- User enters into the main page
- User can view the items, select and order the items

2. Problem Definition and Design Thinking

2.1 Empathy Map

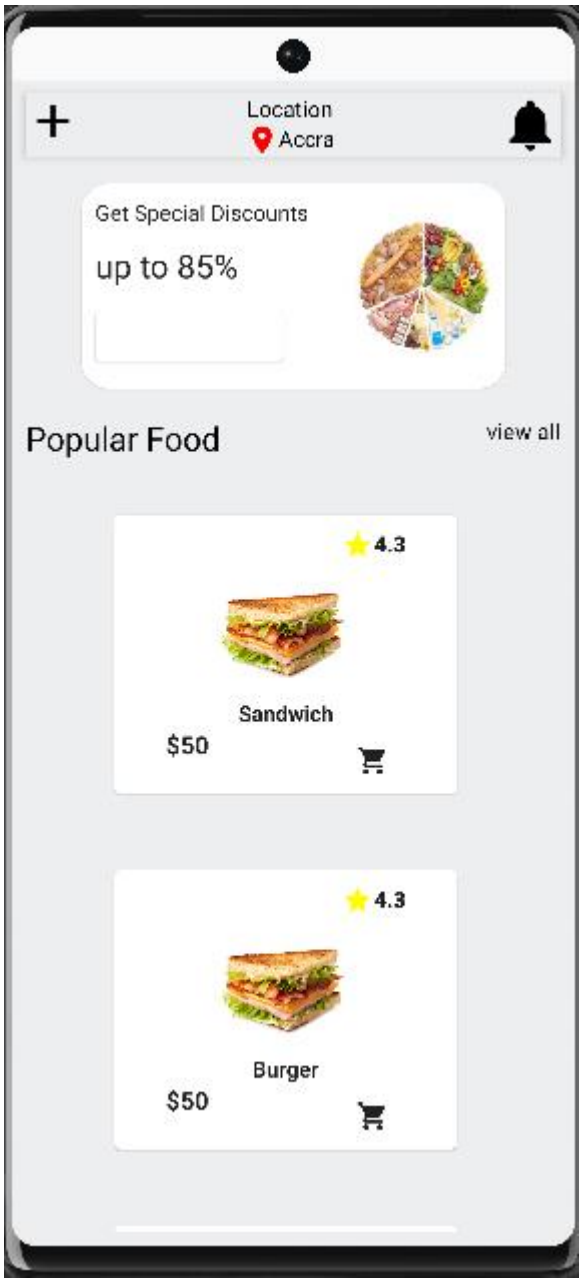


2.2 Ideation & Design thinking



3.Result


Login
Username
ganesh
Password
123456
Invalid username or password
Login
Sign up
Forget password?


Location
Accra
Get Special Discounts
up to 85%
Popular Food
view all
4.3
Sandwich
$50
4.3
Burger
$50

4.1 Advantages

Running an online food ordering system adds flexibility to the business, which will ultimately increase sales and profits.

Easy, fast, and comfortable:

In short, your customers choose to order food online because it is really at their fingertips.Anyone with a smartphone can order food online from their favorite restaurant. According to a Harris poll, millennials (under the age of 30) are your most important target audience today.More than 97% of millennials use their phones for anything. Ordering food online comes into the same broad category.So using the online food ordering system is the easiest way to attract millennials.

Health benefits:

One of the important benefits of food ordering systems is health benefits. Because the meal is planned, it is easy to determine the exact number of calories consumed in each meal. Many food ordering systems retain their menu for health benefits and weight loss, which can be very helpful for individuals who are trying to lose weight and start a healthy diet.

Safer and healthier:

To reopen, food businesses will have to set up shop to meet the health and safety regulations of the Indian government. Owners must maintain social distances, use non-contact ordering and payment methods, and ensure surfaces are regularly cleaned.Even if you are a small shop owner, Switching to the online ordering system for businesses means that your customers can order food without coming to the store and pay online without contact. This method not only brings profit to your business but also protects from the spread of covid-19.

