

# SIT708

```
> /unitchair "Henry Larkin"  
> /topic "Mobile Systems Development"  
> /build mobile web apps
```



# SIT708

## Table of Contents

<b>Overview .....</b>	<b>3</b>
<b>Timetable.....</b>	<b>3</b>
<b>Weekly Blackboard - Ask Henry Anything.....</b>	<b>4</b>
<b>What You Can Expect from Me (Henry) .....</b>	<b>5</b>
<b>What I Expect from You .....</b>	<b>5</b>
Attitude.....	5
<b>How to Get Help .....</b>	<b>6</b>
How to Email Henry (template) .....	7
Banned List.....	7
Feedback to me.....	7
<b>Assessment .....</b>	<b>9</b>
<b>Assessment Rules &amp; Standards .....</b>	<b>9</b>
Effort: Demonstrate 14 hours per week.....	9
Marking Justification .....	9
University Policy on Late Submissions.....	10
Plagiarism Advice .....	10
Presentation.....	11
<b>#1: Project Plan .....</b>	<b>12</b>
Topic .....	12
Required Sections.....	13
Marking.....	13
<b>#2: Project.....</b>	<b>17</b>
Platform .....	17
Project Directory .....	17
Minimum Requirements.....	18
Marking.....	19
<b>#3: Portfolio .....</b>	<b>22</b>
Purpose.....	22
Required Sections.....	23
Marking.....	23
Submission.....	25
<b>Game Ideas .....</b>	<b>26</b>
<b>Sample Apps .....</b>	<b>27</b>

# Overview

---

Welcome!! I'm very happy you have chosen to enrol in SIT708.

This entire unit is ready for you now, with the information you'll find in Cloud Deakin. There are no lectures or other material to give, you have everything you need now.

This is an intensive unit. In just 10 weeks, I'll be pushing you to go from zero to making a complete mobile web app. That means each week you'll need to keep pushing ahead, making sure you not only cover all theory material and complete all lab exercises, but also write up your summaries and lab answers into your Portfolio document each week, **and** most importantly, be working on your Project each week.

At a minimum, you will need at least 14 hours of free time each week for this unit to pass. If you miss a week, that means 28 hours the following week, so keep that in mind. As there's only one unit to cover mobile apps in the Master's program, we need to cover as much as possible in these short 10 weeks.

## Timetable

---

There are no lectures in this unit. All theory content is online in the many videos I have now created for you. (You can safely ignore any Monday lecture class listed on your timetable).

We do have a weekly webinar in BlackBoard Collaborate where you can ask me questions and show me your work progress.

For Burwood campus students, we will also have physical lab times in HE1.021, which you are welcome to enrol in if you are on campus. These sessions will be run by our tutors. You should try to officially enrol in a lab session through the student timetable, but even if you don't have official enrolment in your preferred time, I don't mind if you come to other sessions, so long as there are free chairs available. You don't need my permission.

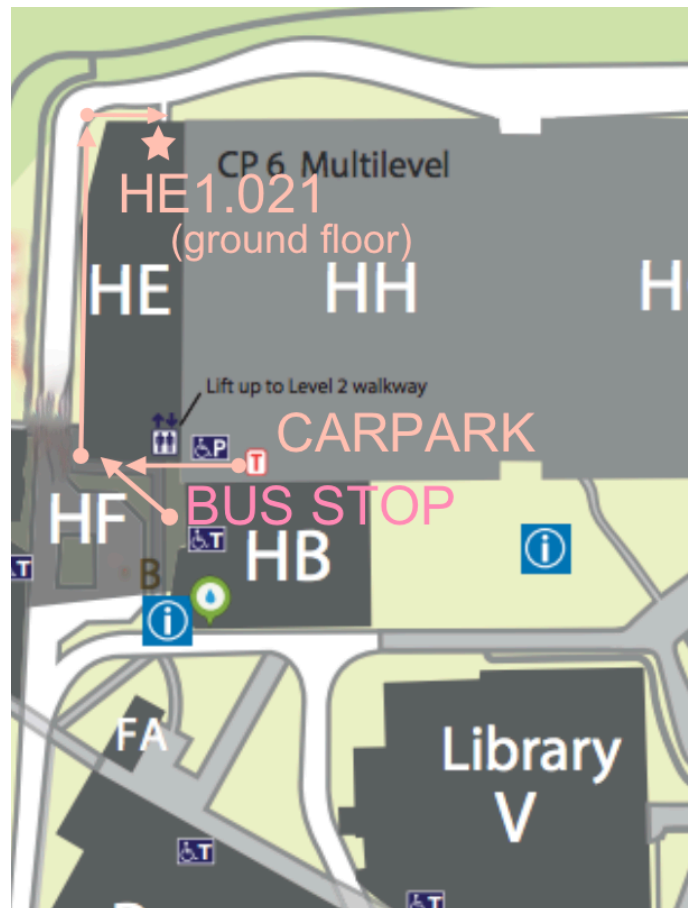
### Practical Labs:

- Thursday 9am - 3pm [HE1.021] (Tutor: Sachin)
- Friday 9am - 3pm [HE1.021] (Tutor: Vikrant)

### Free Workspace:

We also have free lab time on Wednesdays: 9am to 3pm. This time isn't for any tutorial sessions, and there is no tutor on Wednesdays. Wednesday lab times are just some booked lab time so that you can work on your assignments if you need a computer.

- Wednesday 9am - 3pm [HE1.021] (no tutor in room)



## Weekly Blackboard - Ask Henry Anything

For those who want to chat to me live, we'll have a weekly online session open to everyone (on-campus + cloud students).

- **Monday 4pm-5pm** (starting Week 1), every week of trimester.
- Access via Blackboard Collaborate link on our unit's Cloud Deakin site.

Mostly I'll use this session to give short reviews / ideas for your project. But you can ask me about lecture content, about labs, or about assignments and projects, about the weather, what foods you like to eat, where I like to travel - literally anything you want.

How to access them? Use the link for Blackboard Collaborate on our unit's Cloud Deakin site. Best viewed with Google Chrome.

## What You Can Expect from Me (Henry)

---

1. Henry will detail requirements and expectations for all assessment & learning objectives in this unit.
2. Henry will provide grade & feedback of assessment items within 7-14 calendar days of the due date of assessment.
3. Henry will reply to forums and emails every Monday.
4. Henry will be available for 1 hour each week online in the weekly Blackboard session.

## What I Expect from You

---

1. You will always make an attempt to find answers yourself first. You will accept that there are things you don't know, and won't let that frustrate you. I know that you don't know. I want to see you be able to search for answers, read through stackoverflow replies, and be able to sift through & identify the most probable solutions for your problem.
2. You will approach this unit with a genuine interest in achieving the learning goals. This isn't a unit you want to attempt for an easy 1 credit point.
3. You will timetable enough time to progress in this unit, a minimum of 10 hours per week, most likely 15 hours per week.
4. You will endeavour to keep a positive and friendly manner in all your interactions in this unit.

## Attitude

There are always times in life when things aren't what we expect, or when things are just plain and simply difficult, and we feel overwhelmed. I get it. In fact, I face that same situation all the time, as us academic staff are given a lot of tasks on very short notice all the time.

So, I do try to present all the information you need in order to complete this unit from day 1. Everything you need to know is available now.

That said, I can't give you "the answers" to everything, because there aren't really any. Learning how to do things is a process, a process that's different for each student, and a process you only develop by doing it yourself with your own hands. It is amending & enhancing your current neurological, conceptual maps & understandings of this field of programming. Everyone's existing knowledge base is different, and that's totally normal. So, what you spend time on solving will likely be different from other students.

It is natural to feel overwhelmed and frustrated when things don't work. This happens a lot, whenever you develop with new tools, or with tools that have bugs, or with many many software libraries with poor or zero documentation and undocumented features.

