# **Kotlin Travelling App Assessment Rubric**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **10-9** | **8-7** | **6-5** | **4-0** |
| **Functionality & Robustness** | Application thoroughly demonstrates functionality & robustness on the following:   * Application opens without file structure modification. * Application runs without code modification. * Text translation support. * Text-to-speech support. * Selection of key phrases. * Interactive quiz. * Localization support. * Exiting via a dialog fragment. * Google markers & map. * Light & dark mode. * Splash screen with image view widget & transition animation. * Adaptive launcher icon. * Bottom navigation view widget which navigates the user to the appropriate activities. * Privacy policy in a web view widget. * Visually attractive user interface. * Published to Google Play Store. * Application downloadable from Google Play Store. | Application mostly demonstrates functionality & robustness on the following:   * Application opens without file structure modification. * Application runs without code modification. * Text translation support. * Text-to-speech support. * Selection of key phrases. * Interactive quiz. * Localization support. * Exiting via a dialog fragment. * Google markers & map. * Light & dark mode. * Splash screen with image view widget & transition animation. * Adaptive launcher icon. * Bottom navigation view widget which navigates the user to the appropriate activities. * Privacy policy in a web view widget. * Visually attractive user interface. * Published to Google Play Store. * Application downloadable from Google Play Store. | Application demonstrates some functionality & robustness on the following:   * Application opens without file structure modification. * Application runs without code modification. * Text translation support. * Text-to-speech support. * Selection of key phrases. * Interactive quiz. * Localization support. * Exiting via a dialog fragment. * Google markers & map. * Light & dark mode. * Splash screen with image view widget & transition animation. * Adaptive launcher icon. * Bottom navigation view widget which navigates the user to the appropriate activities. * Privacy policy in a web view widget. * Visually attractive user interface. * Published to Google Play Store. * Application downloadable from Google Play Store. | Application does not or does not fully demonstrate functionality & robustness on the following:   * Application opens without file structure modification. * Application runs without code modification. * Text translation support. * Text-to-speech support. * Selection of key phrases. * Interactive quiz. * Localization support. * Exiting via a dialog fragment. * Google markers & map. * Light & dark mode. * Splash screen with image view widget & transition animation. * Adaptive launcher icon. * Bottom navigation view widget which navigates the user to the appropriate activities. * Privacy policy in a web view widget. * Visually attractive user interface. * Published to Google Play Store. * Application downloadable from Google Play Store. |
| **Code Elegance** | Kotlin & XML files thoroughly contain no magic numbers/strings & are stored in their appropriate XML files.  Application code thoroughly demonstrates code elegance on the following:   * Correct use of intermediate variables, i.e., no method calls as arguments. * Idiomatic use of control flow, data structures & other in-built functions. * Sufficient modularity, i.e., code adheres to various OO design principles. * Adhere to an OO architecture, i.e., classes, functions, concise naming & functions assigned to the correct classes. * Efficient algorithmic approach. | Most Kotlin & XML files contain no magic numbers/strings & are stored in their appropriate XML files.  Application code mostly demonstrates code elegance on the following:   * Correct use of intermediate variables, i.e., no method calls as arguments. * Idiomatic use of control flow, data structures & other in-built functions. * Sufficient modularity, i.e., code adheres to various OO design principles. * Adhere to an OO architecture, i.e., classes, functions, concise naming & functions assigned to the correct classes. * Efficient algorithmic approach. | Some Kotlin & XML files contain no magic numbers/strings & are stored in their appropriate XML files.  Application code demonstrates some code elegance on the following:   * Correct use of intermediate variables, i.e., no method calls as arguments. * Idiomatic use of control flow, data structures & other in-built functions. * Sufficient modularity, i.e., code adheres to various OO design principles. * Adhere to an OO architecture, i.e., classes, functions, concise naming & functions assigned to the correct classes. * Efficient algorithmic approach. | Kotlin & XML files contain frequent magic numbers/strings & are not or are not fully stored in their appropriate XML files.  Application code does not does not fully demonstrate code elegance on the following:   * Correct use of intermediate variables, i.e., no method calls as arguments. * Idiomatic use of control flow, data structures & other in-built functions. * Sufficient modularity, i.e., code adheres to various OO design principles. * Adhere to an OO architecture, i.e., classes, functions, concise naming & functions assigned to the correct classes. * Efficient algorithmic approach. |
| **Documentation & Git Usage** | README file thoroughly contains evidence of:   * Privacy policy discloses user information collected by the application. * Application wireframes sketched & reflect the final application. * Step-by-step user guide describes each activity in detail. * Application code commented with KDoc & generated with Dokka.   Git branches thoroughly named with convention & contain the correct code relating to the functional requirement.  Git commit messages thoroughly reflect the functional requirement changes. | README file mostly contains evidence of:   * Privacy policy discloses user information collected by the application. * Application wireframes sketched & reflect the final application. * Step-by-step user guide describes each activity in detail. * Application code commented with KDoc & generated with Dokka.   Git branches mostly named with convention & contain the correct code relating to the functional requirement.  Git commit messages mostly reflect the functional requirement changes. | README file briefly contains evidence of:   * Privacy policy discloses user information collected by the application. * Application wireframes sketched & reflect the final application. * Step-by-step user guide describes each activity in detail. * Application code commented with KDoc & generated with Dokka.   Some git branches named with convention & contain the correct code relating to the functional requirement.  Some git commit messages reflect the functional requirement changes. | README file does not or does not fully contain evidence of:   * Privacy policy discloses user information collected by the application. * Application wireframes sketched & reflect the final application. * Step-by-step user guide describes each activity in detail. * Application code commented with KDoc & generated with Dokka.   Git branches are not or are not fully named with convention & do not or do not fully contain the correct code relating to the functional requirement.  Git commit messages do not or do not fully reflect the functional requirement changes. |

# **Kotlin Travelling App Marking Cover Sheet**

Name:

Date:

Learner ID:

Assessor’s Name:

Assessor’s Signature:

|  |  |  |  |
| --- | --- | --- | --- |
| **Criteria** | **Out Of** | **Weighting** | **Final Result** |
| Functionality & Robustness | 10 | 40 |  |
| Code Elegance | 10 | 45 |  |
| Documentation & Git Usage | 10 | 15 |  |
| **Final Result** | | | /100 |
| **This assessment is worth 35% of the final mark for the Design and Development of Application Mobile Devices course.** | | | |

Feedback: