College of Engineering, Construction and Living Sciences
Bachelor of Information Technology
IN721: Design and Development of Applications for Mobile Devices
Level 7, Credits 15

React Native Hacker News App

Assessment Overview

For this assessment, you will develop & publish a Hacker News application using React Native in Visual Studio Code & Google Play Store. We won't be covering the basic features of React Native formally in class; you will be learning these features **independently**. The main purpose of this assessment is not just to build a simple application, rather demonstrate your ability to effectively learn a new technology which differs, both programmatically & syntactically from Kotlin. In addition, marks will be allocated for application robustness, code elegance, documentation & git usage.

The Hacker News application will help you keep up to date with the latest & greatest news in computer science & entrepreneurship. A user of your Hacker News application will be able to view & read the 100 top stories, best stories & job stories.

Assessment Table

Assessment Activity	**Weighting**	**Learning Outcomes**	**Assessment Grading Scheme**	**Completion Requirements**
Practicals	10%	1, 3, 4	CRA	Cumulative
Kotlin Travelling App	35%	1, 3, 4	CRA	Cumulative
React Native Hacker News App	25%	1, 3, 4	CRA	Cumulative
Kotlin Exam	15%	2, 3, 4	CRA	Cumulative
React Native Exam	15%	2, 3, 4	CRA	Cumulative

Conditions of Assessment

You will complete this assessment outside timetabled class time, however, there will be availability during the teaching sessions to discuss the requirements and progress of this assessment. Your application code must be

written in **JavaScript**. Submitted **TypeScript** code will not be accepted. This assessment will need to be completed by Wednesday, 18 November 2020 at 5pm.

Pass Criteria

This assessment is criterion-referenced with a cumulative pass mark of 50%.

Submission Details

You must submit your program files via **GitHub Classroom**. Here is the link to the repository you will use for your submission – https://classroom.github.com/a/5Ntput82.

Group Contribution

All git commit messages must identify which member(s) participated in the associated work session. Proportional contribution will be determined by inspection of the commit logs. If the commit logs show evidence of significantly uneven contribution proportion, the lecturer may choose to adjust the mark of the lesser contributor downward by an amount derived from the individual contributions.

Authenticity

All parts of your submitted assessment must be completely your work and any references must be cited appropriately including, externally-sourced graphic elements. All media must be royalty free (or legally purchased) for educational use. Failure to do this will result in a mark of zero.

Policy on Submissions, Extensions, Resubmissions & Resits

The school's process concerning **Submissions, Extensions, Resubmissions and Resits** complies with Otago Polytechnic policies. Students can view policies on the Otago Polytechnic website located at https://www.op.ac.nz/about-us/governance-and-management/policies.

Extensions

Please familiarise yourself with the assessment due date. If you need an extension, please contact your lecturer before the due date. If you require more than a week's extension, a medical certificate or support letter from your manager may be needed.

Resubmissions

Students may be requested to resubmit an assessment following a rework of part/s of the original assessment. Resubmissions are completed within a short time frame (usually no more than 5 working days) and usually must be completed within the timing of the course to which the assessment relates. Resubmissions will be available to students who have made a genuine attempt at the first assessment opportunity. The maximum grade awarded for resubmission will be C-.

Learning Outcomes

At the successful completion of this course, students will be able to:

1. Implement complete, non-trivial, industry-standard mobile applications following sound architectural and code-quality standards.

2. Explain relevant principles of human perception and cognition and their importance to software design.
3. Identify relevant use cases for a mobile computing scenario and incorporate them into an effective user experience design.
4. Follow industry standard software engineering practice in the design of mobile applications.