This unit is partially designed to prepare you for this reality, to have you face difficult problems that you don't have answers for, and allow you to try resolving them independently.

So, I want to see you at least attempt, and usually overcome, each & every issue you face, while maintaining your attitude & composure. In a nutshell, stay relaxed, breathe, take a step back, and let your brain do its job of brainstorming possible approaches to finding a solution.

I can, and will, give you as many hints as you ask for. But I won't be giving you any answers step-by-step, as there won't be someone to do that in your future careers when you face new problems each day.

## How to Get Help

Here's how to get help:

How to Get Help	
Email Henry	<ul style="list-style-type: none"> <li>• Personal issues</li> <li>• Extensions</li> <li>• Questions you don't feel comfortable asking online</li> </ul> <p>I'll generally reply within 5 business days (on Mondays). If longer than 5 business days, email again, as your email either got lost or couldn't be understood.</p>
Forums	<ul style="list-style-type: none"> <li>• Code questions</li> <li>• Unit query questions</li> <li>• Hints / advice to other students</li> </ul> <p>I'll reply to forums on Mondays.</p>
Webinar / Blackboard (Monday 4pm)	<ul style="list-style-type: none"> <li>• General chit-chat</li> <li>• Ask-Henry-anything</li> <li>• Questions on your project.</li> <li>• Quick reviews / short feedback of your work.</li> </ul>
Burwood labs	<ul style="list-style-type: none"> <li>• iMac machines to work with</li> </ul>

Unfortunately, I can't do phone calls or office meetings, as the School of IT has moved to Open Plan space and we're frowned upon for making noise. So, you won't see me much, except for during the Blackboard webinar sessions.

## How to Email Henry (template)

1. **Subject:** SIT708 (Burwood / Cloud)
2. **From:** (your deakin email address only)
3. **Email Body:**
  - a. Greeting / comment / short intro (< 5 sentences)
  - b. Numbered list of specific, short-answer questions.
  - c. Attachments (screenshots of issue / code), numbered to match questions (e.g. Q1.jpg, Q2-code.jpg, Q2-issue-with-popup.png, Q3-code.png, etc.)

Please always use this template. I teach multiple units to hundreds of students, so it's a big help to me if I know who you are. I also have limited time to answer emails, so I'll often check emails on my phone between meetings or while on the train. So, don't give me links to Microsoft OneDrive or anything that I won't be able to open on mobile ☺ .

Also, no non-Deakin email addresses. Madman007@hotmail.com, I still don't know who you are, and that's why I haven't replied in all these years.

## Banned List

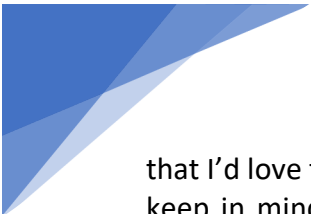
This title sounds a little bit negative doesn't it. I don't want you to be alarmed. But I do want to help you know what emails/forum posts/comments will get responses, and which ones will be deleted. While I want to support your learning as best I can, I will delete and won't reply to any of the following found in emails / forum posts:

1. Negativity, defeatism, frustration turned to anger.
2. Emails/Posts ignoring the instructions given above.
3. Emails/Posts that the answer could be found in under a minute using Google. E.g. if you see "colour theory" as a requirement and you don't know what it is... google it. Your first avenue for help should be at least 5 minutes on Google. Then forums.
4. Emails/Posts that aren't written in proper English. It's wise to write your email/post in MS Word first, then you can check spelling & grammar with MS Word tools. Even if I can understand the writing, if someone hasn't invested enough time to even spell my name correctly, or to capitalize the first letter of a sentence, I won't exactly feel like investing any time with a reply.

I hope the above list is fairly self-explanatory.

