<https://github.com/JoeySuttonPreece/Challenge_Sem2_Assessment>

code: 83866

|  |  |
| --- | --- |
| **Section 1 – General Assessment Information** | |
| **Decision Making Rules** | Every task must be completed satisfactorily to be assessed as competent in the unit.  *\* For graded units, competence must be demonstrated before a mark can be given.* |
| **Reasonable Adjustment** | Students may request reasonable adjustment for assessment tasks.  Reasonable adjustment usually involves varying:   * the processes for conducting the assessment (eg: allowing additional time, varying the venue) * the evidence gathering techniques (eg: oral rather than written questioning, use of a scribe, modifications to equipment)   However, the evidence collected must allow the student to demonstrate all requirements of the unit. |
| **Special Consideration** | Students can apply for Special Consideration where personal circumstances have adversely affected their task result or ability to undertake an assessment. A Special Consideration form can be completed prior to, but no later than 3 days after, the date of the assessment and submitted to the relevant Manager. |
| **Re-submission** *(where tasks are not satisfactorily completed)* | Assessment tasks that are not satisfactory can be resubmitted up until the end of the unit as scheduled on the Unit Outline. The timing on this may depend on the equipment required for this assessment task.  **NOTE**: Assessment tasks submitted for the first time after the end of the unit as scheduled on the Unit Outline will not be assessed and student should be told to re-enrol in the unit. |
| **Plagiarism** | There are serious penalties for plagiarism. Students must ensure that all assessments are their own work (or group work).  Please refer to <https://www.swinburne.edu.au/current-students/manage-course/exams-results-assessment/plagiarism-academic-integrity/> |

| **Section 2 – Student and Assessor Instructions** | |
| --- | --- |
| **Conditions** | *Must be completed in class within the given time limit*  *Enterprise Architect must be used to create diagrams*  *Keep this document open as the assessor will provide feedback on the open document. Changes may be made to the submission based on feedback.*  *When the assessor has completed this assessment they will sign off with a code unique to you, do not leave until this code is entered in the assessor signature section.* |
| **Task Overview and/or Description** | **Diploma of Software Development – End of Semester Challenge**  **‘Basketball Team App’ Prototype**    The client is an (adult) basketball team who play together every week.  This team often have trouble keeping up with who paid the court fees for which game.  They want some software developed that will let them communicate when their games are scheduled and who has paid court fees for which game.  Ultimately the client would like a mobile app developed, but for now are happy for prototype / proof of concept software to be developed in/on ANY technology.  They want the software to do/enable the following and / or meet these requirements:    **Pass Requirements**   * A log in and sign up system with roles. * When the team manager (an authorised member) of the team learns the fixture details, they can enter the time, date and venue of games into the system. * Team members can see on the software the details of any future / upcoming (not past) games. * After a game, any team member can enter who paid the court fees and the dollar amount they spent. * If a game is forfeited a team member can delete the game from the system. * Have source code available on Github so future developers can access it easily.     **Credit Requirements**   * Team members can view a listing of past games and drill down to view who paid court fees for each game and how much they spent. * Team members can view a list of all members and the total amount they have spent on court fees (total for all time, no need to filter date period etc)   **Distinction Requirements**   * When a new player signs up, they are added to list of accounts pending approval. They cannot view game or payment details until their account has been approved by an existing team member. * Any team member can view a list of new player accounts pending approval, and either approve or reject them, in which case they move to authorised or rejected status accordingly     **HD Requirements**   * Use an external account log in to sign up or log in to the system (eg. Microsoft, Google or Facebook) * Create a Kanban with user stories and tasks for this project * Have at least one of the following documents for the software produced   + ERD and / or Class Diagram as appropriate. |
| **How the Assessment will be Conducted** | *Must be completed in class within a given three hour time limit*  *Assessment and feedback will be given immediately upon completion.* |
| **Submission Details** | 1. GitHub link 2. Marking Guide / Observation - below |

| **Section 3 – Assessment Criteria (Evidence to be Provided by the Student)** |
| --- |
| All of the required evidence within the task listed below must be satisfactorily demonstrated for the task to be assessed as satisfactory.  \* For graded units, the task must be satisfactorily completed before marks will be allocated. |

