React Native Assessment Item

The **React Native Assessment Item** has a **20%** weighting. Important points to note:

- You will also need to do a **Code Review** on your **React Native Project** worth an additional **10%** weighting in your Week 13 Contact.
- If you have chosen to do a custom project, you will need to continue with your custom idea for your React Native Assessment item. For Custom projects you can't build the same app for Web and Mobile display. If you do a Custom project you need to consider both the Web and Mobile aspects.
- If you have done the standard React Web project (StoryPath), you will need to continue with the StoryPath Player (mobile app).
- The React Native project is intended to evaluate your proficiency in designing and developing mobile apps using React Native components.
- **Table 1** provides an overview of the Standard project and guidelines for Custom projects.
- **Table 2** provides the detailed requirements/rubric for the Standard React Native Project. Please make sure you carefully read them and feel free to ask questions.
- **You must come up with your own UI design for your project.** Screen grabs in the Getting Started Guide are just meant to explain how the app functionality works. Styles work differently in React Native. Bootstrap is not available but there is a library that supports some Tailwind (https://www.nativewind.dev/) and Shadcn/ui (https://rnr-docs.vercel.app/getting-started/introduction/). You are also able to create your own styles. You must use React Native components and can only use Webview and other HTML rendering components where specified in the Rubric.
- You can use open source React Native Components where detailed in Table 2. You can use other components as long as the components are not high level form building or API calling ones that won't allow you to learn the fundamentals of React Native. You are encouraged to explore and incorporate other React components but will need to read the documentation yourself. This is an important real-world skill to have.
- You are not building a full SaaS application – No login is required. For the Standard Project, the previously provided API has been extended to allow for participant tracking.
- You can use Typescript if you know how program with it.
- What happens if you don't complete the React Web Assessment Item? How will you be able to complete the React Native project? You will still have access to the API to add records that can be used in the React Native project. You can use a tool like Postman or HTTPie to add Projects and Locations to the API. You will also need to build API access into your React Native app.

Table 1: Available Project Options

<table>
<tr><th>Project Name & Description</th><th>Main Features of StoryPath Player React Native App</th></tr>
<tr><td><b>StoryPath</b>

StoryPath is a location experience platform designed to allow users to create and explore virtual museum exhibits, location-based tours, and treasure hunts with clues. The platform features a Web app built in React that enables users to author these experiences (React Web Assessment Item), and a React Native App called StoryPath Player for deploying them (React Native Assessment Item), making it easy to bring location-driven narratives to life.</td><td>- Allows participants to enter their name and upload a photo (Profile)
- Allows participants to view a list of published Projects and select one to participate in
- Allows participants to read instructions and follow clues to find Locations
- Scores participants based on either entering the radius of a Location or scanning a QR code
- Allows participants to view their location on a map and see the locations they have unlocked
- Allows participants to see their score and number of Locations they have unlocked
- Allows participants to view the number of participants in a project and the number of participants that have unlocked/visited each location</td></tr>
<tr><td colspan="2"><b>Custom Project Guidelines</b>

• A map must be included in the React Native Project which will complement the React Web Project.
• You will need to provide your own API to store and retrieve your data. You can use external services or deploy an API to your own UQ Cloud Zone. If you are already familiar with React, you might like to explore Next.js and build your front-end and back-end with it. The Teaching Team have used PostgREST (https://docs.postgrest.org/en/v12/) for the API and have provided a self-paced tutorial on making an API.</td></tr>
</table>

Code Submission:

You must submit a single **zip file** named 's1234567_Firstname_Lastname_ReactNative.zip' (replace with your student number) that includes your **source code folder(s)** and a Readme.md detailing where you used GenAI (e.g. ChatGPT or Claude Sonnet 3.5)**.** If you have developed a custom project, please ensure that API keys are provided, so that the Marker is able to test your code. Provide 1 example location experience to showcase your app.

Additional Questions:

If you have any questions about this assessment brief, you're welcome to post them on the course Ed Discussion and we'll get back to you soon.