## Feedback to me

While I do appreciate feedback on the unit (emailed), I hope you will understand that academic staff (e.g. myself) usually get ridiculously short hours assigned to tasks (like "4 hours for all emails in Trimester 1, 1 hour to prepare each lab"), which just aren't physically possible. So, each unit you take at Deakin is mostly a "best effort, given the time limits imposed on us". There is so much more I'd like to put in this unit, further explanation documents and resources



that I'd love to write, but I'm never given the time to do them. So, while I do want feedback, keep in mind that complaints alone don't help, as I don't have the power to change the situation.

I'll be asking for general ideas to improve the unit during Week 11, which is the best time to give me thoughts about the unit.



# Assessment

There are 3 items of assessment that you need to complete in this unit:

<b>1</b>	Project Plan	30%	<b>Week 2</b> , Friday 5pm
<b>2</b>	Project	40%	<b>Week 10</b> , Friday 5pm
<b>3</b>	Portfolio / Unit Summary	30%	<b>Week 9</b> , Friday 5pm

Assessment is always due on a Friday at 5pm, so that I can potentially (maybe) answer any last-minute issues with server uploading (although I recommend you always strive to submit earlier than the final day). Unfortunately, I won't be available on weekends and at night time to help anyone, so due dates can't be on weekends nor at night time.

Note: The Portfolio (Unit Summary Report) must be updated (re-submitted) every week. Also note that this is due in Week 9 (a week before Project), because I want to know that you've studied and internalized each concept before you finish your project.

## Assessment Rules & Standards

### Effort: Demonstrate 14 hours per week

The university expects 150 hours of effort to be demonstrated throughout the semester for an average (pass / credit) grade.

This works out at roughly 14 hours per week for a unit without exams like this one.

Given that this unit is entirely focused on assessment (even the theory + lab exercises are assessed – in the Portfolio), that means you're expected to demonstrate 14 hours of work per week through your assessment.

Please plan & schedule time each week accordingly, as we will be looking for the “quantity of your learning” you demonstrate in all assessment items.

### Marking Justification

This year I want to try something different with you. Rather than have you submitting your work unsure of what grade you'll get, I want you to prepare and submit a Marking Justification sheet for each assessment item to explain the grade you want. In this short document, you'll give details of what grade you think you can justify (using the rubrics in this document), along with evidence to support your justification (as screenshots and/or brief bullet points to explain).

For screenshots, they should be at least 800px (width or height, depending on orientation), but image file sizes no larger than 300kb each before you drag & drop them into MS Word.

This way they'll be clear enough for me to read, but not so large that your Word Doc ends up over 10mb (after that size, MS Word starts to become buggy).

This will also help me to “find” features that you are particularly proud of, that you want me to be aware of (as in the short time I have to mark each student, I will inevitably miss things as I go through your code). So, this document will help me find what you want to be graded for.

## University Policy on Late Submissions

The university has an automatic policy on late submissions, starting at 1 minute late after due date & time with a 5% penalty, and penalizing an additional 5% of your grade every 24 hours (including weekend days). After 5 days, a grade of 0 is assigned.

If you do have “significant, completely unpredictable” circumstances, you must apply (via email to me) before the due date (e.g. at least 1 week before due date), along with accompanying evidence (e.g. doctor certificate / copy of police report / etc.), so that any extension I grant I can then justify to the academic board if I'm required to. Extensions on the last 3 days towards the due date will not be granted, as any “completely unpredictable” problem shouldn't matter at that stage, you can submit what you've done so far and get marks. So, plan ahead appropriately.

You can make repeated submissions to Cloud Deakin for all assessment items in this unit, so you are welcome to upload drafts / work-to-date (e.g. once a week) as a form of backup (highly recommended).

## Plagiarism Advice

All work must be your own from your own mind, created new for this trimester. You cannot use any previous assignments you've done in the past. You cannot use any projects you've worked on before, due to Deakin's policy that I assess new work you are creating. Everything must be your own work, and must be new work, created during this trimester. If you've done a project before that you want to continue, that's great for your experience, but you do that in your own time. This unit is about going through the entire app development life cycle, from start to finish. So, this means having fresh ideas.

You're allowed to use code snippets from other sources (e.g. stackoverflow), but you must attribute (credit) them before the block of code. E.g.

```
/* This idea to sort in reverse is from a user zimbaba at URL_HERE */
... code ...
/* end zimbaba's idea*/
```

Likewise, you're allowed to use images / sounds that are licensed for **commercial reuse with modifications**, or images / sounds that are in the **public domain**, provided you list them in the app in an About page under a section called Legal, as well as in a separate licenses.txt file, to credit everything you use.



## Presentation

Please make your work **look** amazing. The look (and feel) of any document / mobile app has been proven again and again to be the biggest influencer in success. If your work looks good, statistically you are far more likely to get a higher paying job as well. Always use MS Word styles for document consistency. Always apply Colour Theory to your app colour choices.

# #1: Project Plan

Critical Information Summary	
Weight	30%
Due	Friday Week 2, 5pm
Individual / Group	Individual
Submission via	Cloud Deakin Dropbox (multiple submissions allowed)
Required Files	2000 – 4500+ word report in a HTML file.  Zip file (SIDxxxxxxx-A1.zip), containing: <ol style="list-style-type: none"> <li>1. <i>SIDxxxxxxx-Marking Justification.docx</i></li> <li>2. <i>SIDxxxxxxx-A1.html</i></li> <li>3. <i>SIDxxxxxxx-A1.css</i></li> </ol>
Topic	See below.

You are to create an app proposal document in preparation of your Project, detailing an idea for an app, its justification, and an appraisal of similar apps within the app store. This is an individual student assignment.

This document does **not** have to be strictly adhered to for your Project. In fact, you can change your Project completely if you wish (a completely different direction / features / even topic is acceptable). But this document should be as accurate as possible, to help you move quickly through the Project. Without a step-by-step plan on what to do each day you work on your Project, you'll feel lost and unsure where to start or what to do next.

## Topic

The topic of your project (and thus also the topic of this Project Plan), is an educational game / brain-training game (see section “Game Ideas” at the end of this document for examples). You want to pick a problem-solving / brain-training idea that interests you.

The idea you pick cannot be the same as any other student you know (you want to completely avoid any similar code to anyone you know, so the type of game, not just the contents, need to be completely different).

Note that it's important to narrow down what kind of game and/or subject area you feel like creating first. Then, you should look at the competitor analysis, to see what similar games are in your specific area, to see if you are still comfortable with your chosen area. Then you can write the rest of your Project Plan. So: (1) pick a topic, (2) competitor analysis, and repeat step 1 if necessary, (3) write the Project Plan.

## Required Sections

This Project Plan is worth 30% of your grade, so the requirements are quite substantial. You would be expected to spend at least 30 hours researching & writing this plan.

1. **Marking Justification:** A cover page detailing the grade you are aiming for, and evidence for each individual rubric.
2. **Overview:** A brief, well-explained overview of the app you'll develop.
3. **Background:** Explanation of the problem / need for your app. "Why" should this app be created, what's the justification?
4. **Competitor Analysis:** An analysis of approx. 10 competitors, either mobile app, website, or traditional brick & mortar (physical business) competitors. Generally, a minimum of 150 words per review. It is up to you to decide what you think is relevant, but I would expect a minimum of: app name, app price / monetisation, core features, most common good comments (summarised), most common bad comments (summarised), and a screenshot of the most common user interface within the app (so you know what the look & feel is, what the colour themes are, etc.).
5. **Features:** A feature list of "must have" and "would like" features that your app may include. For an app of this level, assume each customer will pay \$30 for your app. What does your app need to contain, and how professional does it need to look & feel, for you to justify a \$30 asking price?
6. **Milestones:** A list of milestones, ordered and structured extremely well, such that your natural progression in your app is highly likely to follow the milestone order exactly.
7. **Design:** Wireframe / simple mockup of the visual screens your app will have.
8. **Data:** A top-level overview of the data tables / structures you will store in your app.
9. **API:** A top-level list of classes & significant functions/method signatures, that your app will need to be created.
10. **Resources Required (optional):** A list of any external images / sounds you plan to source.

## Marking

To achieve a particular grade, you must meet **all** criteria for that grade, as per the table below. Each higher grade requires **all** features of the previous grade as well. For example, if you want a Credit, then you need to meet everything down the credit column, **and** everything down the Pass column.


Criteria	Pass	Credit	Distinction	HD
<b>Presentation</b>	Your doc has consistent styling, title page, headers/footers, heading styles, and has been spell	Your doc also uses Styles for all main formatting.	Your doc also has consistent colours (using colour theory).	Your work easily passes for a professional pitch, where professionals and venture capitalists would not be able

	checked & grammar checked.			to tell the difference.
<b>Rich text</b>	You write in mostly plain text.	You include some bullets / lists / tables / images to explain your ideas.	Your doc has a good mix of text, rich text, and images.	Your doc uses Style padding and margins to maximise clarity through white space.
<b>Explanation</b>	You have accurate spelling & grammar, have appropriate paragraph structure.	Your ideas are easy to understand and are explained well, such that you could give your doc to any generic IT student, and they would understand.	Your ideas are both easy to understand (E.g. ELI5), while simultaneously cover enough detail for a team of developers to take your idea and develop it.	Your work is comfortable & engaging to read from a layperson (ELI5 / your parents), technical programmer team, and non-IT-background investor.
<b>Competitor Analysis</b>	5 competitors, each with a minimum 150 words, identifying the strengths and weaknesses of each.	7 competitors, each also with a relevant in-app screenshot.	10 competitors, including at least 2 non-app competitors. Further, each competitor also contains a list of most common occurring feedback from users (summarised).	An additional sub-section identifying, summarising & highlighting the critical features and critically missing features from all competitors, that form the basis of your own feature list.
<b>Features</b>	A list of at least 3 primary (must-have) and 10 secondary (future) features.	Each feature also has a sentence explaining why it is primary/secondary, and has an estimated duration in man hours.	Each feature also relates back to Competitor Analysis, especially user feedback of competitor products.	Each feature also includes details of how that feature will operate, especially in comparison to competitors. You may also include screenshots / sketches if appropriate.
<b>Milestones</b>	A list of milestones you plan on making, and the estimated man-hour durations multiplied by two.	A supplementary sub-section explaining the dependencies of each milestone.	Identification of all milestones capable of being used-tested [release], so long as it is justifiable (based on the dependencies).	You also identify a breakdown of most tasks for each milestone (can be a simple sub bullet point list)
<b>Design</b>	At least 1 wireframe of the main interface.	Wireframes of all screens planned for your app.	You have also selected a colour palette (listed as	You include a sub-section identifying a bullet list of most

			RGB values), using any colour theory, and display the colours on one of your wireframes.	common user actions, and the number of taps to achieve the result. (e.g. Button 1 -> Action 4 -> Scroll)
Data		You briefly explain the types of data you'll use, and a bullet list of the major variables.	You include a table for each major data object, outlining their structure.	You include examples for each major data object.
API		You include a list of all classes that should be developed for your primary features, and what their purpose will be.	For each class, you outline data fields, as well as any substantial functions/methods as method signatures.	For each substantial function/method, you briefly explain where it will be used / called from, how it will work, followed by brief pseudo-code comments. You will also include examples of how to call the function/method.

