ДОДАТОК А

Текст програми клієнтського додатку

```
apply plugin: 'com.android.application' android {
//задаємо версії засобів розробки compileSdkVersion 23
buildToolsVersion "23.0.3 defaultConfig {
applicationId "com.example.koza4.ServerAnalizer" minSdkVersion 21
targetSdkVersion 23
versionCode 1
versionName "1.0"
}
buildTypes {
release {
minifyEnabled false
proguardFiles getDefaultProguardFile('proguard-android.txt'),
'proguard-rules.pro'
}
}
packagingOptions {
exclude 'META-INF/LICENSE' exclude 'META-INF/NOTICE' exclude
'META-INF/NOTICE.txt'
}
}
//підключення зовнішніх бібліотек dependencies {
compile fileTree(dir: 'libs', include: ['*.jar']) testCompile
'junit:junit:4.12'
compile project(':myapplication')
compile 'com.android.support:design:24.0.0-beta1' compile
'com.android.support:cardview-v7:24.0.0-beta1' compile
'com.github.bumptech.glide:glide:3.6.0' compile
'de.hdodenhof:circleimageview:1.3.0'
compile 'com.android.support:appcompat-v7:24.0.0-beta1' compile
'com.flaviofaria:kenburnsview:1.0.6'
compile 'com.google.android.gms:play-services-auth:9.0.2' compile
'com.firebase:firebase-client-android:2.3.1' compile
```

Текст програми клієнтського додатку

```
'com.android.support:recyclerview-v7:24.0.0-beta1' compile
'com.facebook.android:facebook-android-sdk:4.6.0' compile
'com.viewpagerindicator:library:2.4.1'
compile 'com.github.medyo:fancybuttons:1.5@aar'
compile 'com.google.android.gms:play-services-identity:9.0.2'
compile 'com.google.android.gms:play-services:9.0.2'
compile 'com.google.android.gms:play-services-ads:9.0.2' compile
'com.google.android.gms:play-services-gcm:9.0.2'
    }

<resources>
    <!--- Задаємо значення строковим ресурсам додатку -->
    <string name="app_name">ServerAnalizer</string>
    <string name="splash_bg">qq</string>
    <string name="fontello_logo">&#xe800;</string>
    <string name="fontello_search">&#xe816;</string>
    <string name="fontello_x_mark">&#xe81f;</string>
    <string name="fontello_x_mark_masked">&#xe806;</string>
    <string name="fontello_microfon">&#xe80f;</string>
    <string name="fontello_user">&#xe81b;</string>
    <string name="fontello_password">&#xe823;</string>
    <string name="fontello_upload">&#xe81a;</string>
    <string name="fontello_play">&#xe813;</string>
    <string name="fontello_heart_empty">&#xe80b;</string>
    <string name="fontello_heart_full">&#xe80a;</string>
    <string name="fontello_drag_and_drop">&#xe821;</string>
    <string name="welcome">Welcome!</string>
    <string name="facebook_app_id">377505602442747</string>
    <string
name="twitter_app_key">vvvcQrekKC6d1r9FNhhxxW10K</string>
    <string
name="twitter_app_secret">hSgApoxqojbfEH13yub3B8QlVs2pd7KQooyQHlrwELa6
Lryrqh<
    /string>
```

Текст програми клієнтського додатку

```
<string name="action_settings">Settings</string>
<string name="app_overview_1st_item_text"> </string>
<string name="app_overview_2nd_item_text"> </string>
<string name="app_overview_3rd_item_text"> </string>
<string name="app_overview_4th_item_text"> </string>
<string name="icon_user">&#xf007;</string>
<string name="icon_add">&#xf067;</string>
<string name="icon_readmore">&#xf178;</string>
<string name="icon_answer">&#xf095;</string>
<string name="icon_mute">&#xf131;</string>
<string name="icon_like">&#xf087;</string>
<string name="icon_share">&#xf064;</string>
<string name="icon_follow">&#xf005;</string>
<string name="icon_android">&#xf17b;</string>
<string name="icon_dropbox">&#xf16b;</string>
<string name="icon_soundcloud">&#xf1be;</string>
<string name="icon_spotify">&#xf1bc;</string>
<string name="icon_vine">&#xf1ca;</string>
<string name="icon_twitter">&#xf099;</string>
<string name="icon_tumbler">&#xf173;</string>
<string name="icon_stackoverflow">&#xf16c;</string>
<string name="icon_gplus">&#xf0d5;</string>
<string name="icon_facebook">&#xf09a;</string>
<string name="icon_entypo_facebook">&#59140;</string>
<string name="icon_behance">&#xf1b4;</string>
<string name="icon_dribble">&#xf17d;</string>
<string name="icon_send">&#xf1d8;</string>
<string name="icon_download">&#xf0ed;</string>
<string name="icon_upload">&#xf0ee;</string>
<string name="icon_config">&#xf085;</string>
<string name="icon_creditcard">&#xf09d;</string>
<string name="ghost_button_title">Ghost Button</string>
<string  name="icon_envelope">□</string>
<string name="checkin">Checkin</string>
```

Текст програми клієнтського додатку

```
    <string name="menu_settings">Settings</string>
    <string name="title_activity_maps">Map</string>
    </resources>

    <!--- Задаємо розмітку для вікна входу у систему -->
    <RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:app="http://schemas.android.com/apk/tools"
android:layout_height="match_parent"
android:layout_width="match_parent"
xmlns:fancy="http://schemas.android.com/apk/res-auto"
android:fillViewport="true">
    <FrameLayout
    android:layout_width="match_parent"
android:layout_height="match_parent">
    <FrameLayout
    android:layout_width="match_parent"
android:layout_height="match_parent"
android:layout_marginBottom="95dp">
    <com.flaviofaria.kenburnsview.KenBurnsView
android:layout_width="match_parent"
android:layout_height="match_parent"  android:id="@+id/icon_image1"
android:scaleType="centerCrop"/>
    <com.flaviofaria.kenburnsview.KenBurnsView
android:layout_width="match_parent"
android:layout_height="match_parent"  android:id="@+id/icon_image2"
android:scaleType="centerCrop" />
    </FrameLayout>
    <android.support.v4.view.ViewPager
android:layout_width="fill_parent"
android:layout_height="fill_parent"
android:id="@+id/intro_view_pager"/>
    <com.viewpagerindicator.CirclePageIndicator
android:id="@+id/indicator" android:layout_width="fill_parent"
```

Текст програми клієнтського додатку

```
android:layout_height="wrap_content"
android:layout_gravity="center_horizontal|bottom"
app:fillColor="@color/material_grey_800"
app:pageColor="@color/material_grey_400" app:radius="3dp"
     app:strokeWidth="1dp"                app:strokeColor="#ff838b"
android:layout_marginBottom="84dp" />
     <LinearLayout
     android:layout_width="match_parent"
android:layout_height="wrap_content"   android:background="#ff5f69"
android:gravity="center"               android:paddingBottom="30dp"
android:paddingTop="30dp"
android:layout_gravity="center_vertical|bottom">
     <mehdi.sakout.fancybuttons.FancyButton
android:id="@+id/btn_create_account"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_marginRight="20dp"            android:padding="10dp"
fancy:fb_borderColor="#FFFFFF"           fancy:fb_borderWidth="1dp"
fancy:fb_defaultColor="#ff5f69"       fancy:fb_focusColor="#ff838b"
fancy:fb_radius="30dp"     fancy:fb_text="Create     an     account"
fancy:fb_textColor="#FFFFFF" >
     </mehdi.sakout.fancybuttons.FancyButton>
     <mehdi.sakout.fancybuttons.FancyButton
android:id="@+id/btn_login"     android:layout_width="wrap_content"
android:layout_height="wrap_content"   android:paddingBottom="10dp"
android:paddingLeft="20dp"              android:paddingRight="20dp"
android:paddingTop="10dp"            fancy:fb_borderColor="#FFFFFF"
fancy:fb_borderWidth="1dp"          fancy:fb_defaultColor="#ff5f69"
fancy:fb_focusColor="#ff838b"
fancy:fb_fontIconResource="@string/icon_user"
fancy:fb_iconPosition="left" fancy:fb_radius="30dp"
     fancy:fb_text="Login" fancy:fb_textColor="#FFFFFF" />
     </LinearLayout>
     </FrameLayout>
```