A Message About Plagiarism:

⚠ **Plagiarism is considered a serious offence at UQ.** Failure to declare the distinction between your work and the work of others will result in academic misconduct proceedings.

- The use of Generative AI (i.e. ChatGPT, Claude Sonnet 3.5, Google Gemini, Microsoft Bing Chat, Github Copilot, Cursor) is allowed for this assessment item to assist you in designing your web application and learning new concepts. However, treat what you're producing here as a "trade secret" and don't share your code with other students. Also include details of where Generative AI has been used in a Readme.md file or withing your code comments.
- If you're inspired by design or code from online tutorials or any other external source, ensure you reference any inspirations for academic purposes (using APA/IEEE referencing styles) in an Readme.md file.

The Projects, Location and Tracking API endpoints for the StoryPath Standard Project:

A RESTful API is provided to provide Create, Read, Update and Delete access to the Project, Location and Tracking API endpoints. You will need to use the JWT provided via a Blackboard GradeCentre column: “A2 React Web JWT” – the same token applies for A3 React Native Web Assessment. The Tracking API has been added for the React Native Assessment Item.

Project: https://0b5ff8b0.uqcloud.net/api/project

Field Name	Data Type	Recommended Form Element	Description
id	INTEGER	None (Auto-generated)	A unique identifier for the project, automatically generated by the database.
title	VARCHAR(255)	Text Input (<input type="text">)	The name of the project, required for submission.
description	TEXT	Textarea (<textarea>)	A brief description of the project, not displayed to participants.
is_published	BOOLEAN	Checkbox (<input type="checkbox">)	Indicates whether the project is published. A checked box means the project is published.
participant_scoring	VARCHAR(100)	Dropdown (<select>)	Select how participants will be scored. Options include: "Not Scored", "Number of Scanned QR Codes", "Number of Locations Entered".
username	VARCHAR(255)	None (Auto-populated)	The username of the project creator. All projects you add need your student username. You don’t need to show this on the form. It is used to provide row level security to only allow you to retrieve what you submit.
instructions	TEXT	Textarea (<textarea>)	Instructions for participants, explaining how to engage with the project.
initial_clue	TEXT	Textarea (<textarea>)	The first clue to start the project. This field is optional.
homescreen_display	TEXT	Dropdown (<select>)	Choose what to display on the homescreen of the project. Options include: "Display initial clue", "Display all locations".

Location: https://0b5ff8b0.uqcloud.net/api/location

Field Name	Data Type	Form Element	Description
`location_name`	`VARCHAR(255)`	Text Input (`<input type="text">`)	The name of the location, required for submission.
`location_trigger`	`VARCHAR(50)`	Dropdown (`<select>`)	Select how this location content will be triggered. Options include: "Location Entry", "QR Code Scan", "Both Location Entry and QR Code Scan".
`location_position`	`VARCHAR(100)`	Text Input (`<input type="text">`)	Enter the latitude and longitude for this location. This is the format: (27.4975,153.013276)
`score_points`	`INTEGER`	Number Input (`<input type="number">`)	Specify the number of points participants earn by reaching this location, required for submission.
`clue`	`TEXT`	Textarea (`<textarea>`)	Enter the clue that leads to the next location. Entry is optional.
`location_content`	`TEXT`	Rich Text Editor (`<ReactQuill>`)	Provide additional content that will be displayed when participants reach this location. User’s can format text and add images. Images must be stored as base64 when the HTML is stored. You need to ensure the images are small in file size and in dimensions. You don’t need to provide an image editor, but when building sample content, you should use another editor to resize.

Tracking: https://0b5ff8b0.uqcloud.net/api/tracking

Field Name	Data Type	Recommended Form Element	Description
`id`	`INTEGER`	None (Auto-generated)	A unique identifier, automatically generated by the database.
`project_id`	`INTEGER`	None	Related field to the project.
`location_id`	`INTEGER`	None	Related field to the location.
`points`	`INTEGER`	None	Number of points the participant has received for entering a location or scanning a QR code at a Location.
`username`	`VARCHAR(255)`	None	The username of the student. This is needed to keep your data separate from other students.
`participant_username`	`VARCHAR(255)`	Text Input	The participants username, as entered on the Profile screen.

StoryPath Player React Native Core Functionality:

The StoryPath Player implements the functionality that was also implemented in the “Preview” feature of the React Web app. Within the react native app participants can view Location content (from the location_content field) and receive points once they are within the radius of a Location or have scanned the QR Code at a Location.

The StoryPath Player uses the project.instructions, project.participant_scoring, project.initial_clue, project.homescreen_display, location.score_points and location.clue fields to work out what is displayed on the Project Homescreen and the Location pages.

- The Project Homescreen needs to display a header with the Project title, project.instructions, the project.initial_clue if project.homescreen_display is set to "Display initial clue" or display all locations if "Display all locations" is selected.
- If a location.clue is not blank it should also be displayed. Use project.participant_scoring to work out how to score changing between Locations.
- All Locations including the Homescreen must display a Score count (depending upon project.participant_scoring) and Locations visited count.
- Visits and scores should be sent (only once) to the Tracking endpoint.
- Each participant should be able to access a Map which shows their current position and any Locations that they have unlocked. If all Locations are displayed by default (based on the Project Settings) then the Map should display Markers for all locations with a circle radius.
- Each participant should also be able to access an in app QR code scanner to scan QR codes. This functionality will allow a participant to scan a QR that is placed withing a Location and receive points.

The StoryPath Player also includes counts of participants for each project and at each location.

React Native Rubric

Table 2 includes the grade breakdown for each required feature. Note the assessment will be marked out of 100 but then scaled to be out of 20. “Custom” is used in the rubric to allow the rubric to also apply to custom projects.

Table 2: Project functionality and grade breakdown.

ID	Feature Description	Max Grade = 100
1	**Core Functionality**	**60 marks**
1.1	Package.json includes all dependencies, React Native code runs without errors, follows required folder naming convention and includes an example Project for grading.	5
1.2	Implement Expo Router with Tab and Drawer and Stacked navigation to allow effective navigation between the Projects List and Locations | Custom. You can either use Expo file based routing or React Navigation.	5
1.3	Implement a Profile screen that allows the participant to upload a photo and enter their participant_username. The participant_username must be displayed at the top of the Drawer and be used in the Tracking API. | Custom.	5
1.4	Implement a Project List Screen that displays a List of published Projects on the App homepage with the number of participants for each project. | Custom.	5
1.5	Implement the StoryPath Location Tracking, Location Content display and Scoring (Points and Location) functionality. The WebView component can be used to display the HTML content for a location (i.e. the location_content field). When location content is displayed, you should also include the number of previous participants that have unlocked/visited the location. | Custom. (You should be able to re-use the logic from your React Web projects Preview, but now need to use the Tracking API endpoint.)	25
1.6	Includes a Map that tracks the Participants current position and shows unlocked Location markers with a circle radius. The MapView component can be used with geolib and expo-location libraries. | Custom	10
1.7	Includes an in App QR Code Scanner that can score a user based on the QR code scan. | Custom.	5

2.	**User Interface Design**	**20 marks**
2.1	UI is visually appealing and uses consistent styling. Custom styles or a framework can be used.	15
2.2	UI is intuitive with all implemented features, easy to navigate and includes user instructions.	5
3	**Code Style and Quality** (Note: Max Grade will be capped at 10 grade points if only 50% of functionality is implemented)	**20 marks**
3.1	JSX and Javascript/Typescript adheres to Functional paradigm (i.e. avoids embedded imperative logic).	5
3.2	Breaks down functionality into well-defined, manageable react native components and centralised Fetch API access, reducing duplicate code in each component that uses the API. Uses API querystrings to query the required data instead of processing using Javascript/Typescript code.	5
3.4	Implements appropriate error handling for each component, ensuring robust error detection and management while displaying clear, user-friendly error messages that enhances user experience and facilitates troubleshooting.	6
3.5	Well formatted and documented code with inline comments and function doc strings included.	4