

Android App Anatomy (应用结构)		
Manifest <code>/src/main/AndroidManifest.xml</code>	元信息、权限、组件	<code>/app/build.gradle</code> 模块级 (App依赖文件)
Source files <code>.java / .kt</code>		<code>/build.gradle</code> 顶层项目构建文件
Layout files <code>res/layout/*.xml</code>	UI 布局	类型 <code>dependencies</code> 部分说明:
Images <code>res/drawable/</code>	图标/图像资源	<code>implementation</code> 主项目运行时依赖 (UI库、网络库等)
String constants <code>res/values/strings.xml</code>	多语言字符串	<code>testImplementation</code> 单元测试相关依赖
Build script <code>build.gradle</code>	项目构建配置 minSDK / targetSDK、dependencies 版本、插件等	<code>androidTestImplementation</code> 仪器测试依赖 (如 Espresso)
		<code>compileOnly / api</code> 特定场景下使用的依赖控制选项
<code>AndroidManifest.xml</code>		<code>res/layout/*.xml</code>
组件声明 (Activities、Services、BroadcastReceivers)	<code><activity android:name=".MainActivity" /></code>	存放于 <code>res/layout/</code> 文件夹中
应用权限 (如网络、定位等)	<code><uses-permission android:name="android.permission.INTERNET"/></code>	描述界面中有哪些 UI 元素 (Widgets)，它们的属性、位置、行为
应用入口点定义 (含 MAIN 和 LAUNCHER intent-filter)	<code><intent-filter> <action android:name="android.intent.action.MAIN"/></code>	元素类型 (如 <code>Button</code> , <code>TextView</code>)
设置应用级别配置 (图标、主题、最低 API 等)	<code><application android:icon="@mipmap/ic_launcher" /></code>	布局尺寸、边距、排列方式 (如 <code>layout_width</code> , <code>layout_height</code> , <code>margin</code> , <code>gravity</code>)
		内容 (<code>text</code> , <code>src</code> , <code>hint</code> 等)
Android 应用由 结构化文件系统 + 框架驱动的生命周期 + MVVM 架构 组成; Android 不只是操作系统, 它还是:		
• <input checked="" type="checkbox"/> 一个框架 (Framework)		
• <input checked="" type="checkbox"/> 一个插件架构 (Pluggable Architecture)		
• <input checked="" type="checkbox"/> 一个软件生态系统 (Ecosystem)		
分类	描述	
插件 (Plugins)	增强用户体验的扩展模块, 可选安装。类似 microkernel 架构。	
框架 (Framework)	提供开发“骨架”, 只需扩展。好处是控制反转 Inversion of Control, IoC (如 <code>onClickListener</code>)	
生态系统 (Ecosystem)	由多个开发者共享的模块集合, 如 Jetpack、NPM、Maven。	
分层结构 (Layered Architecture)	Android Framework 架构	
Presentation	UI 展示层 (Activity、View)	
Application	控制与服务层 (App logic)	
Domain Logic	业务逻辑层 (模型/数据)	
Data Access	数据访问层 (数据库、网络)	
特点: 层与层之间只能“上下相连”, 不跨层通信	有利于维护与模块复用	
UX antipattern		
• <input checked="" type="checkbox"/> 名称	问题描述	正确做法
Misdirection	用户被引导点击非他们本意的选项, 例如: 消极操作按钮比积极操作更突出, 或表达方式故意绕弯	Use clear, neutral wording. Avoid double negatives and misleading design.
Roach Motel	用户很容易“进入”某个状态 (如订阅、注册), 却很难“退出”或取消 (如隐藏的取消按钮)	Provide clear and accessible exit options like "Unsubscribe", "Logout", or "Delete Account".
Bait and Switch	用户以为点击是某个功能 (如“播放歌曲”), 结果弹出广告或付款界面	Keep UI elements consistent with their intended function and avoid changing behavior unexpectedly.
Trick Question	通过文字游戏混淆用户选择, 如勾选项表述为否定句	Use straightforward, intuitive language aligned with user expectations.
Inconsistent UI Elements	同样的功能按钮使用了不同格式或样式, 容易误导用户	Ensure consistent styling and semantics across similar elements.
No Hierarchy of Actions	所有按钮颜色/大小相同, 用户不清楚哪个操作是主次	Use contrast, color, and placement to distinguish primary, secondary, and dangerous actions.
Bad Feedback / Error Messaging	表单错误提示太模糊, 或登录错误提示暴露过多信息 (安全风险)	Provide clear, helpful, and secure feedback without overexposing system behavior.
Contrast & Color Issues	图标、导航栏颜色对比不足, 影响可读性和可访问性	Ensure strong visual contrast; follow WCAG accessibility guidelines.
No KISS	KISS = Keep It Simple, Stupid; 界面信息过多、逻辑混乱	Follow KISS: Keep It Simple, Stupid. Reduce unnecessary clutter.