Текст програми клієнтського додатку

```
</RelativeLayout>
<?xml version="1.0" encoding="utf-8"?><!--- Малюємо панель навігації -->
<shape
xmlns:android="http://schemas.android.com/apk/res/android"
android:shape="rectangle">
<gradient
android:angle="135" android:endColor="@color/colorPrimaryDark"
android:startColor="@color/colorPrimary" android:type="linear"/>
</shape>
/*
*  Даний клас передбачає вікно входу у систему, де користувач можескористатися
*  або логіном та паролем, або увійти за допомогою профілю соціальної мережі.
*  Також у наведеному класі передбачено слайдер, завдяки якому користувачможе
*  познайомитись з коротким описом функцій додатку.
*  Якщо користувач вже був авторизований у системі, вікно входу незʼявляється
*/
package com.example.koza4.ServerAnalizer; import android.content.Intent;
import android.content.pm.ActivityInfo; import android.os.Bundle;
import android.support.annotation.NonNull;
import android.support.design.widget.NavigationView; import android.support.v4.view.PagerAdapter;
import android.support.v4.view.ViewPager; import android.support.v7.widget.RecyclerView; import android.util.Log;
import android.view.View; import android.view.ViewGroup; import android.view.Window;
```

Текст програми клієнтського додатку

```
import android.view.animation.Animation; import android.view.animation.AnimationUtils; import android.widget.Button;
import android.widget.ImageView; import android.widget.TextView; import android.widget.Toast; import com.androidquery.AQuery;
import com.example.koza4.myapplication.ui.auth.core.AuthProviderType;
import com.example.koza4.myapplication.ui.auth.core.FirebaseLoginBaseActivity; import com.example.koza4.myapplication.ui.auth.core.FirebaseLoginError;
import com.firebase.client.AuthData;
import com.firebase.client.Firebase; import com.firebase.client.Query;
import com.google.firebase.auth.FirebaseAuth; import com.google.firebase.auth.FirebaseUser; import com.viewpagerindicator.CirclePageIndicator; import com.viewpagerindicator.PageIndicator; import mehdi.sakout.fancybuttons.FancyButton;

public class LoginActivity extends FirebaseLoginBaseActivity {
private ViewPager viewPager;
private PageIndicator indicator; private ImageView topImage1;
private ImageView topImage2;
private int lastPage = 0; private int[] icons; private int[] messages; private Firebase mRef;
public static String TAG = "FirebaseUI.chat"; private String mName;
private FirebaseAuth mAuth;
private FirebaseAuth.AuthStateListener mAuthListener;
//функція, що з'являється під час створення Activity @Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
```

Текст програми клієнтського додатку

```
requestWindowFeature(Window.FEATURE_NO_TITLE); mAuth = FirebaseAuth.getInstance();
setRequestedOrientation(ActivityInfo.SCREEN_ORIENTATION_PORTRAIT); Firebase.setAndroidContext(this); //зв'язуємось з сервером Firebase setContentView(R.layout.activity_login);
//mLoginButton = (Button) findViewById(R.id.btn_login); FancyButton startMessagingButton = (FancyButton)
findViewById(R.id.btn_login);
startMessagingButton.setOnClickListener(new View.OnClickListener() { @Override
public void onClick(View view) {
Runnable runnable = new Runnable() {//створюємо новий потік public void run() {
showFirebaseLoginPrompt();
}};
Thread thread = new Thread(runnable);thread.start();
}});
FancyButton CreateAccButton = (FancyButton) findViewById(R.id.btn_create_account);
CreateAccButton.setOnClickListener(new View.OnClickListener() {@Override
public void onClick(View view) {
Intent mainIntent = new Intent(LoginActivity.this, ThreeActivity.class);
LoginActivity.this.startActivity(mainIntent);
/* Finish splash activity so user cant go back to it. */ LoginActivity.this.finish();
}
});
mRef = new Firebase("https://glaring-torch-7273.firebaseio.com"); AQuery aq = new AQuery(this);//задаємо слайдер
aq.id(R.id.icon_image1).image(R.drawable.app_overview_1_buy);
icons = new int[]{
```

Текст програми клієнтського додатку

```
R.drawable.app_overview_1_buy, R.drawable.app_overview_2_smartphone,
R.drawable.app_overview_3_save_money,
R.drawable.app_overview_4_coupons
    };
    messages = new int[]{//повідомлення слайдера
R.string.app_overview_1st_item_text,
    R.string.app_overview_2nd_item_text,
R.string.app_overview_3rd_item_text,
R.string.app_overview_4th_item_text
    };
    viewPager = (ViewPager) findViewById(R.id.intro_view_pager);
topImage1 = (ImageView) findViewById(R.id.icon_image1); topImage2
= (ImageView) findViewById(R.id.icon_image2); indicator =
(CirclePageIndicator) findViewById(R.id.indicator);
topImage2.setVisibility(View.GONE);
    viewPager.setAdapter(new IntroAdapter());
viewPager.setPageMargin(0); viewPager.setOffscreenPageLimit(1);
indicator.setViewPager(viewPager);
indicator.setOnPageChangeListener(new
    ViewPager.OnPageChangeListener() {@Override
    public void onPageScrolled(int position, float positionOffset,
int positionOffsetPixels) {//при прокручуванні сторінки
    if (lastPage != viewPager.getCurrentItem()) { lastPage =
viewPager.getCurrentItem(); final ImageView fadeoutImage;
    final ImageView fadeinImage;
    if (topImage1.getVisibility() == View.VISIBLE) { fadeoutImage
= topImage1;
    fadeinImage = topImage2;
    } else {
    fadeoutImage = topImage2;fadeinImage = topImage1;
    }
```

```
    fadeinImage.bringToFront();
fadeinImage.setImageResource(icons[lastPage]);
fadeinImage.clearAnimation(); fadeoutImage.clearAnimation();
```

Текст програми клієнтського додатку

```
    Animation                    outAnimation                    =
AnimationUtils.loadAnimation(LoginActivity.this,
R.anim.icon_anim_fade_out);   outAnimation.setAnimationListener(new
Animation.AnimationListener() {
    @Override
    public void onAnimationStart(Animation animation) {
    }
    @Override
    public    void    onAnimationEnd(Animation    animation)    {
fadeoutImage.setVisibility(View.GONE);
    }
    @Override
    public void onAnimationRepeat(Animation animation) {
    }
    }); //анімація при перелистуванні Animation inAnimation =
    AnimationUtils.loadAnimation(LoginActivity.this,
R.anim.icon_anim_fade_in);inAnimation.setAnimationListener(new
    Animation.AnimationListener() {
    @Override
    public    void    onAnimationStart(Animation    animation)    {
fadeinImage.setVisibility(View.VISIBLE);
    }
    @Override
    public void onAnimationEnd(Animation animation) {}
    @Override
    public void onAnimationRepeat(Animation animation) {}
    fadeoutImage.startAnimation(outAnimation);
fadeinImage.startAnimation(inAnimation);
    }}
    @Override
    public void onPageSelected(int position) {
```

```
    }
    @Override
    public void onPageScrollStateChanged(int state) {
```

Текст програми клієнтського додатку

```
    }});
    mAuthListener = new FirebaseAuth.AuthStateListener() {
@Override
    public void onAuthStateChanged(@NonNull FirebaseAuth
firebaseAuth) {
    user.getUid());
    FirebaseUser user = firebaseAuth.getCurrentUser();if (user !=
null) {
    // User is signed in
    Log.d(TAG, "onAuthStateChanged:signed_in:" +

    } else {
    // User is signed out
    Log.d(TAG, "onAuthStateChanged:signed_out");
    }}};}
    private class IntroAdapter extends PagerAdapter {//клас для
зображення слайдера
    @Override
    public int getCount() {return 4;
    }
    @Override
    public boolean isViewFromObject(View view, Object object) {
return view.equals(object);
    }
    @Override
    public Object instantiateItem(ViewGroup container, int
position) {View view = View.inflate(container.getContext(),
    R.layout.intro_view_layout, null);
    TextView messageTextView = (TextView)
view.findViewById(R.id.message_text);
```

```
container.addView(view, 0);
messageTextView.setText(AndroidUtilities.replaceTags(getString(message
s[posit ion]), getApplicationContext()));
return view;}
```

Текст програми клієнтського додатку

```
@Override
public void destroyItem(ViewGroup container, int position,
Objectobject) {
((ViewPager) container).removeView((View) object);
}}
@Override
protected void onStart() {super.onStart();
setEnabledAuthProvider(AuthProviderType.FACEBOOK);
setEnabledAuthProvider(AuthProviderType.TWITTER);
setEnabledAuthProvider(AuthProviderType.GOOGLE);
setEnabledAuthProvider(AuthProviderType.PASSWORD);
mAuth.addAuthStateListener(mAuthListener);
}
@Override
public void onStop() {super.onStop();
if (mAuthListener != null) {
mAuth.removeAuthStateListener(mAuthListener);
}
// ...
}
@Override //при успішній авторизації
public void onFirebaseLoggedIn(AuthData authData) {
Log.i(TAG, "Logged in to " +
authData.getProvider().toString()); switch (authData.getProvider())
{
case "password":
mName = (String) authData.getProviderData().get("email");
break;
default:
mName = (String) authData.getProviderData().get("displayName");
```

```
    break;
    }
    Toast.makeText(getApplicationContext(),                mName,
Toast.LENGTH_LONG).show();
```

Текст програми клієнтського додатку

```
    invalidateOptionsMenu();
    Intent    mainIntent    =    new    Intent(LoginActivity.this,
ThreeActivity.class);
    LoginActivity.this.startActivity(mainIntent);
    /* Finish splash activity so user cant go back to it. */
LoginActivity.this.finish();
    }
    @Override
    public void onFirebaseLoggedOut() {//при виході з облікового
записуLog.i(TAG, "Logged out");
    mName = ""; invalidateOptionsMenu();
    }
    @Override //при помилці входу
    public  void  onFirebaseLoginProviderError(FirebaseLoginError
firebaseError) {
    Log.e(TAG,      "Login      provider      error:      "      +
firebaseError.toString());resetFirebaseLoginPrompt();
    }
    @Override //при помилці входу
    public    void    onFirebaseLoginUserError(FirebaseLoginError
firebaseError)   {   Log.e(TAG,   "Login   user   error:   "   +
firebaseError.toString()); resetFirebaseLoginPrompt();
    }
    @Override
    public Firebase getFirebaseRef(){return mRef;}}
```

Лістинг класу OverviewActivity.java

```
    package          com.example.koza4.splashscreen;          import
android.content.Intent;
```

```
import android.os.Bundle;
import android.support.design.widget.FloatingActionButton;
import android.support.design.widget.NavigationView; import
android.support.design.widget.Snackbar;
```

Текст програми клієнтського додатку

```
import android.support.design.widget.TabLayout; import
android.support.v4.app.Fragment;
import android.support.v4.app.FragmentManager; import
android.support.v4.app.FragmentPagerAdapter; import
android.support.v4.view.GravityCompat; import
android.support.v4.view.ViewPager;
import android.support.v4.widget.DrawerLayout; import
android.support.v7.app.ActionBar;
import android.support.v7.app.AppCompatActivity; import
android.support.v7.widget.Toolbar;
import android.view.LayoutInflater;import android.view.Menu;
import android.view.MenuItem; import android.view.View;
import android.widget.TextView;import android.widget.Toast;
import com.firebase.client.AuthData; import
com.firebase.client.Firebase;
import com.google.firebase.auth.FirebaseAuth; import
java.util.ArrayList;
import java.util.List;
   /* Клас який зображує вікно з панеллю навігації, на цьому
                        вікні користувач
 *  може переходити від виконання запитів до списку доступних
                                                  серверів, або
 *  обраного списку серверу з можливостями, на навігаційній
панелі він можекурувати своїм обліковим записом
 */
public class OverviewActivity extends AppCompatActivity {
private DrawerLayout mDrawerLayout;
private NavigationView navigationView, navigation_view;
private String userName;
private Firebase ref;
```

```
    private TextView txtProfileName, txtEmail; private
FirebaseAuth mAuth;
    private FirebaseAuth.AuthStateListener mAuthListener;
```

Текст програми клієнтського додатку

```
    public static String TAG = "FirebaseUI.chat"; private int[]
imageResId = {
    R.drawable.ic10, R.drawable.ic2, R.drawable.ic4,
R.drawable.ic3
    };
    @Override
    protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_three);
Firebase.setAndroidContext(this);
    Toolbar toolbar = (Toolbar) findViewById(R.id.toolbar);
setSupportActionBar(toolbar);
    ref = new Firebase("https://glaring-torch-
7273.firebaseio.com");mAuth = FirebaseAuth.getInstance();
    final ActionBar ab = getSupportActionBar();
ab.setHomeAsUpIndicator(R.drawable.ic_menu);
ab.setDisplayHomeAsUpEnabled(true);
    mDrawerLayout = (DrawerLayout)
findViewById(R.id.drawer_layout); navigationView = (NavigationView)
findViewById(R.id.nav_view); if (navigationView != null) { //
задаємо навігаційну панель
    setupDrawerContent(navigationView);
    }
    navigationView.setNavigationItemSelectedListener(new
NavigationView.OnNavigationItemSelectedListener() {
    // This method will trigger on item Click of navigation menu
@Override
    public boolean onNavigationItemSelected(MenuItem menuItem) {
    //Checking if the item is in checked state or not, if notmake
it in checked state
```

```
    if (menuItem.isChecked()) menuItem.setChecked(false); else
menuItem.setChecked(true);
    //Closing drawer on item clickmDrawerLayout.closeDrawers();
```

Текст програми клієнтського додатку

```
    //Check to see which item was being clicked and perform
appropriate action
    switch (menuItem.getItemId()) {
    //Replacing the main content with ContentFragment Which is our
Inbox View;
    case R.id.nav_home: Toast.makeText(getApplicationContext(),
"Inbox
    Selected", Toast.LENGTH_SHORT).show();
    ContentFragment fragment = new ContentFragment();
android.support.v4.app.FragmentTransaction
    fragmentTransaction =
getSupportFragmentManager().beginTransaction();
    fragmentTransaction.replace(R.id.frame,fragment);
fragmentTransaction.commit();
    return true;
    // For rest of the options we just show a toast on clickcase
R.id.nav_messages:
    Toast.makeText(getApplicationContext(), "Stared Selected",
Toast.LENGTH_SHORT).show();
    return true;
    case R.id.nav_friends: Toast.makeText(getApplicationContext(),
"Send
    Selected", Toast.LENGTH_SHORT).show();
    return true;
    case R.id.nav_discussion:
Toast.makeText(getApplicationContext(), "Drafts
    Selected", Toast.LENGTH_SHORT).show();
    return true; case R.id.dashboard:
    Toast.makeText(getApplicationContext(), "All Mail Selected",
Toast.LENGTH_SHORT).show();
```

```
return true;case R.id.forum:
Toast.makeText(getApplicationContext(), "Loggin out",
Toast.LENGTH_SHORT).show();// повідомлення про вихід з профілю
ref.unauth();
```

Текст програми клієнтського додатку

```
Intent mainIntent = new Intent(ThreeActivity.this,
LoginActivity.class); // перехід на вікно входу
ThreeActivity.this.startActivity(mainIntent);
ThreeActivity.this.finish();
return true;default:
Toast.makeText(getApplicationContext(), "Somethings Wrong",
Toast.LENGTH_SHORT).show();
return true;}}});
ViewPager viewPager = (ViewPager)
findViewById(R.id.viewpager);if (viewPager != null) {
setupViewPager(viewPager);
}
FloatingActionButton fab = (FloatingActionButton)
findViewById(R.id.fab);
fab.setOnClickListener(new View.OnClickListener() {@Override
public void onClick(View view) { Snackbar.make(view, "Here's a
Snackbar",
Snackbar.LENGTH_LONG)
.setAction("Action", null).show();
}
});
TabLayout tabLayout = (TabLayout) findViewById(R.id.tabs);
tabLayout.setupWithViewPager(viewPager);
tabLayout.getTabAt(0).setIcon(imageResId[0]);
tabLayout.getTabAt(1).setIcon(imageResId[1]);
tabLayout.getTabAt(2).setIcon(imageResId[2]);
tabLayout.getTabAt(3).setIcon(imageResId[3]);
View header =
LayoutInflater.from(this).inflate(R.layout.nav_header,null);
navigationView.addHeaderView(header);
```

```
txtProfileName = (TextView) header.findViewById(R.id.username); txtEmail = (TextView) header.findViewById(R.id.textView); ref.addAuthStateListener(new Firebase.AuthStateListener() {
```

Текст програми клієнтського додатку

```
@Override
public void onAuthStateChanged(AuthData authData) { if (authData != null) {
userName = authData.getProviderData().get("displayName").toString();
//зображення даних користувача на навігаційній панелі
txtProfileName.setText(userName);
txtEmail.setText("test_mail@gmail.com");
} else {
Toast.makeText(getApplicationContext(), "Somethings Wrong", Toast.LENGTH_SHORT).show();
}}});}
@Override
public boolean onCreateOptionsMenu(Menu menu) { //меню налаштувань getMenuInflater().inflate(R.menu.sample_actions, menu);
return true;
}
@Override
public boolean onOptionsItemSelected(MenuItem item) { switch (item.getItemId()) {
case android.R.id.home: mDrawerLayout.openDrawer(GravityCompat.START); return true;
}
return super.onOptionsItemSelected(item);
}
private void setupViewPager(ViewPager viewPager) {
Adapter adapter = new Adapter(getSupportFragmentManager()); adapter.addFragment(new CheeseListFragment(), "Hand picked"); adapter.addFragment(new LocationFragment(), "Map"); adapter.addFragment(new CardContentFragment(), "Places");
```

```
adapter.addFragment(new TileContentFragment(), "My list");
viewPager.setAdapter(adapter);
    }
```

Текст програми клієнтського додатку

```
    private void setupDrawerContent(NavigationView navigationView)
{navigationView.setNavigationItemSelectedListener(
    new NavigationView.OnNavigationItemSelectedListener() {
@Override
    public boolean onNavigationItemSelected(MenuItem menuItem) {}
    menuItem.setChecked(true); mDrawerLayout.closeDrawers(); return
true;
    }});
    static class Adapter extends FragmentPagerAdapter
{//зображення вкладокпереходу
    private final List<Fragment> mFragments = new ArrayList<>();
private final List<String> mFragmentTitles = new ArrayList<>();
public Adapter(FragmentManager fm) {
    super(fm);
    }
    public void addFragment(Fragment fragment, String title)
{mFragments.add(fragment);// додання вкладинки
mFragmentTitles.add(title);}
    @Override
    public Fragment getItem(int position) { return
mFragments.get(position);
    }
    @Override
    public int getCount() { return mFragments.size();
    }
    @Override
    public CharSequence getPageTitle(int position) { return
mFragmentTitles.get(position);
    }}}
```

```
package com.example.koza4.ServerAnalizer.ui; import
android.app.Activity;
import android.view.View; import android.view.ViewGroup;
```

Текст програми клієнтського додатку

```
import android.widget.BaseAdapter; import
com.firebase.client.DataSnapshot; import
com.firebase.client.Firebase; import com.firebase.client.Query;
/** Даний клас реалізує підтримку компоненту ListView з
сервером Firebase*/
public abstract class FirebaseListAdapter<T> extends
BaseAdapter {private final Class<T> mModelClass;
protected int mLayout;
protected Activity mActivity;//містить ListView FirebaseArray
mSnapshots;
public FirebaseListAdapter(Activity activity, Class<T>
modelClass, intmodelLayout, Query ref) {
mModelClass = modelClass;
mLayout = modelLayout;// представлення однієї строки в
ListViewmActivity = activity;
mSnapshots = new FirebaseArray(ref);//змінна що представляє
Firebase mSnapshots.setOnChangedListener(new
FirebaseArray.OnChangedListener()
{
@Override
public void onChanged(EventType type, int index, int oldIndex)
{notifyDataSetChanged();
}});}
public FirebaseListAdapter(Activity activity, Class<T>
modelClass, intmodelLayout, Firebase ref) {
this(activity, modelClass, modelLayout, (Query) ref);
}
public void cleanup() {
// We're being destroyed, let go of our mListener and forget
aboutall of the mModels
```

```
mSnapshots.cleanup();
}
@Override
public int getCount() {
```

Текст програми клієнтського додатку

```
return mSnapshots.getCount();
}
@Override
public T getItem(int position) {
return parseSnapshot(mSnapshots.getItem(position));
}// Цей метод аналізує отримані дані та повертає в необхідний тип.protected T parseSnapshot(DataSnapshot snapshot) {
return snapshot.getValue(mModelClass);
}
public Firebase getRef(int position) {
return mSnapshots.getItem(position).getRef();
}
@Override
public long getItemId(int i) {
//       http://stackoverflow.com/questions/5100071/whats-the-purpose-of-item-ids-in-android-listview-adapter
return mSnapshots.getItem(i).getKey().hashCode();
}
@Override
public View getView(int position, View view, ViewGroup viewGroup) {if (view == null) {
view = mActivity.getLayoutInflater().inflate(mLayout, viewGroup,
false);}
T model = getItem(position);
// Call out to subclass to marshall this model into the provided viewpopulateView(view, model, position);
return view;
abstract protected void populateView(View v, T model, int position);}
```

```
package com.example.koza4.ServerAnalizer.ui.auth.core; import android.content.Context;
import android.content.Intent;import android.util.Log;
```