Less chance for errors:

One of the best advantages of an online food ordering system for customers is that it ensures prices are accurate and there is less room for error when it comes time to settle the bill.This is because customers have to select an item in the menu at the appropriate price and make sure that the right amount is always paid.

This has some good benefits for your business; The chances of mischarging are low, less time in ordering errors, and helping to provide satisfactory service to customers.

More customers:

As the new life progresses with technologies, online orders and payments are expected to be accepted.If your payment and menu method is hassle free, your regular customers will recommend you to their friends and will share on social media about your restaurant.You can maximize your customers and your profits by providing a seamless customer experience.

Increased customer loyalty:

If you give customers a reason to come back, they will choose your store over your competitor. You can promote their loyalty through the loyalty program.According to a recent study, a personalized digital experience is also a great way to encourage customers to come back. According to a recent survey out Of the 1000 customers, 50% said they change brands that offer a worse online experience, and 73% expect online customization.Through a restaurant online ordering system, you can give personalized offers to loyal customers, request reviews to increase your ratings, and get feedback on your service.

Higher customer spends:

We all know that more and more customers are now engaging in digital products and services than ever before.They also spend more when ordering online. Because reading the online menu is different from standing in line.Customers have more time to make decisions. If they want a rich, gooey chocolate cake, they can order a lean latte without fearing the judgment of others. Even better if you have a carry-on bag!Those who do not have food intolerance can take their time to read all the necessary information. This means customers are more likely to place large orders.

Highly customizable:

Food ordering apps are highly customizable so you can easily advertise your logo, brand colors, or other features that make your business unique. Additionally, if you want to delete or add an item to the menu, you must sign in, make your changes, and it’s done.

4.2 Disadvantages:

While there are many advantages to the online food ordering system, there are also some disadvantages to online food ordering systems. They are

Price:

One of the major drawbacks of online food ordering systems is price. When food is ordered for more than one person, the cost is usually equal to eating at a good restaurant every night. Many food ordering systems cost more than $ 20 per person per day. Even more expensive for some other food ordering systems. For individuals with a limited food budget, online food ordering systems are often too expensive.

Limited menu:

Another disadvantage for food ordering systems is menu choices. Most food ordering systems have a limited number of meals. The menu changes every few weeks or months, but if you stick to the system for more than a few months the menu items will come back again and again. You should also eat the food provided for that week. If you do not want to eat that particular food, you may have to order another food from another place or eat food you do not like.

Preparation:

The preparation factor may be a disadvantage to food ordering systems. Most food ordering systems give frozen food. They are usually easy to prepare, but they usually take more than an hour to cook because the food is frozen. To avoid long cooking times, you can remove the food from the freezer the day before. However, remember to eliminate food from the freezer to reduce cooking time.

Quality of food may be suffer:

One problem with the food ordering system is that the quality of the food served is often worse than eating at a restaurant. Often, food has to be fed over long distances, and over time, precious vitamins can be lost. Also, food from the ordering system is often served in plastic packaging, which may not be very appealing to your eyes compared to the food neatly placed on your plate in a restaurant.

Food may get cold:

Due to the long ordering distances, your food may also be cold once it is finally delivered to your home.You need to reheat it or eat it cold.This is especially true, if you order in an emergency the streets are often crowded and the ordering person will be stuck in traffic.

The vibe of the restaurant is missing:

In some restaurants, there is also a good circumstance which you will miss if you order your food at home.For example, if you spend your evening in a good Chinese restaurant, you will often feel like you are actually in China because the decoration and the whole atmosphere are in line with the Chinese way of life.If you order food at your home, you will lose all of these.Also, from time to time, it would be great if you could take your partner or family to a nice restaurant for dinner to spend a good evening.