No Layout Testing	不同屏幕或设备下布局错乱	Use responsive design tools and test on multiple devices/resolutions.
No Undo	No way to reverse an action	Always provide a way to undo destructive actions or confirm them in advance.
• <input checked="" type="checkbox"/> Trick Question: 用户以为勾选是“订阅”, 实际上是“拒绝订阅”		
• <input checked="" type="checkbox"/> Roach Motel: 注册很容易, 但找不到退订按钮		
• <input checked="" type="checkbox"/> Bait and Switch: 商品标明“免费”, 点击却变成“付费”		
android.view.View		
ViewGroup	布局容器特点	适用场景
LinearLayout	元素按垂直或水平顺序依次排列, 可设置 <code>layout_weight</code> 控制空间分配	<input checked="" type="checkbox"/> “在此布局中, 元素逐一排列。”
RelativeLayout	元素相对父容器或其他元素位置排列	button 位置基于 <code>editText</code>
GridLayout / GridView	元素按网格排列, 支持横向或纵向滚动	<input checked="" type="checkbox"/> “宠物图库切换大图小图视图: 用 RecyclerView + Grid/List layout”
ConstraintLayout	高级布局, 支持任意方向约束, 适用于复杂 UI	更强控制、更少嵌套 Lab 示例使用
FrameLayout	所有子视图层叠, 显示在彼此之上	用于堆叠组件、浮动按钮 Dialog 内嵌内容常见
UI Design Tips		
• <input checked="" type="checkbox"/> 指南 (Guide)	建议 (Design Tip)	
元素需清晰可见	使用颜色对比、留白、字号大小使元素脱颖而出	Make elements clearly visible using contrast, whitespace, and size
减少用户输入	利用自动完成、语音输入、选择器、传感器等方式降低输入负担	Minimize user input using autocomplete, voice input, pickers, sensors
使用一目了然的图标或文字	图标 + 文本组合最易理解, 也更易传达意义	Use intuitive icons or icon+text combinations for clear communication
避免多个按钮混杂	每个界面尽量只提供 1~2 个主要按钮 (KISS 原则)	Avoid cluttering with too many buttons (Follow KISS: Keep It Simple, Stupid)
控制元素位置	把主要操作按钮放在右下角等用户拇指易触区域, 符合手势设计	Place key action buttons in reachable thumb zones (e.g., bottom-right corner)
Testing		
Software Testing	Strategies	特点
Big Bang Testing	大爆炸测试	一次性测试整个系统, 适合小项目; 缺点是难以定位错误。
Incremental Testing	增量测试	按模块逐步集成和测试, 易于定位错误, 开发中常用。
Top-down	自顶向下测试	使用 stub 替代底层模块, 早发现架构问题
Bottom-up	自底向上测试	使用 driver 替代高层模块, 适合已有底层实现
Step	测试流程总结	Testing Flow Summary:
1	明确测试目标 (功能 / 性能 / 用户行为)	Define test goals (functional, performance, user interaction)
2	编写测试脚本或设计测试场景	Write test scripts or design test cases/scenarios
3	执行测试 (手动或自动)	Execute tests (manual or automated)
4	收集结果并分析	Collect results and analyze
5	生成报告并进行回归测试	Generate reports and perform regression testing
测试类型	特点 / 区别	
Alpha Testing	<input checked="" type="checkbox"/> 在开发方或公司内部进行	测试人员是 QA 或开发人员 <input checked="" type="checkbox"/> 可访问源代码 <input checked="" type="checkbox"/> 发现大多数 bug、崩溃等问题
Beta Testing	<input checked="" type="checkbox"/> 在真实用户环境中进行 (非开发者) <input checked="" type="checkbox"/> 收集用户反馈和使用行为 <input checked="" type="checkbox"/> 通常是“封测”或“公开测试” <input checked="" type="checkbox"/> 评估真实可用性、接受度和用户体验	
White Box Testing	<input checked="" type="checkbox"/> 测试人员了解代码逻辑 <input checked="" type="checkbox"/> 测试语句覆盖、分支覆盖、条件覆盖等 <input checked="" type="checkbox"/> 开发人员常用, 配合单元测试	
Black Box Testing	<input checked="" type="checkbox"/> 测试人员只关注输入和输出, 不关心代码内部 <input checked="" type="checkbox"/> 用于 UI 测试、功能测试 <input checked="" type="checkbox"/> 真实模拟用户操作	
Grey Box Testing	<input checked="" type="checkbox"/> 测试人员了解部分内部结构 <input checked="" type="checkbox"/> 综合功能 + 部分结构性测试 (如接口层)	
Gestures	类型	类 / 工具
Basic Touch Gestures	基础手势: 点击、滑动、长按等	<code>GestureDetector</code> 单点触控手势
Pinch/Zoom	双指捏合、缩放	<code>ScaleGestureDetector</code> 多点触控手势 (如缩放地图)
Unistrokes / Graffiti	单笔迹输入 / 类似 Palm Graffiti 的笔迹输入	<code>Unistrokes.java</code> , 自定义手势库
Shape Writing	连续手写输入 (如 Swype)	通过形状与轨迹识别词语