Текст програми клієнтського додатку

```
import com.firebase.client.AuthData; import com.firebase.client.Firebase; import com.firebase.client.FirebaseError;import java.util.HashMap;
import java.util.Map;
/*
* Даний абстрактний клас передбачає вибір способу авторизації
*/
public abstract class FirebaseAuthProvider {
private static final String TAG = "FirebaseAuthProvider"; private final Context mContext;
private final AuthProviderType mAuthProviderType; private final String mProviderName;
private final Firebase mRef;
private final TokenAuthHandler mHandler;public abstract void logout();
public Context getContext() {return mContext;}
//отримуємо спосіб авторизації
public AuthProviderType getProviderType() { return mAuthProviderType;}
//отримуємо ім’я сервісу
public String getProviderName() {return mProviderName;}
public Firebase getFirebaseRef() {return mRef;}
public TokenAuthHandler getHandler() {return mHandler;}
//повертаємо дані необхідні для авторизації
protected FirebaseAuthProvider(Context context, AuthProviderType providerType, String providerName, Firebase ref, TokenAuthHandler handler) {
mContext = context; mAuthProviderType = providerType; mProviderName = providerName; mRef = ref;
mHandler = handler;
```

```
    }
    //перегрузка функції public void login() {
    Log.w("FirebaseAuthProvider", "Login() is not supported for
providertype " + getProviderName());}
```

Текст програми клієнтського додатку

```
    //перегрузка функції
    public void login(String email, String password) {
Log.w("FirebaseAuthProvider", "Login(String email, String password)
    is not supported for provider type " + getProviderName());
    }
    //при відправленні токена
    public void onFirebaseTokenReceived(FirebaseOAuthToken token,
TokenAuthHandler handler) {
    authenticateRefWithOAuthFirebasetoken(token, handler);
    }

    public void onActivityResult(int requestCode, int resultCode,
Intentdata) {
    }

    private                                                 void
authenticateRefWithOAuthFirebasetoken(FirebaseOAuthToken
    token,      final      TokenAuthHandler      handler)      {
Firebase.AuthResultHandler authResultHandler = new
    Firebase.AuthResultHandler() {
    @Override
    public    void    onAuthenticated(AuthData    authData)    {
handler.onSuccess(authData);
    }

    @Override
    public void onAuthenticationError(FirebaseError firebaseError)
{Log.e(TAG, "Authentication failed: " +
    firebaseError.toString());
```

```
handler.onProviderError(new
FirebaseLoginError(FirebaseResponse.PROVIDER_NOT_ENABLED,     "Make
sure " + getProviderName() + " login is enabled and configured in
your Firebase. (" +firebaseError.toString() + ")"));
}};
```

Текст програми клієнтського додатку

```
if (token.mode == FirebaseOAuthToken.SIMPLE) {
// Simple mode is used for Facebook and Google auth
getFirebaseRef().authWithOAuthToken(token.provider, token.token,
authResultHandler);
} else if (token.mode == FirebaseOAuthToken.COMPLEX) {
// Complex mode is used for Twitter auth Map<String, String>
options = new HashMap<>(); options.put("oauth_token",
token.token); options.put("oauth_token_secret", token.secret);
options.put("user_id", token.uid);
getFirebaseRef().authWithOAuthToken(token.provider, options,
authResultHandler);}}}
```

```
package com.example.koza4.ServerAnalizer.ui.auth.core; import
android.app.AlertDialog;
import android.app.Dialog;
import android.app.DialogFragment; import
android.content.Context; import android.content.Intent; import
android.os.Bundle;
import android.view.LayoutInflater;import android.view.View;
import android.widget.EditText; import
com.firebase.client.AuthData;import com.firebase.client.Firebase;
import com.example.koza4.myapplication.R;
import
com.example.koza4.myapplication.ui.auth.google.GoogleAuthProvider;
import java.util.HashMap;
import java.util.Map;
/*Даний абстрактний клас передбачає з'єднання з сервером та
діалогове вікно
*авторизації
```

```
     */
     public class FirebaseLoginDialog extends DialogFragment {
Map<AuthProviderType, FirebaseAuthProvider>
mEnabledProvidersByType = newHashMap<>();
```

Текст програми клієнтського додатку

```
     TokenAuthHandler mHandler; AuthProviderType mActiveProvider;
Firebase mRef;
     Context mContext;View mView; @Override
     public void onStop() {super.onStop(); cleanUp();
     }
     @Override
     public void onDestroy() {super.onDestroy(); cleanUp();
     }
     @Override
     public void onPause() {super.onPause(); cleanUp();
     }
     public void cleanUp() {
     if (getGoogleAuthProvider() != null)
getGoogleAuthProvider().cleanUp();
     }
     public void onActivityResult(int requestCode, int resultCode,
Intentdata) {
     for (FirebaseAuthProvider provider :
mEnabledProvidersByType.values()) {
     provider.onActivityResult(requestCode, resultCode, data);
     }}
     @Override
     public Dialog onCreateDialog(Bundle savedInstanceState) {
AlertDialog.Builder builder = new
AlertDialog.Builder(getActivity()); LayoutInflater inflater =
getActivity().getLayoutInflater();
     mView = inflater.inflate(R.layout.fragment_firebase_login,
null); for (AuthProviderType providerType :
AuthProviderType.values()) {
```

```
     if (mEnabledProvidersByType.keySet().contains(providerType)) {
showLoginOption(mEnabledProvidersByType.get(providerType),
     providerType.getViewId());
     } else {
```

Текст програми клієнтського додатку

```
     mView.findViewById(providerType.getViewId()).setVisibility(Vie
w.GONE);}}
     if
(mEnabledProvidersByType.containsKey(AuthProviderType.PASSWORD) &&

     !(mEnabledProvidersByType.containsKey(AuthProviderType.FACEBOO
K)  || mEnabledProvidersByType.containsKey(AuthProviderType.GOOGLE)
||  mEnabledProvidersByType.containsKey(AuthProviderType.TWITTER)))
{
     mView.findViewById(R.id.or_section).setVisibility(View.GONE);
     }
     mView.findViewById(R.id.loading_section).setVisibility(View.GONE)
; builder.setView(mView);
     this.setRetainInstance(true); return builder.create();
     }
     public FirebaseLoginDialog setRef(Firebase ref) {mRef = ref;
     return this;}
     public  FirebaseLoginDialog  setContext(Context  context)  {
mContext = context;
     return this;
     }
     public void reset() { //перезавантаження вікна if (mView !=
null) {
     mView.findViewById(R.id.login_section).setVisibility(View.VISIBLE
);
mView.findViewById(R.id.loading_section).setVisibility(View.GONE);
     }
     }
```

```
    public void logout() { //при виході з акаунту for
(FirebaseAuthProvider provider :
    mEnabledProvidersByType.values()) {
    provider.logout();
    }mRef.unauth();}
```

Текст програми клієнтського додатку

```
    public FirebaseLoginDialog setHandler(final TokenAuthHandler
handler) {mHandler = new TokenAuthHandler() {
    @Override
    public void onSuccess(AuthData auth) { dismiss();
handler.onSuccess(auth);
    }
    @Override
    public void onUserError(FirebaseLoginError err) {
handler.onUserError(err);
    }
    @Override
    public void onProviderError(FirebaseLoginError err) {
handler.onProviderError(err);
    }
    };
    return this;
    }
    public FirebaseLoginDialog setEnabledProvider(AuthProviderType
provider)
    {
    if (!mEnabledProvidersByType.containsKey(provider)) {
mEnabledProvidersByType.put(provider,
    provider.createProvider(mContext, mRef, mHandler));}
    return this;
    }
    private void showLoginOption(final FirebaseAuthProvider
helper, int id) { mView.findViewById(id).setOnClickListener(new
View.OnClickListener()
```

```
{
@Override
public void onClick(View view) {
if (AuthProviderType.getTypeForProvider(helper) ==
AuthProviderType.PASSWORD) {
```