Instructions

Functionality & Robustness - Learning Outcomes 1, 3, 4

- Application must open without file structure modification in Visual Studio Code.
- Application must run without code modification on multiple mobile devices.
- Asynchronously fetch the first 100 top stories, best stories & job stories from Hacker News using Axios & the Hacker News API.
  - **Resources:**
    * https://www.npmjs.com/package/axios
    * https://github.com/HackerNews/API
- For each story, display its title & score from the response contents as an item in a flat list widget.
  - **Resource:** https://reactnative.dev/docs/flatlist.html
- When a story item is clicked/pressed, display its URL from the response contents in a web view widget.
  - **Resource:** https://www.npmjs.com/package/react-native-webview
- Bottom navigation widget which navigates the user to the appropriate collection of stories. You should have at least three menu icons, i.e., one for each story type.
  - **Resource:** https://www.npmjs.com/package/react-native-material-bottom-navigation
- Splash screen with an image view widget & transition animation.
  - **Resources:**
    * https://reactnative.dev/docs/image
    * https://reactnative.dev/docs/animations
- Adaptive launcher icon which displays a variety of shapes across different mobile devices.
  - **Resource:** https://romannurik.github.io/AndroidAssetStudio/icons-launcher.html
- Visually attractive user-interface with a coherent graphical theme & style.
- Application is published to Google Play Store.
  - When you create the application, please name the package appropriately, for example, **op.johndoe.hacker**. **Note:** replace **johndoe** with your Otago Polytechnic Ltd username.
  - **Resources:**
    * https://docs.expo.io/workflow/publishing
    * https://docs.expo.io/distribution/uploading-apps
- Ability to download the application from Google Play Store on to multiple mobile devices.

Documentation & Git Usage - Learning Outcomes 3, 4

- Provide the following in the repository README file:
  - How do you set up the environment for development, i.e., after the repository is cloned, what do I need to run the application?
  - Privacy policy which discloses user information collected by the application.
  - Sketched wireframes of the application. This can be hand-written or digitalised. **Note:** You must design you application before you start coding.
  - Step-by-step user guide detailing each screen. The user guide must contain a screenshot of each screen in the application.
  - Commented code is documented using JSDoc & generated to **Markdown**.

    * **Resources:** https://jsdoc.app

- At least 10 feature branches excluding the **main** branch.
  - Your branches must be prefix with **feature**, for example, **feature-<name of functional requirement>**.
  - For each branch, merge your own pull request to the **main** branch.
- Commit messages must reflect the context of each functional requirement change.
  - **Resource:** https://www.freecodecamp.org/news/writing-good-commit-messages-a-practical-guide

Assessment 02: React Native Hacker News 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. • Asynchronously fetch data from Hacker News using Axios & the Hacker News API. • Display the story's title & score from the response contents as an item in a flat list widget. • Display the story's URL from the response contents in a web view widget. • Bottom navigation widget which navigates the user to the appropriate collection of stories. • Splash screen with an image view widget & transition animation. • Adaptive launcher icon. • Visually attractive user-interface with a coherent graphical theme & style. • Application is published to Google Play Store. • Ability to download the application from Google Play Store on to multiple mobile devices.	Application mostly demonstrates functionality & robustness on the following: • Application opens without file structure modification. • Application runs without code modification. • Asynchronously fetch data from Hacker News using Axios & the Hacker News API. • Display its title & score from the response contents as an item in a flat list widget. • Display its URL from the response contents in a web view widget. • Bottom navigation widget which navigates the user to the appropriate collection of stories. • Splash screen with an image view widget & transition animation. • Adaptive launcher icon. • Visually attractive user-interface with a coherent graphical theme & style. • Application is published to Google Play Store. • Ability to download the application from Google Play Store on to multiple mobile devices.	Application demonstrates some functionality & robustness on the following: • Application opens without file structure modification. • Application runs without code modification. • Asynchronously fetch data from Hacker News using Axios & the Hacker News API. • Display its title & score from the response contents as an item in a flat list widget. • Display its URL from the response contents in a web view widget. • Bottom navigation widget which navigates the user to the appropriate collection of stories. • Splash screen with an image view widget & transition animation. • Adaptive launcher icon. • Visually attractive user-interface with a coherent graphical theme & style. • Application is published to Google Play Store. • Ability to download the application from Google Play Store on to multiple mobile devices.	Application does not or does not fully demonstrate functionality & robustness on the following: • Application opens without file structure modification. • Application runs without code modification. • Asynchronously fetch data from Hacker News using Axios & the Hacker News API. • Display its title & score from the response contents as an item in a flat list widget. • Display its URL from the response contents in a web view widget. • Bottom navigation widget which navigates the user to the appropriate collection of stories. • Splash screen with an image view widget & transition animation. • Adaptive launcher icon. • Visually attractive user-interface with a coherent graphical theme & style. • Application is published to Google Play Store. • Ability to download the application from Google Play Store on to multiple mobile devices.