GestureListener	手势监听接口	<code>SimpleOnGestureListener</code> 重写如 <code>onFling</code> , <code>onLongPress</code>
项目	Drivers (驱动模块)	Stubs (桩模块)
用于	测试“底层”模块是否能正确工作	测试“顶层”模块调用是否正常
作用	模拟上层模块, 向下传递数据并触发调用	模拟下层模块, 返回固定响应或数据
数据流	“向下”传递, 执行真正底层代码	“向上”模拟, 跳过底层实现
举例	测试数据库模块: 用 driver 模拟 UI 输入调用	测试 UI 模块: 用 stub 模拟数据库响应
TestCases	概念	Example
Equivalence Class	等价类	每个变量的有效/无效分类
Marginal Value	边界值	0, 3, 3.1, 6 小时数边界测试
Valid Test Cases	有效用例	TC1~TC3, 逻辑正常应通过
Invalid Test Cases	无效用例	TC7~TC9, 应抛出提示
Coverage Strategy	覆盖策略	每类至少测试一次: 覆盖所有输入组合类型
Touch	触摸事件	补充说明
Touch Mode	触摸模式 (Android 中的输入模式)	用户使用手指输入时, 不需要焦点选中 (Unlike D-Pad)
MotionEvent	触摸事件类	包含 <code>ACTION_DOWN</code> , <code>ACTION_MOVE</code> , <code>ACTION_UP</code> 等
Pointer	指针 (每根手指)	每个 pointer 有一个固定不变的 ID 和可能变化的索引 (index)
事件顺序	手指按下/移动/离开顺序	示例顺序: DOWN → POINTER_DOWN → MOVE → POINTER_UP → UP
ACTION_INDEX & ID	多点触控下的手指识别	<code>getActionIndex()</code> 获取变动手指索引, <code>getPointerId()</code> 获取其 ID
标识符 (ID)	<code>Pointer ID</code>	√ 按下时分配, 整个手指生命周期不变, 用于唯一标识
索引 (Index)	<code>Pointer Index</code>	! 在当前事件中的位置, 每次事件中可能不同, 用于访问坐标
Performance Testing	Types	特点 / 目标
Load Testing	负载测试	测试在“预期负载”下的系统表现 (正常使用压力)
Stress Testing	压力测试	测试系统超出最大承载时的表现 (宕机等)
Endurance Testing	耐久性测试	长时间高负载下的稳定性测试
Spike Testing	峰值测试	模拟瞬间涌入或减少用户, 检测系统弹性
Capacity Testing	容量测试	确定系统最大承载用户或任务量
UI Testing	测试类型	特点
Manual UI Testing		人工操作并比对结果
Automated UI Testing (Espresso)		使用代码模拟用户行为并验证结果
Record and Replay		录制回放测试
Scripted Testing		脚本测试
Exploratory Testing		探索式测试
UX Testing		用户体验测试
Design	Design is achieving goals within constraints.	
Goals: Who is it for? Why do they need it?	设计是在限制条件下实现目标的过程。	
Constraints: time, cost, platform, materials	限制: 时间、成本、平台、材料	
Design is about making trade-offs.	设计就是不断权衡取舍。	
维度	解释	E³ Framework
Effective	Gets the job done—for the user!	有效: 完成用户的任务!
Efficient	Minimal steps, clear, no distractions	高效: 最少的操作, 无干扰
Exciting	Fun, rewarding, emotionally engaging	有趣: 愉悦、有吸引力的用户体验
类型	Design vs Engineering vs Research	
Research	Understands why/how things work	研究: 探索问题本质与机制
Design	Creates solutions to meet goals	设计: 提出解决方案
Engineering	Builds working implementations	工程: 将方案落地实现
Trade-off		
Performance vs. Reusability	性能 vs 可重用性	
Cost vs. Robustness	成本 vs 稳定性	
Rapid Dev vs. Feature Complete	快速开发 vs 功能完备	
Backward Compatibility vs. Readability	向后兼容 vs 代码可读性	
原则		Mobile UX Best Practices
<input checked="" type="checkbox"/> KISS	Keep it simple, stupid!	保持简单, 一屏一事
<input checked="" type="checkbox"/> Cross-device	Seamless experience	跨设备一致性
<input checked="" type="checkbox"/> Clear Navigation	Intuitive and consistent	导航直观一致
<input checked="" type="checkbox"/> Less Input	Use autocomplete, sensors	减少输入, 调用自动填充或传感器
<input checked="" type="checkbox"/> Thumb Zone	Design for one-hand use	设计考虑拇指区域
Monitoring		
Profiling		
Instrumentation	插桩	精确度高, 适合短操作, 但会引入性能开销
Sampling	采样	精度较低, 但性能影响小, 适合长期采样
Espresso Feature		
精确匹配	可根据 <code>ID</code> 、 <code>Text</code> 、 <code>State</code> 查找 UI 元素	
自动断言	验证页面状态或组件值是否符合预期	
支持手势操作	支持滑动、点击、长按等用户操作	
意图校验 (Espresso-intents)	支持 Activity 切换与意图传递验证	
列表控件测试 (Espresso-lists)	支持 ListView / RecyclerView 等动态列表测试	
录制与回放测试 (Record and Replay)	自动记录交互生成测试代码, 可重复执行	
测试脚本可编程扩展	可以修改自动生成的测试代码	
支持断言与状态检查	使用 <code>assert</code> 检查视图文本、是否可见等	
支持多层 UI 层级定位	能深入 Activity → Layout → View 层级	
适配异步任务 (IdlingResource)	等待异步操作完成再继续测试	
阶段		
生产期		上线后用于追踪运行状态
开发期		开发阶段用于分析和优化性能
上线		监控与性能工程 (SPE)

手势类型	手势识别方法	动作常量	解释	触发条件	多点触控特性	实用 API/类汇总	用途
单指点击	<code>onSingleTapConfirmed()</code>	<code>ACTION_DOWN</code>	第一根手指按下	第一次触摸屏幕时触发	表示“开始触摸序列”	<code>MotionEvent</code>	获取触摸信息（位置、压力、多指等）
滑动 (Swipe)	<code>onFling()</code>	<code>ACTION_POINTER_DOWN</code>	其他手指按下	第二根或更多手指按下时	多点触控的额外入口点	<code>GestureDetector</code>	单指手势识别（滑动、点击）
长按 (Long Press)	<code>onLongPress()</code>	<code>ACTION_MOVE</code>	手指在屏幕上滑动	任意手指发生位移时都会触发	会持续回调	<code>ScaleGestureDetector</code>	多指手势识别（缩放）
拖动 (Drag)	<code>onScroll()</code>	<code>ACTION_POINTER_UP</code>	其他手指抬起	非主手指抬起	主手指未抬起时	<code>SensorManager</code>	管理各种传感器
双指缩放 (Pinch)	<code>onScale() in ScaleGestureDetector</code>	<code>ACTION_UP</code>	最后一根手指抬起	结束整个触摸事件流	与 <code>DOWN</code> 配对	<code>SensorEventListener</code>	监听传感器数值变化
		<code>ACTION_CANCEL</code>	事件被系统取消	系统中断，如来电	非人主动抬起	<code>Sensor</code>	表示单个传感器