***Notes for the teacher***

*List in the Required Evidence column below all aspects of the task that are required to be demonstrated by the student for satisfactory completion of the task.*

*Name: Joe Sutton Preece*

*Student ID: 102568393*

*Date: 26/11/2020*

*Time: 4:57*

| **Marking Guide / Observation List** | | | | |
| --- | --- | --- | --- | --- |
| **Required Evidence** | | **Satisfactory** | **Not Submitted** | **Unsatisfactory** |
|  | ***Pass Requirements*** |  |  |  |
| **1** | New sign up |  |  |  |
| **2** | Access denied if user not in the system |  |  |  |
| **3** | Successful login |  |  |  |
| **4** | Add new game details as manager |  |  |  |
| **5** | View upcoming games |  |  |  |
| **6** | Enter fees paid |  |  |  |
| **7** | Delete game |  |  |  |
| **8** | Test Data |  |  |  |
|  |  |  |  |  |
|  | ***Credit Requirements*** |  |  |  |
| **1** | View past games |  |  |  |
| **2** | View fees paid from selected past game |  |  |  |
| **3** | View total fees paid by player |  |  |  |
|  |  |  |  |  |
|  | ***Distinction Requirements*** |  |  |  |
| **1** | New signups pending |  |  |  |
| **2** | Create facility for new player approval |  |  |  |
|  |  |  |  |  |
|  | ***High Distinction Requirements*** |  |  |  |
| **1** | OAuth |  |  |  |
| **2** | Kanban |  |  |  |
| **3** | Design Docs |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Q&A – Dot points (About 2-3 for each question)** | | | | |
| **Required Evidence** | | | **Satisfactory** | **Not Submitted** | **Unsatisfactory** | |
| **1** | | Describe the medium used to transfer data from API to Front-End |  |  |  | |
|  | | Firebase uses a combination of RESTful requests and websockets for realtime connections.  The requests are managed by the firestore system and provided inside the app through the firestore api. |  |  |  | |
| **2** | | Describe how testing was performed on this system. |  |  |  | |
|  | | The testing was performed manually by running through intended interactions and likely mistakes.  The app has not been rigorously tested for instability and the backend does not have fully secured endpoints. |  |  |  | |
| **3** | | How might this system be deployed in a live environment? (Automated deployments) |  |  |  | |
|  | | Github actions might be used to automatically build and deploy the app when a commit is made to a certain branch.  The app is currently and could continue to be deployed to the firebase hosting cdn. |  |  |  | |
| **4** | | Describe the third-party tools used for this build |  |  |  | |
|  | | This project is built on:   * Node.js * Typescript * Angular * Firebase   Using:   * Firefox * VS Code * Git |  |  |  | |
| **5** | | What may be some security issues with your prototype? |  |  |  | |
|  | | The database is not completely locked down, meaning an attack could submit bad data to some endpoints successfully.  The total tracking for member contributions currently relies on the client making two different requests which would allow an attack to corrupt the database state with a partial submission. |  |  |  | |
| **6** | | What other languages/frameworks could have been used to create this prototype? |  |  |  | |
|  | | An API could have been written in c# using .net 5.  A database could have been written in t-sql or sqlite.  The frontend could have been built with react, vue, hyperapp, or even raw js.  The frontend could also have taken the form of a native app built with WPF, flutter, or SwiftUI. |  |  |  | |
| **7** | | Describe the process used to create your data structures |  |  |  | |
|  | | The required fields were extracted from the brief then organised in a convenient shape for Firestore.  This meant using a document model and utilising the lack of strictly typed columns.  The main pattern that makes use of the document model is the payment field of the game object. It starts off with payment set to “pending” and once complete sets it to an object recording the payment details. |  |  |  | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Feedback to Student** | | | | | |
| Click or tap here to enter text. | | | | | |
| **Task Result** |  | **Satisfactory** |  | **Unsatisfactory** | **Grade:** |

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Name** | **Signature** | **Date** |
| **Assessor** |  |  |  |