Code Elegance	Application code thoroughly demonstrates code elegance on the following: • Idiomatic use of control flow, data structures & other in-built functions. • Sufficient modularity, i.e., UI split into independent, reusable pieces. • Handling of API response codes. • Header comments explain each function. • In-line comments explain complex logic. • Components adhere to composition & not inheritance. • Concise naming of components, functions & variables. • Efficient algorithmic approach. • Code formatted using Prettier or ESLint.	Application code mostly demonstrates code elegance on the following: • Idiomatic use of control flow, data structures & other in-built functions. • Sufficient modularity, i.e., UI split into independent, reusable pieces. • Handling of API response codes. • Header comments explain each function. • In-line comments explain complex logic. • Components adhere to composition & not inheritance. • Concise naming of components, functions & variables. • Efficient algorithmic approach. • Code formatted using Prettier or ESLint.	Application code demonstrates some code elegance on the following: • Idiomatic use of control flow, data structures & other in-built functions. • Sufficient modularity, i.e., UI split into independent, reusable pieces. • Handling of API response codes. • Header comments explain each function. • In-line comments explain complex logic. • Components adhere to composition & not inheritance. • Concise naming of components, functions & variables. • Efficient algorithmic approach. • Code formatted using Prettier or ESLint.	Application code does not does not fully demonstrate code elegance on the following: • Idiomatic use of control flow, data structures & other in-built functions. • Sufficient modularity, i.e., UI split into independent, reusable pieces. • Handling of API response codes. • Header comments explain each function. • In-line comments explain complex logic. • Components adhere to composition & not inheritance. • Concise naming of components, functions & variables. • Efficient algorithmic approach. • Code formatted using Prettier or ESLint.

Documentation & Git Usage	README thoroughly describes how to set the environment for development. Privacy policy thoroughly discloses user information collected by the application. Application wireframes thoroughly sketched & reflect the final application. Step-by-step user guide thoroughly describes each screen in detail. Application code thoroughly commented with JSDoc & generated to Markdown. 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 mostly describes how to set the environment for development. Privacy policy mostly discloses user information collected by the application. Application wireframes mostly sketched & reflect the final application. Step-by-step user guide mostly describes each screen in detail. Application code mostly commented with JSDoc & generated to Markdown. 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 briefly describes how to set the environment for development. Privacy policy briefly discloses user information collected by the application. Application wireframes briefly sketched & reflect the final application. Step-by-step user guide briefly describes each screen in detail. Some application code commented with JSDoc & generated to Markdown. 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 does not or does not fully describe how to set the environment for development. Privacy policy does not or does not fully disclose user information collected by the application. Application wireframes are not or are not fully sketched & do not or do not fully reflect the final application. Step-by-step guide does not or does not fully describe each screen in detail. Application code is not or not fully commented with JSDoc & generated to Markdown. 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.

Assessment 02: React Native Hacker News 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	50	
Documentation & Git Usage	10	10	
Final Result			/100
This assessment is worth 25% of the final mark for the Design and Development of Application Mobile Devices course.			

Feedback: