Senior Design Project

Healthscope

Low Level Design Report

Cüneyt Erem
Doğukan Ömer Gür
Kaan Kale
Melih Sancak
Mert Gürcan

Advisor: A. Ercüment Çiçek
Jury Members: Selim Aksoy, Özcan Öztürk

19.06.2017
This report is submitted to the Department of Computer Engineering of Bilkent University in partial fulfilment of the requirements of the Senior Design Project course CS492

Content

1. Introduction

Healthscope is java based project, running on the Android platform, aims to make people more aware of impact of the products on their health. When customer do shopping, Healthscope identifies the product that customer like and tell him/her whether it is a healthy choice or not according to the customers' health status which is reflected as codes in our app. As a result, this project would help customers to fill their shopping carts with more healthy foods that improve their medical conditions. Our "Healthscope" application helps user to shop on supermarkets with aware of their medical conditions. Users create their own profile on this application. There are fields for their age, physical appearances (height, weight), diseases and allergies. And our "Healthscope" doctor evaluate every food and their ingredients in the supermarket for our users. There are five levels for every ingredients agreeableness (white, yellow, orange brown, red). It is very easy to use because users just read the barcodes of each food they want to buy with their androids, ingredients and agreeableness will appear on their screen and they can see appropriateness respect to their medical conditions.

This report aims to give a detailed explanation of the implementation of Healthscope by describing object design trade-offs, engineering standards, fundamental packages, and interfaces of important classes.

1.1. Object design trade-offs

Trade-offs we decide to consider;

- **User-Friendliness vs Functionality**

  We want our application to easy to use since our users will use them whenever they want to learn a product is good for them or not. We decide not to add additional functionalities to make our application more user friendly.

- **Compatibility vs Extensibility**

  We decided that our application supports at least Android 4.4 Kitkat or better versions of Android operating system since we want to use more recent technologies and UI components in our application but also, we want to reach as much people as possible. Android 4.4 and after versions correspond seventy percent of the Android users.

- **Security vs Cost**

  We take very detailed information about our users and store it in our servers and it's extremely important to keep the user information safe. We will use the latest protocols and technologies in our system to make sure user information are safe.

- **Performance vs Memory**

  We want our program has fast response times and smooth user experience however it's essential to have a high-performance application even if we hate to more resources of the device we are working on. We decided that we use more resource to increase performance and user experience.

1.2 Interface documentation guidelines

Class Name	Name of the class
Package Name	Name of the package which includes class
Description	Short description of class
Attributes	List of attributes of class
Operations	Methods which are included in class
Layout	If exists, name of layout of class

1.3 Engineering standards

We used Git to combine, update and share the code between contributors. Github.com is our interface of the repositories of the project. Also, Standard software engineering charts such as flow chart, use case, sequence are used in reports to document the overall functionality and architecture of the system.

2. Class Interfaces

Class Name	MainActivity.java
Package Name	src
Description	Main Activity before login screen
Attributes	
Operations	public void onCreate()
Layout	activity_main.xml

Class Name	Login.java
Package Name	
Description	shows login screen and handles the access
Attributes	String email String password
Operations	public void attemptLogin() public boolean authentication() public boolean isPassValid() public boolean isMailValid()
Layout	login.xml

Class Name	Register.java
Package Name	
Description	shows register screen and handles the new user input
Attributes	String email String password String name String surname Int age Int weight Int height Arraylist<boolean> illness: Contains ilnesses that will be used in analysis phase Boolean onDiet: Contains information whether user wants to lose weight. Arraylist<Boolean> allergies: Contains allergies that will be used in analysis phase. Button register
Operations	public void sendDatabase(User) public boolean isAvailablOnDatabase(User) public void onClick() public void onCreate()
Layout	register.xml

Class Name	User.java
Package Name	
Description	Contains user profile information
Attributes	String email String password String name String surname Int age Int weight Int height User profile Arraylist<boolean> illness: Contains ilnesses that will be used in analysis phase Boolean onDiet: Contains information whether user wants to lose weight. Arraylist<Boolean> allergies: Contains allergies that will be used in analysis phase. SharedPreferences userlocaldatabase
Opearions	Accessors and mutators for attributes public void storeUserData(User user)
Layout	

Class Name	MainPage.java
Package Name	
Description	Passing view before scanning barcode and updating profile or logout
Attributes	Button goToBarcodeMain Button goToProfileSettings Button logout
Operations	public void onCreate() public void onClick()
Layout	main_page.xml

Class Name	ProfileSettings.java
Package Name	
Description	
Attributes	User onlineUser Button goToMainPage
Operations	public void getUserData() public void updateUserData() public void updateDatabase( ) public void onCreate() public void onClick()
Layout	settings_profile.xml

Class Name	BarcodeMain.java
Package Name	com.google.android.gms.samples.vision.barcodereader
Description	Main activity demonstrating how to pass extra parameters to an activity that reads barcodes
Attributes	private CompoundButton autoFocus private CompoundButton useFlash private TextView statusMessage private TextView barcodeValue
Operations	protected void onCreate(Bundle savedInstanceState) public void onClick(View v) protected void onActivityResult(int requestCode, int resultCode, Intent data)
Layout	barcode_main.xml

Class Name	BarcodeTrackerFactory.java
Package Name	com.google.android.gms.samples.vision.barcodereader
Description	Factory for creating a tracker and associated graphic to be associated with a new barcode
Attributes	private GraphicOverlay<BarcodeGraphic> mGraphicOverlay
Operations	public Tracker<Barcode> create(Barcode barcode)
Layout	

Class Name	BarcodeGraphicTracker.java
Package Name	com.google.android.gms.samples.vision.barcodereader
Description	Generic tracker which is used for tracking or reading a barcode
Attributes	private GraphicOverlay<BarcodeGraphic> mOverlay private BarcodeGraphic mGraphic
Operations	public void onNewItem(int id, Barcode item) public void onUpdate(Detector.Detections<Barcode> detectionResults, Barcode item) public void onMissing(Detector.Detections<Barcode> detectionResults) public void onDone()
Layout	

Class Name	BarcodeGraphic.java
Package Name	com.google.android.gms.samples.vision.barcodereader
Description	Graphic instance for rendering barcode position, size, and ID within an associated graphic view
Attributes	private int mId private static final int COLOR_CHOICES[] private static int mCurrentColorIndex private Paint mRectPaint private Paint mTextPaint private volatile Barcode mBarcode
Operations	accessors and mutators void updateItem(Barcode barcode) public void draw(Canvas canvas)
Layout	

Class Name	BarcodeCaptureActivity.java
Package Name	com.google.android.gms.samples.vision.barcodereader
Description	This app detects barcodes and displays the value with the rear facing camera
Attributes	private static final String TAG public static final String AutoFocus public static final String BarcodeObject private CameraSource mCameraSource private CameraSourcePreview mPreview private GraphicOverlay<BarcodeGraphic> mGraphicOverlay
Operations	public void onCreate(Bundle icicle) private void requestCameraPermission() public boolean onTouchEvent(MotionEvent e) private void createCameraSource(boolean autoFocus, boolean useFlash) protected void onResume() protected void onPause() protected void onDestroy() public void onRequestPermissionsResult() private void startCameraSource() private boolean onTap(float rawX, float rawY) private class ScaleListener
Layout	barcode_capture.xml

Class Name	CameraSource.java
Package Name	com.google.android.gms.samples.vision.barcodereader.ui.camera
Description	Manages the camera in conjunction
Attributes	private Context mContext private final Object mCameraLock private Camera mCamera private int mRotation private Map<byte[], ByteBuffer> mBytesToByteBuffer = new HashMap<>()
Operations	public void release() public CameraSource start() public CameraSource start(SurfaceHolder surfaceHolder) public void stop() public void takePicture(ShutterCallback shutter, PictureCallback jpeg) public boolean setFocusMode(@FocusMode String mode) public void autoFocus(@Nullable AutoFocusCallback cb)
Layout	

Class Name	CameraSourcePreview.java
Package Name	com.google.android.gms.samples.vision.barcodereader.ui.camera
Description	Manages the camera view in conjunction
Attributes	private Context mContext private SurfaceView mSurfaceView private boolean mStartRequested private boolean mSurfaceAvailable private CameraSource mCameraSource private GraphicOverlay mOverlay
Operations	public CameraSourcePreview(Context context, AttributeSet attrs) public void start(CameraSource cameraSource) public void start(CameraSource cameraSource, GraphicOverlay overlay) public void stop() public void release() protected void onLayout(boolean changed, int left, int top, int right, int bottom) private boolean isPortraitMode()
Layout	

Class Name	GraphicOverlay<T extends GraphicOverlay.Graphic>
Package Name	com.google.android.gms.samples.vision.barcodereader.ui.camera
Description	A view which renders a series of custom graphics to be overlayed on top of an associated preview
Attributes	private final Object mLock private int mFacing private Set<T> mGraphics
Operations	public static abstract class Graphic public GraphicOverlay(Context context, AttributeSet attrs) public void clear() public void add(T graphic) public void remove(T graphic) public List<T> getGraphics() protected void onDraw(Canvas canvas)
Layout	

Class Name	AnalyzedProduct.java
Package Name	
Description	Analyzed result from the server for ingredients respect to user profiles will be shown.
Attributes	String barcodeID
Operations	public Product loadAnalyzedProduct(String barcodeID)
Layout	Suggestion.xml

Class Name	ProductService
Package Name	
Description	Class that keeps information,ingredients of products, set new products, load and send the ingredients of the products that came from camera.
Attributes	String barcodeID
Operations	public Product loadProduct(String barcodeID)
Layout	

Class Name	UserService
Package Name	
Description	Keeps all information of the user, load and send personal details of the user and set changes.
Attributes	User user String email String password
Operations	public void loginUser(String email, String password) public void registerUser(User user) public User loadUserProfile() public void sendUserProfile(User user)
Layout	

Class Name	AnalyzeIngredientService
Package Name	
Description	Analyze incoming product from the barcode scanner respect to user profile and send the result, suggestion to analyzedIngredient class.
Attributes	User user String barcodeID
Operations	public void analyzeProduct(User profile,String barcodeID) public void sendResult(User user, String barcodeID)
Layout	

4. References

https://developer.android.com/studio/intro/index.html/, Online, Accessed: 13.05.2017

https://dev.mysql.com/doc/, Online, Accessed: 13.05.2017

https://github.com/zxing/zxing/, Online, Accessed: 13.05.2017

https://ndb.nal.usda.gov/ndb/doc/index/, Online, Accessed: 12.05.2017

https://github.com/, Online, Accessed: 13.05.2017

https://msdn.microsoft.com/en-us/library/ee658117.aspx#ClientServerStyle, Online Accessed : 13.05.2017