类型	示例
<code>Motion Sensors</code>	运动传感器
<code>Position Sensors</code>	位置传感器
<code>Environment Sensors</code>	环境传感器

✓ 常用 Manifest 权限 (Manifest Permissions)

权限	中文说明	用途举例
<code>INTERNET</code>	网络权限	网络请求（如下单、API 调用）
<code>ACCESS_FINE_LOCATION</code>	精确定位权限	GPS 追踪宠物位置
<code>ACCESS_COARSE_LOCATION</code>	粗略定位权限	使用 Wi-Fi/基站定位
<code>ACCESS_BACKGROUND_LOCATION</code>	后台定位权限 (Android 10+)	App 在后台运行时继续定位
<code>WRITE_EXTERNAL_STORAGE</code>	写入外部存储	保存宠物照片到相册
<code>READ_EXTERNAL_STORAGE</code>	读取外部存储	加载本地照片
<code>FOREGROUND_SERVICE</code>	前台服务权限	持续定位（通知栏常驻）
<code>USE_BIOMETRIC</code>	生物识别权限	指纹/面部识别支付验证
<code>CAMERA</code>	相机权限	拍宠物照片或二维码
<code>BLUETOOTH / BLUETOOTH_ADMIN</code>	蓝牙权限	连接智能项圈或喂食器设备
<code>VIBRATE</code>	震动	通知提醒、交互反馈
<code>SCHEDULE_EXACT_ALARM</code>	精准定时	闹钟、定时通知 (Android 12+)
<code>READ_CONTACTS</code>	读取联系人	社交分享、邀请好友等
<code>RECORD_AUDIO</code>	麦克风权限	语音输入、语音识别

⌚ 常用监听器 & 辅助类 (Listeners and Helper Classes)

类名	中文说明	用途
<code>OnClickListener</code>	点击事件监听器	按钮点击（下单、添加宠物等）
<code>OnTouchListener</code>	触摸事件监听器	处理地图触摸、宠物图标点击
<code>ScaleGestureDetector</code>	缩放手势检测器	地图缩放、图片放大缩小
<code>GestureDetector</code>	手势识别器	单击、滑动、长按识别等
<code>SensorEventListener</code>	传感器监听器	连接硬件项圈、检测距离
<code>Adapter</code>	数据适配器	绑定宠物列表、订单数据等
<code>RecyclerView.Adapter</code>	高级适配器	高性能列表展示
<code>ViewHolder</code>	列表项缓存器	优化性能，复用列表项 View
<code>Handler/Runnable</code>	消息处理器 异步任务处理	异步任务或 UI 线程更新、延迟执行、消息机制
<code>Intent</code>	意图类	页面跳转，传递参数（如宠物ID）
<code>AlarmManager</code>	定时任务调度器	定时提醒、自动任务等
<code>MediaPlayer / ExoPlayer</code>	媒体播放辅助类	音视频播放、预览功能
<code>BroadcastReceiver</code>	广播接收器	接收系统广播或事件通知
<code>OnLongClickListener</code>	长按监听器	长按选择、删除等
<code>LocationListener</code>	位置监听器	GPS 实时定位追踪
<code>BroadcastReceiver</code>	广播接收器	接收系统广播或事件通知
<code>TextWatcher</code>	输入框监听	实时监听 EditText 内容变化

📦 Layouts 常用布局类型

布局名	中文说明	特点/适用场景
<code>LinearLayout</code>	线性布局	元素纵向或横向顺序排列
<code>RelativeLayout</code>	相对布局	元素相对其他元素对齐
<code>ConstraintLayout</code>	约束布局 ✓ (推荐)	扁平高效，适用于复杂 UI
<code>FrameLayout</code>	框架布局	层叠视图，常用于 Fragment 容器
<code>GridLayout</code>	网格布局	表格式控件排列，如计算器键盘
<code>ScrollView</code>	滚动视图	内容超出屏幕时可滑动（单子元素）
<code>NestedScrollView</code>	嵌套滚动容器	支持更复杂的嵌套滚动场景
<code>DrawerLayout</code>	侧滑菜单布局	主流 App 菜单结构
<code>CoordinatorLayout</code>	协调布局	与 FloatingActionButton、AppBar 等联动动画

- a. Activities:

 1. User Profile
 2. Pet Profile
 3. Map
 4. Order food
 5. Create profile

We expect at least the 4 first activities. -1 point for every activity missing (max -3 points).

Intent-filters for the following transitions:

User -> Pet
Pet -> Order
Create -> User
Pet -> Map

-1 point for every transition missing (max -3 points).