Текст програми клієнтського додатку

```
EditText emailText = (EditText)
mView.findViewById(R.id.email);EditText passwordText = (EditText)
mView.findViewById(R.id.password);
helper.login(emailText.getText().toString(),
passwordText.getText().toString());
passwordText.setText("");} else {helper.login();}
mActiveProvider = helper.getProviderType();

mView.findViewById(R.id.login_section).setVisibility(View.GONE
);
mView.findViewById(R.id.loading_section).setVisibility(View.VISIBLE);
}
});
}
public GoogleAuthProvider getGoogleAuthProvider() { return
(GoogleAuthProvider)
mEnabledProvidersByType.get(AuthProviderType.GOOGLE);}}

package com.example.koza4.splashscreen; import
android.content.Context;
import android.content.Intent;import android.os.Bundle;
import android.support.design.widget.Snackbar; import
android.support.v4.app.Fragment;
import android.support.v7.widget.LinearLayoutManager; import
android.support.v7.widget.RecyclerView; import
android.view.LayoutInflater;
import android.view.View; import android.view.ViewGroup;
import android.widget.Button;
```

```
import android.widget.ImageButton; import android.widget.Toast;
/**
* Provides UI for the view with Cards.
*/
public class CardContentFragment extends Fragment {@Override
```

Текст програми клієнтського додатку

```
public View onCreateView(LayoutInflater inflater, ViewGroup container,
Bundle savedInstanceState) { RecyclerView recyclerView = (RecyclerView) inflater.inflate(
R.layout.recycler_view, container, false); ContentAdapter adapter = new ContentAdapter(); recyclerView.setAdapter(adapter); recyclerView.setHasFixedSize(true);
recyclerView.setLayoutManager(new
LinearLayoutManager(getActivity())); return recyclerView;
}
public static class ViewHolder extends RecyclerView.ViewHolder { public ViewHolder(LayoutInflater inflater, ViewGroup parent) {
super(inflater.inflate(R.layout.item_card, parent, false)); itemView.setOnClickListener(new View.OnClickListener() {
@Override
public void onClick(View v) {
Context context = v.getContext();
Intent intent = new Intent(context, FourActivity.class); context.startActivity(intent);
}
});
// Adding Snackbar to Action Button inside cardButton button = (Button)
itemView.findViewById(R.id.action_button); button.setOnClickListener(new View.OnClickListener() {
@Override
```

```
    public void onClick(View v) { Snackbar.make(v, "Action is
pressed",
    Snackbar.LENGTH_LONG).show();}});
    ImageButton favoriteImageButton =(ImageButton)
    itemView.findViewById(R.id.favorite_button);
favoriteImageButton.setOnClickListener(new View.OnClickListener()
    {
    @Override
```

Текст програми клієнтського додатку

```
    public void onClick(View v) { Snackbar.make(v, "Added to
Favorite",
    Snackbar.LENGTH_LONG).show();
    }
    });
    ImageButton shareImageButton = (ImageButton)
itemView.findViewById(R.id.share_button);
    shareImageButton.setOnClickListener(new View.OnClickListener()
{@Override
    public void onClick(View v) { Snackbar.make(v, "Share
article",
    Snackbar.LENGTH_LONG).show();
    }});}}
    /*** Adapter to display recycler view.
    */
    @Override
    public void onBindViewHolder(ViewHolder holder, int position)
{// no-op
    }
    @Override
    public int getItemCount() {return LENGTH;
    }}}
```

```
    package com.example.koza4.myapplication.ui.auth.core; import
android.content.Context;
```

```
import com.firebase.client.Firebase; import com.example.koza4.myapplication.R;
import java.lang.reflect.InvocationTargetException;
/**
* Вибір методу входу до системи, з'єднання з соц. мережами
*/
public enum AuthProviderType {
GOOGLE("google", "google.GoogleAuthProvider", R.id.google_button),
```

Текст програми клієнтського додатку

```
FACEBOOK("facebook", "facebook.FacebookAuthProvider", R.id.facebook_button),
TWITTER("twitter", "twitter.TwitterAuthProvider", R.id.twitter_button),
PASSWORD("password", "password.PasswordAuthProvider", R.id.password_section);
private final static String AUTH_PACKAGE = "com.example.koza4.myapplication.ui.auth.";
private final String mName;
private final String mProviderName;private final int mViewId;
AuthProviderType(String name, String providerName, int viewId) {this.mName = name;
this.mProviderName = providerName;this.mViewId = viewId;
}
public String getName() {return mName;
}
public int getViewId() {return mViewId;
}
public FirebaseAuthProvider createProvider(Context context, Firebase ref,TokenAuthHandler handler) {
try {
Class<? extends FirebaseAuthProvider> clazz = (Class<? extends FirebaseAuthProvider>) Class.forName(AUTH_PACKAGE + mProviderName);
```

```
    return clazz.getConstructor(Context.class, AuthProviderType.class, String.class, Firebase.class, TokenAuthHandler.class).newInstance(context, this, this.getName(), ref,handler);
    } catch (NoSuchMethodException e) { throw new RuntimeException(e);
    } catch (IllegalAccessException e) { throw new RuntimeException(e);
    } catch (InstantiationException e) { throw new RuntimeException(e);
```

Текст програми клієнтського додатку

```
    } catch (InvocationTargetException e) { throw new RuntimeException(e);
    } catch (ClassNotFoundException e) { throw new RuntimeException(e);
    }}
    public static AuthProviderType getTypeForProvider(FirebaseAuthProviderprovider) {
    for (AuthProviderType type : AuthProviderType.values()) { if (provider.getProviderName() == type.getName()) {
    return type;}}
    throw new IllegalArgumentException("The provider you specified is notof a known type");}}

    /** Інтерфейс для обробки та стилізації тексту*/
    package com.example.koza4.ServerAnalizer; import android.content.Context;
    import android.graphics.Color;import android.text.Spannable;
    import android.text.SpannableStringBuilder; import android.text.Spanned;
    import android.text.style.ForegroundColorSpan; import java.util.ArrayList;
    public class AndroidUtilities { public static float density = 1;
```

```
    public static int dp(float value, Context c) {
    density = c.getResources().getDisplayMetrics().density; return
(int) Math.ceil(density * value);
    }
    public static Spannable replaceTags(String str, Context
context) {try {
    int start = -1;
    int startColor = -1;int end = -1;
    StringBuilder stringBuilder = new StringBuilder(str); while
((start = stringBuilder.indexOf("<br>")) != -1) {
    stringBuilder.replace(start, start + 4, "\n");}
```

Текст програми клієнтського додатку

```
    while ((start = stringBuilder.indexOf("<br/>")) != -1) {
stringBuilder.replace(start, start + 5, "\n");
    }
    ArrayList<Integer> bolds = new ArrayList<>();
ArrayList<Integer> colors = new ArrayList<>();
    while ((start = stringBuilder.indexOf("<b>")) != -1 ||
(startColor = stringBuilder.indexOf("<c")) != -1) {
    if (start != -1) {
    stringBuilder.replace(start, start + 3, ""); end =
stringBuilder.indexOf("</b>"); stringBuilder.replace(end, end + 4,
""); bolds.add(start);
    bolds.add(end);
    } else if (startColor != -1) {
stringBuilder.replace(startColor, startColor + 2, ""); end =
stringBuilder.indexOf(">", startColor);
    int color =
Color.parseColor(stringBuilder.substring(startColor, end));
    stringBuilder.replace(startColor, end + 1, ""); end =
stringBuilder.indexOf("</c>"); stringBuilder.replace(end, end + 4,
""); colors.add(startColor);
    colors.add(end); colors.add(color);
    }
    }
```