Wrapping up:

Despite some drawbacks, online food ordering systems generally benefit customers and restaurants. By trying out new restaurants (or ordering convenient meals) and helping restaurants attract new customers, customers can taste the delicacies foods as if they were in their bed. We understand both sides of this mess. brilliant technologies are here, contact us to provide growth to your online food ordering system through mobile apps.

5. Application

16 Must-Have Features Of An Online Food Ordering App

1. Push Notifications

2. Discount/Rewards, Cashback, And Loyalty Programs

3. Real-Time GPS Tracking

4. Easy Payment Options

5. Social Media Integration

6. Reviews & Ratings

7. Easy Order Placement

8. Order Scheduling And Pickup

9. QR Codes

10. Geofencing

11. Search Filters

12. Order History

13. In-App Messages

14. Favourites (Restaurants/Dishes)

15. Contactless Delivery

16. Voice integration

6. Conclusion

• Should allow Computer Science students to browse through the code and application:

This can be achieved when students are able to run and install the application. When they run theapplication, they can browse through the implementation of different objects.

• Should allow users to browse through different product categories: This is achieved throughan easy to use graphical interface menu options.

• Should allow users to save items to the cart and view detailed information about the order:

The users can add any number of items to the cart from any of the available food categories bysimply clicking the Add to Cart button for each item. Once item is added to the cart, user is

presented with detailed order to review or continue shopping.

• Should allow the user to CheckOut the item(s): This is achieved using the “Proceed to checkoutbutton” in the cart initially and then “CheckOut” button at last step after “review Order” step..

Button is disabled when there are no items in the cart.

• Should allow the user to process the payment: This is achieved when user selects “Processed to

Checkout” button and fill up the Payment information details.

• Should allow the user to see Success message after placing an order: This is achieved when

user successfully places an order. The user is given the order conformation number along withsuccess message.

7. Future Scope

The following section describes the work that will be implemented with future releases of the software.

• Customize orders: Allow customers to customize food orders

• Enhance User Interface by adding more user interactive features. Provide Deals and promotional

Offer details to home page. Provide Recipes of the Week/Day to Home Page

• Payment Options: Add different payment options such as PayPal, Cash, Gift Cards etc. Allow to

save payment details for future use.

• Allow to process an order as a Guest

• Delivery Options: Add delivery option

• Order Process Estimate: Provide customer a visual graphical order status bar

• Order Status: Show only Active orders to Restaurant Employees.

• Order Ready notification: Send an Order Ready notification to the customer

• Restaurant Locator: Allow to find and choose a nearby restaurant

• Integrate with In store touch screen devices like iPad.

8. Appendix

Source code:

https://docs.google.com/document/d/1RJi8v1X0sKrE9RD6oNlmiVo3IJQauo5TPhqRMWMmMXM/edit?usp=share_link