🧩 常用 UI 元素 (UI Elements)

元素	中文说明	用途
<code>TextView</code>	文本视图	显示宠物信息、标签等
<code>EditText</code>	输入框	输入宠物昵称、地址、数量等
<code>ImageView</code>	图像视图	显示宠物照片、地图图标等
<code>Button</code>	按钮	提交订单、切换界面
<code>FloatingActionButton</code>	悬浮按钮	快速添加宠物或跳转地图
<code>ListView / RecyclerView</code>	列表视图	展示宠物、订单、商品等信息
<code>Switch / ToggleButton</code>	开关	控制功能（如开启提醒）
<code>CardView</code>	卡片视图	美化布局（宠物信息卡）
<code>ConstraintLayout</code>	约束布局	灵活布局控制（如头像+文字）
<code>LinearLayout</code>	线性布局	垂直或水平排列元素
<code>FrameLayout</code>	框架布局	用于 Fragment 容器或覆盖图层
<code>ProgressBar</code>	进度条	显示加载、下单进度
<code>Toast / Snackbar</code>	吐司提示	下单成功、错误提示等
<code>AlertDialog</code>	弹窗对话框	确认下单、取消提醒
<code>TabLayout</code>	标签页	多个功能页切换（如宠物/订单/设置）
<code>ScrollView</code>	滚动容器	可滚内容页
<code>TabLayout + ViewPager2</code>	标签页 + 页面滑动	页面导航（宠物、订单、设置）
<code>SeekBar / Slider</code>	滑动条	调节音量、亮度、等级等
<code>CheckBox</code>	多选框	勾选偏好
<code>RadioButton + RadioGroup</code>	单选框	选择性别/类型等

◆ 输入变量及等价类 Equivalence Classes

变量	有效值 (Valid Classes)	无效值 (Invalid)	边界值 (Marginal Values)
家具类型	Beds, Cabinets, Drawers (+10) Desks, Dining sets (+5) Bookcases, Chairs (basic)	其他 (如 Sofa, Mirror)	无
安装时间	Morning (基本) Afternoon (+20)	其他时间 (如 "Evening")	无
工时	0-3h (基本) 3.1-6h (+10/h)	>6h, <0h	0, 3, 3.1, 6

◆ 测试用例设计 Test Cases

✓ 有效测试用例 (Valid):

1. Bookcase, Morning, 2 hours
2. Desk, Afternoon, 5 hours
3. Bed, Morning, 3 hours

✓ 边界测试用例 (Marginal):

4. Drawer, Morning, 0 hours
5. Dining Set, Morning, 3 hours
6. Cabinet, Afternoon, 3.1 hours
7. Chair, Afternoon, 6 hours

✓ 无效测试用例 (Invalid):

8. Sofa, Morning, 2 hours (invalid furniture)
9. Desk, "Evening", 2 hours (invalid time)
10. Bookcase, Morning, 6.5 hours (invalid duration)

Permissions

1. Internet (for online order)
2. Google Services (for Google Maps)
3. Local storage (for Pet photos)
4. GPS (for location)

-1 point for every transition missing (max -2 points).

Unless empty answer, award 2 points.

We expect to see at least the following mentioned:

OnTouchListener (for simple touch interactions with the lists)
ScaleGestureDetector (for zoom in and out of the map)
OnClickListener (for regular buttons)
Adapter (to handle list items)

We expect to see at least these 4 mentioned. For every class missing, -1 point (max -3 points)

Unless empty answer, award 2 points.

We need to have a list in the User profile for the pets, a button to add a pet and a menu to edit profile, sign out, settings and so on.

We can have an Activity with two fragments for the Pet profile: one with photo(s) of the pet at the top and another with the pet's information at the bottom. We also need buttons to locate the pet (that would open the Map Activity) and to order food (that would open the Order Activity).

We have 7 UI elements in total. For every element missing, -1 point (max -5 points). For every other element that is well justified, award 1 point (max +5 points).

Unless empty answer, award 2 points.

1. Small fonts in the title of the app; Increase font
2. Important action (Add event) included in the Action Bar menu; Add a button in the main Activity.
3. No information about what the colors in the calendar mean; Add a legend or a menu item.
4. The color of the Action Bar menu is not appropriate to distinguish it in the Action Bar; Change color
5. "Sign out" is an action that should not be as prominent; Put in the Action Bar menu.
6. Add meeting: icon is not intuitive to the action it represents; Change icon