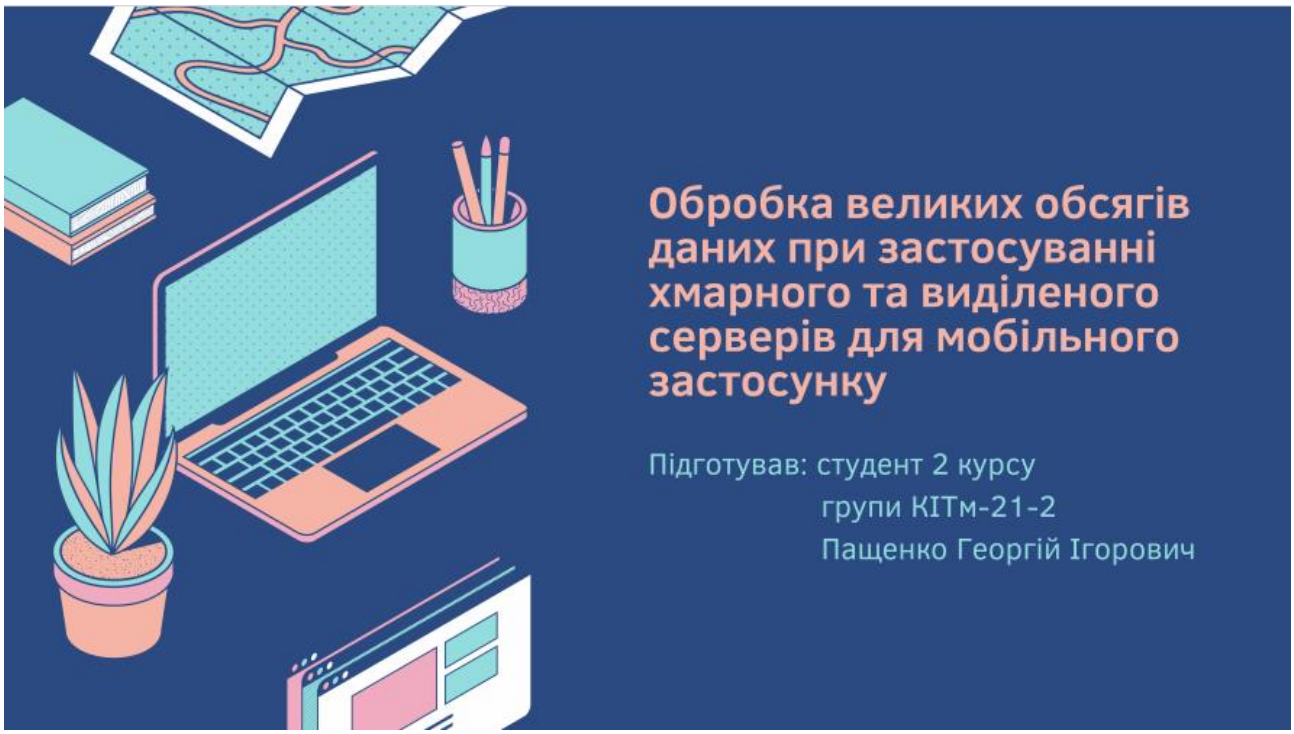
```
SpannableStringBuilder spannableStringBuilder = new SpannableStringBuilder(stringBuilder);
for (int a = 0; a < colors.size() / 3; a++) { spannableStringBuilder.setSpan(new
ForegroundColorSpan(colors.get(a * 3 + 2)), colors.get(a * 3), colors.get(a *
3 + 1), Spanned.SPAN_EXCLUSIVE_EXCLUSIVE);
}
return spannableStringBuilder;
} catch (Exception e) {}
return new SpannableStringBuilder(str);}}
```

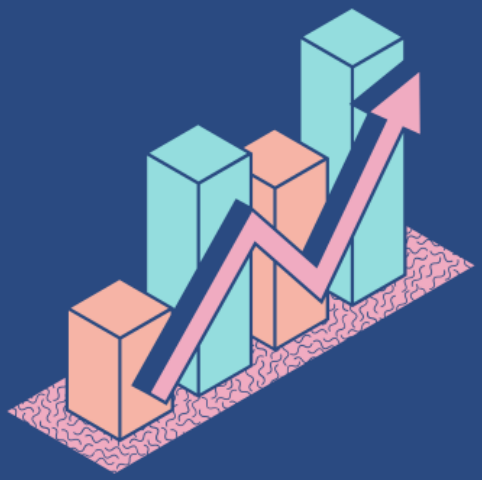
Текст програми клієнтського додатку

```
package com.example.koza4.splashscreen; import android.content.Context;
import android.graphics.Typeface; import android.util.AttributeSet;import android.widget.TextView;
//віджет для зображення тексту у компоненті TextView public class FontelloTextView extends TextView {
private static Typeface sFontello;
public FontelloTextView(Context context) {super(context);
if (isInEditMode()) return; //Won't work in Eclipse graphical layoutsetTypeface();
}

public FontelloTextView(Context context, AttributeSet attrs) { super(context, attrs);
if (isInEditMode()) return;setTypeface();
}

public FontelloTextView(Context context, AttributeSet attrs, int defStyle)
{
super(context, attrs, defStyle); if (isInEditMode()) return; setTypeface();
}
```

```
    private void setTypeface() {if (sFontello == null) {
    sFontello = Typeface.createFromAsset(getContext().getAssets(),
"fonts/Fontello.ttf");
    }
    setTypeface(sFontello);
    }
    }
```

ПРЕЗЕНТАЦІЯ



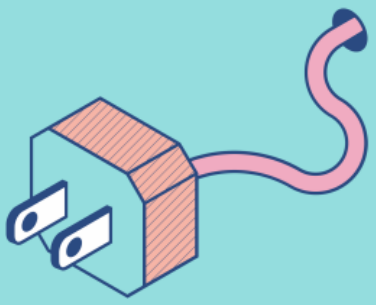


Актуальність

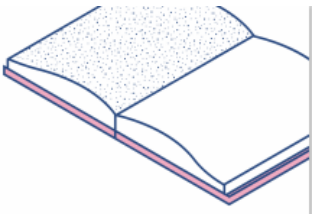
зумовлена наявністю значних відмінностей у традиційному підході до проектування мобільних додатків з використанням хмарних технологій та виділених серверів, такі як різні затрати часу, коштів та функцій.

Мета

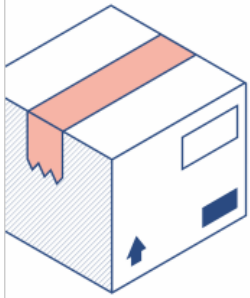
розробка програмного засобу Android додатку для статистичного аналізу та дослідження ефективності обробки великихоб'ємів даних на прикладі використання виділеного та хмарного серверів та опису відмінностей в технологіях.



Об’єкт досліджень



технологія хмарної реалізації бекенду, розробка системи для аналізу даних з отриманих результатів виконання операцій запитів на сервер.



Предмет дослідження

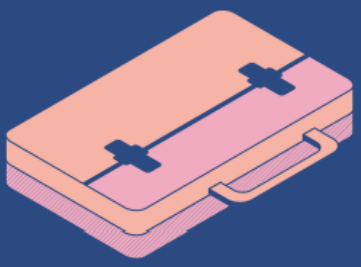
методики проектування серверу для мобільних додатків.

Веб-додаток

ДОДАТОК, В ЯКОМУ КЛІЄНТОМ Є ОГЛЯДАЧ ІНТЕРНЕТА, А СЕРВЕРОМ — ВЕБ-СЕРВЕР.