Gradle

```
#!/usr/bin/env sh

#
# Copyright 2015 the original author or authors.
#
# Licensed under the Apache License, Version 2.0 (the "License");
# you may not use this file except in compliance with the License.
# You may obtain a copy of the License at
#
#      https://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.
#

##############################################################################
##
##  Gradle start up script for UN*X
##
##############################################################################
```

```
# Attempt to set APP_HOME
# Resolve links: $0 may be a link
PRG="$0"
# Need this for relative symlinks.
while [ -h "$PRG" ] ; do
    ls=`ls -ld "$PRG"`
    link=`expr "$ls" : '.*-> \(.*\)$'`
    if expr "$link" : '/.*' > /dev/null; then
        PRG="$link"
    else
        PRG=`dirname "$PRG"`"/$link"
    fi
done
SAVED="`pwd`"
cd "`dirname \"$PRG\"`/" >/dev/null
APP_HOME="`pwd -P`"
cd "$SAVED" >/dev/null

APP_NAME="Gradle"
APP_BASE_NAME=`basename "$0"`

# Add default JVM options here. You can also use JAVA_OPTS and GRADLE_OPTS to pass JVM options to this script.
DEFAULT_JVM_OPTS='"-Xmx64m" "-Xms64m"'

# Use the maximum available, or set MAX_FD != -1 to use that value.
MAX_FD="maximum"

warn () {
    echo "$*"
}

die () {
    echo
    echo "$*"
    echo
    exit 1
```

```
}

# OS specific support (must be 'true' or 'false').
cygwin=false
msys=false
darwin=false
nonstop=false
case "`uname`" in
  CYGWIN* )
    cygwin=true
    ;;
  Darwin* )
    darwin=true
    ;;
  MINGW* )
    msys=true
    ;;
  NONSTOP* )
    nonstop=true
    ;;
esac

CLASSPATH=$APP_HOME/gradle/wrapper/gradle-wrapper.jar



# Determine the Java command to use to start the JVM.
if [ -n "$JAVA_HOME" ] ; then
    if [ -x "$JAVA_HOME/jre/sh/java" ] ; then
        # IBM's JDK on AIX uses strange locations for the executables
        JAVACMD="$JAVA_HOME/jre/sh/java"
    else
        JAVACMD="$JAVA_HOME/bin/java"
    fi
    if [ ! -x "$JAVACMD" ] ; then
        die "ERROR: JAVA_HOME is set to an invalid directory: $JAVA_HOME

Please set the JAVA_HOME variable in your environment to match the
```

```
location of your Java installation."
    fi
else
    JAVACMD="java"
    which java >/dev/null 2>&1 || die "ERROR: JAVA_HOME is not set and no 'java' command could
be found in your PATH.

Please set the JAVA_HOME variable in your environment to match the
location of your Java installation."
fi

# Increase the maximum file descriptors if we can.
if [ "$cygwin" = "false" -a "$darwin" = "false" -a "$nonstop" = "false" ] ; then
    MAX_FD_LIMIT=`ulimit -H -n`
    if [ $? -eq 0 ] ; then
        if [ "$MAX_FD" = "maximum" -o "$MAX_FD" = "max" ] ; then
            MAX_FD="$MAX_FD_LIMIT"
        fi
        ulimit -n $MAX_FD
        if [ $? -ne 0 ] ; then
            warn "Could not set maximum file descriptor limit: $MAX_FD"
        fi
    else
        warn "Could not query maximum file descriptor limit: $MAX_FD_LIMIT"
    fi
fi

# For Darwin, add options to specify how the application appears in the dock
if $darwin; then
    GRADLE_OPTS="$GRADLE_OPTS \"-Xdock:name=$APP_NAME\" \"-
Xdock:icon=$APP_HOME/media/gradle.icns\""
fi

# For Cygwin or MSYS, switch paths to Windows format before running java
if [ "$cygwin" = "true" -o "$msys" = "true" ] ; then
    APP_HOME=`cygpath --path --mixed "$APP_HOME"`
    CLASSPATH=`cygpath --path --mixed "$CLASSPATH"`
```

```
JAVACMD=`cygpath --unix "$JAVACMD"`

# We build the pattern for arguments to be converted via cygpath
ROOTDIRSRAW=`find -L / -maxdepth 1 -mindepth 1 -type d 2>/dev/null`
SEP=""
for dir in $ROOTDIRSRAW ; do
    ROOTDIRS="$ROOTDIRS$SEP$dir"
    SEP="|"
done
OURCYGPATTERN="(^($ROOTDIRS))"
# Add a user-defined pattern to the cygpath arguments
if [ "$GRADLE_CYGPATTERN" != "" ] ; then
    OURCYGPATTERN="$OURCYGPATTERN|($GRADLE_CYGPATTERN)"
fi
# Now convert the arguments - kludge to limit ourselves to /bin/sh
i=0
for arg in "$@" ; do
    CHECK=`echo "$arg"|egrep -c "$OURCYGPATTERN" -`
    CHECK2=`echo "$arg"|egrep -c "^-"`                        ### Determine if an option

    if [ $CHECK -ne 0 ] && [ $CHECK2 -eq 0 ] ; then                    ### Added a condition
        eval `echo args$i`=`cygpath --path --ignore --mixed "$arg"`
    else
        eval `echo args$i`="\"$arg\""
    fi
    i=`expr $i + 1`
done
case $i in
    0) set -- ;;
    1) set -- "$args0" ;;
    2) set -- "$args0" "$args1" ;;
    3) set -- "$args0" "$args1" "$args2" ;;
    4) set -- "$args0" "$args1" "$args2" "$args3" ;;
    5) set -- "$args0" "$args1" "$args2" "$args3" "$args4" ;;
    6) set -- "$args0" "$args1" "$args2" "$args3" "$args4" "$args5" ;;
    7) set -- "$args0" "$args1" "$args2" "$args3" "$args4" "$args5" "$args6" ;;
```

```
        8) set -- "$args0" "$args1" "$args2" "$args3" "$args4" "$args5" "$args6" "$args7" ;;
        9) set -- "$args0" "$args1" "$args2" "$args3" "$args4" "$args5" "$args6" "$args7" "$args8" ;;
    esac
fi

# Escape application args
save () {
    for i do printf %s\\n "$i" | sed "s/'/'\\\\''/g;1s/^/'/;\$s/\$/' \\\\/" ; done
    echo " "
}
APP_ARGS=`save "$@"`

# Collect all arguments for the java command, following the shell quoting and substitution rules
eval set -- $DEFAULT_JVM_OPTS $JAVA_OPTS $GRADLE_OPTS "\"-
Dorg.gradle.appname=$APP_BASE_NAME\"" -classpath "\"$CLASSPATH\""
org.gradle.wrapper.GradleWrapperMain "$APP_ARGS"

exec "$JAVACMD" "$@"
```