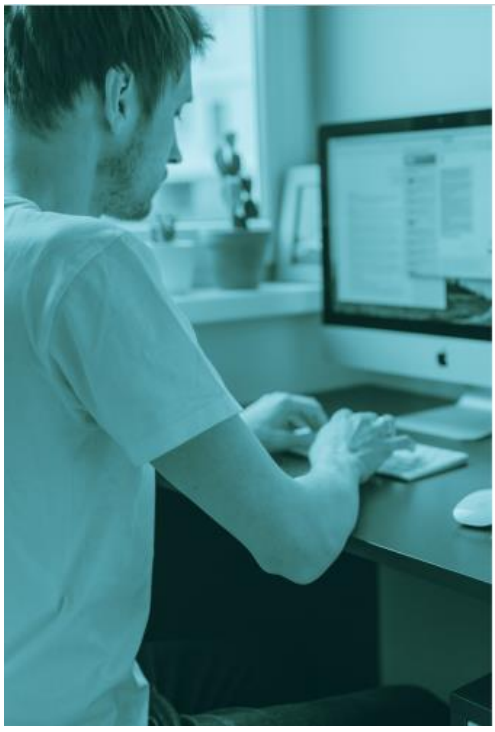
Додаток створюється один раз для довільно вибраної платформи і на ній розгортається. Проте різна реалізація HTML, CSS, DOM і інших специфікацій в браузерах може викликати проблеми при розробці веб-додатків і подальшої підтримки.

Мобільні додатки



Технології створення:

- Нативні. Програми написані рідними мовами мобільної платформи — Java та Kotlin для Android та Swift та objective-C для iOS/iPadOS.
- Гібридні. Універсальні програми, не прив'язані до платформи, у розробці використовуються одночасно нативні та вебтехнології.



Відмінність

веб-додатки працюватимуть в браузері на відміну від мобільних

Порівняння серверів

VPS, віртуальний приватний сервер

Є фіксованим сегментом ресурсів одного фізичного сервера. Хмарний сервер базується на кількох виділених серверах, і ви можете миттєво змінити конфігурацію.

Трудомісткість

$$t - t_o + t_u + t_a + t_n + t_{отл} + t_д, \text{люд./год.}$$

t_o – праця на підготовку й опис поставленої задачі, (приймається t_o – 50);

t_u – витрати праці на дослідження алгоритму рішення задачі;

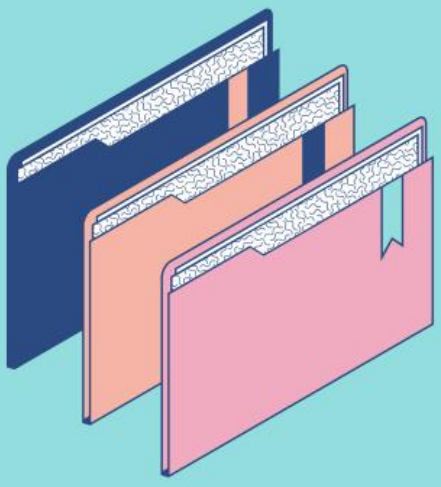
t_a – витрати праці на розробку блок–схеми алгоритму;

t_n – витрати праці на програмування по готовій блок–схемі;

$t_{отл}$ – витрати праці на налагодження програми на ЕОМ;

$t_д$ – витрати праці на підготовку документації.

Трудомісткість розробки ПЗ становить 1200 людино-годин, що дорівнює приблизно 150 робочих днів.



Витрати на створення програмного забезпечення

Зарплата програміста дорівнює 42100 за весь час розробки

Загальні втрати на розробку додатку становлять 56800

Основна мета економічної ефективності

Дати фінансову оцінку передбачуваних витрат та одержуваного корисного результату, а також оцінити прибутковість проекту і, в кінцевому підсумку, економічну доцільність його розробки та впровадження.

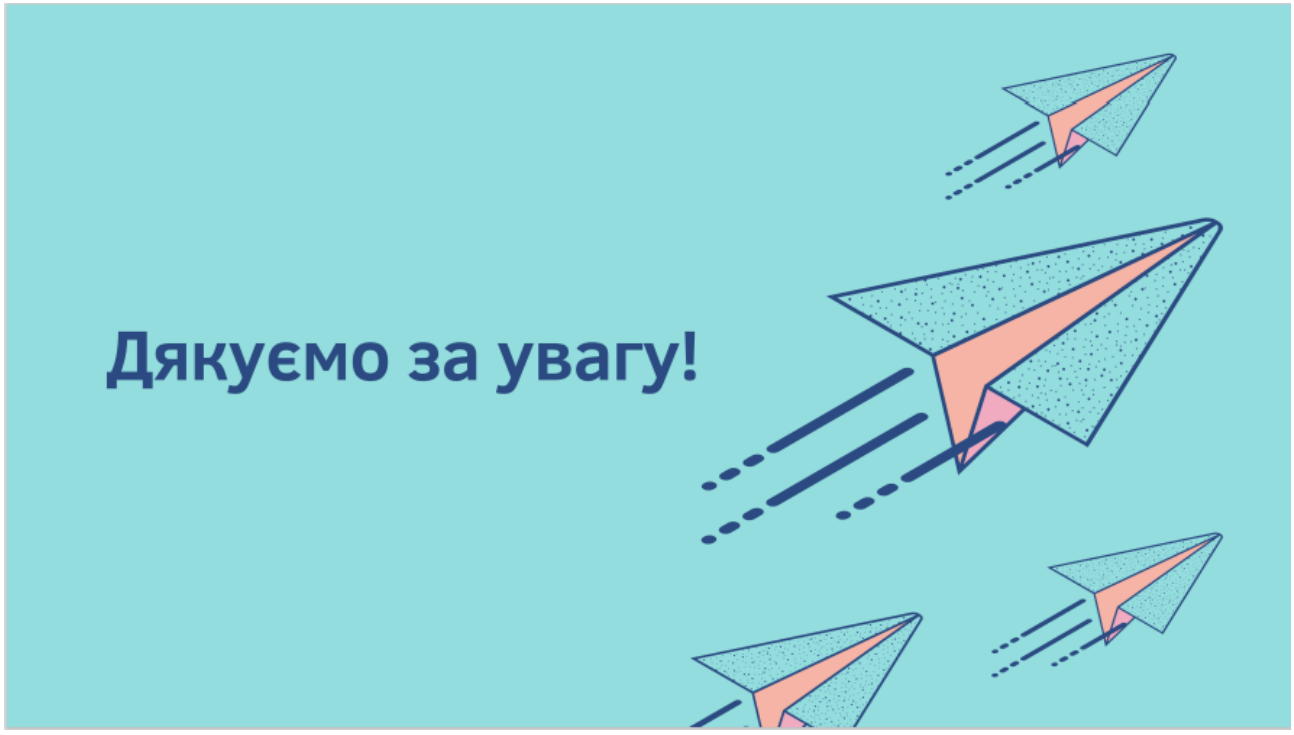
Головна мета

Отримання великої кількості аудиторії та числа користувачів, тому економічна ефективнісь у даного продукту може враховуватись лише у випадку монетизації додатку (поширення реклами)

NPU – чиста поточна вартість доходів за роки реалізації впровадження (3–5 років) складе 6000 грн

Висновки

- спроектовано та розроблено Androidдодаток з порівнянням хмарних та виділених серверів для обробки великого обсягу даних;
- виконано аналіз існуючих серверних технологій для розробки мобільних додатків,виявлення їх преваг та недоліків;
- досягнуто основної мети зробити інтерфейс зручним;
- проаналізовано економічну ефективність розроблюваної системи;
- проведено визначення трудомісткості розробки програмного забезпечення та розраховані витрати на створення програмного забезпечення;
- написана вся необхідна програмна документація.


Дякуємо за увагу!