MAIN

```
plugins {
    id 'com.android.application'
    id 'org.jetbrains.kotlin.android'
}

android {
    namespace 'com.example.snackordering'
    compileSdk 33

    defaultConfig {
        applicationId "com.example.snackordering"
        minSdk 24
        targetSdk 33
        versionCode 1
        versionName "1.0"

        testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"
        vectorDrawables {
            useSupportLibrary 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
    }
    kotlinOptions {
        jvmTarget = '1.8'
    }
    buildFeatures {
        compose true
    }
    composeOptions {
        kotlinCompilerExtensionVersion '1.2.0'
    }
    packagingOptions {
        resources {
            excludes += '/META-INF/{AL2.0,LGPL2.1}'
        }
    }
}

dependencies {

    implementation 'androidx.core:core-ktx:1.7.0'
    implementation 'androidx.lifecycle:lifecycle-runtime-ktx:2.3.1'
    implementation 'androidx.activity:activity-compose:1.3.1'
    implementation "androidx.compose.ui:ui:$compose_ui_version"
    implementation "androidx.compose.ui:ui-tooling-preview:$compose_ui_version"
    implementation 'androidx.compose.material:material:1.2.0'
    implementation 'androidx.room:room-common:2.5.0'
    implementation 'androidx.room:room-ktx:2.5.0'
    testImplementation 'junit:junit:4.13.2'
    androidTestImplementation 'androidx.test.ext:junit:1.1.5'
    androidTestImplementation 'androidx.test.espresso:espresso-core:3.5.1'
    androidTestImplementation "androidx.compose.ui:ui-test-junit4:$compose_ui_version"
    debugImplementation "androidx.compose.ui:ui-tooling:$compose_ui_version"
    debugImplementation "androidx.compose.ui:ui-test-manifest:$compose_ui_version"
}
```

Output video:

https://drive.google.com/file/d/108bcwkos7qq7J6K6Pjdvh3M3SqXKcc3G/view?usp=share_link