Additional marking explanations:

- **Explanation:**
  - Paragraph structure means: topic sentence, supporting sentences, conclusion sentence.
  - ELI5 means “Explain Like I’m 5 (years old)”, as in, imagine you are explaining an introductory sentence or paragraph of a section to a 5-year-old child. How would you use language, analogies, and generally express your idea so that a 5-year-old would understand the gist of your point?
- **Milestones:**
  - Pass: In IT in general, it’s usual to double the estimate of hours required, as we always underestimate how long it takes to overcome unforeseen bugs.
  - Credit: Dependencies of a milestone are the previous milestones (and any additional smaller factors such as software installs, API access, etc.) that need to be completed before this milestone can be begun.
  - Distinction: For user-testing, write “[release]” in a milestone title if it can be released for user testing. This means that the milestones to this point must leave the app in a workable state. This is a good habit to get into when planning, such that if you run out of time, you can submit the last “[release]” of your app and end-users can buy it and use it without issue. This must be justifiable. E.g. you couldn’t write “[release]” on a server-side script, or a client that needs a server and doesn’t yet have server connectivity, as the app wouldn’t “work” in any realistic sense to the end-user. So, this isn’t simply a matter of writing “[release]” anywhere, but requires you to think about the shortest number of milestones necessary to achieve each



“[release]” copy. To achieve a Distinction, I must agree that the “[release]” items are realistic and appropriate to an end-user getting valid use out of the app at that release point.

- **Data & API:** You can achieve a pass without these two items.
- **API:**
  - A function / method signature is the name + parameters + return type. E.g. “int[] fetchDataPointsFromNetwork(String url)”
  - Examples of calling a function/method are things like:
    - fetchDataPointsFromNetwork(“http://introtoapps.com/data1”)
    - fetchDataPointsFromNetwork(“introtoapps.com/data1”)
    - fetchDataPointsFromNetwork(mainUrls[0])

Note that, much against my beliefs, the university does enforce a quota of High Distinctions issued, and generally won’t accept more than 20% of a cohort from receiving an overall HD grade in the unit. There’s very little I can do about this, except try to make the HD criteria restrictive to ensure that we remain just under the quota.



## #2: Project

Critical Information Summary	
Weight	50%
Due	Friday Week 10, 5pm
Individual / Group	Individual
Submission via	Cloud Deakin Dropbox <b>Rolling</b> submissions required (re-upload your latest zip file every week to Cloud Deakin).
Required Files	Zip file <i>Filename: SIDxxxxxxxx-A2.zip</i>

You will individually create a fully complete mobile game using the mobile web design skills shown in class (from the theory + lab content). You will submit your completed application's project folder as a zip file. You also need to re-upload your completed zip file at the end of every week that you work, so that we can see your progress over the entire trimester.

For the final submission, you will also record a video of your completed educational game. This will be an approximately 5-minute video, created in any way you choose. Your video will detail what your game does, what features you wish to be graded on, how you created it, and the challenges faced and overcome.

This is an individual project.

### Platform

The platform **must be strictly** HTML, CSS, and JavaScript. **No other libraries or languages can be used.** No code compilers / converters can be used either (aside from Cordova if you wish to test it on a mobile device).

### Project Directory

Your Project Folder/Directory must contain the following structure:

- **js/** (all your application-code JavaScript files will go here). *Examples:*
  - *app.js*
  - *utilityFunctions.js*
  - *ui.js*
  - *gameElements.js*
  - *game.js*
  - *menu.js*
  - *settings.js*
- **data/** (all your game data will go here, even though they're JavaScript files, they're data). *Examples:*

- *gameLevel1.js*
- *gameLevel2.js*
- *wordbank.js*
- *vocab.js*
- *characterDetails.js*
- *config.js*
- **images/** (all your images are in here)
- **sounds/** (all your sound files go in here)
- **index.html** (the file I open in Google Chrome to run your app)
- **licenses.txt / .csv** (one line per image / sound file you use)
- **changelog.txt / .md** (a list of days you worked and what you achieved)
- **readme.txt / .md** (details of yourself, and an overview of your project)
- **Marking Justification.docx** (A cover page detailing the grade you are aiming for, and evidence for each individual rubric)
- **Demonstration.mp4** (An approximately 5-minute long demonstration video of your project, and the features you wish to be graded for)

If you create your own graphics, put them in a folder: "raw sources"

## Minimum Requirements

**Your game must meet the following standards to pass.** Failure of any one results in a fail grade. The Minimum Passing Requirements:

1. The game must be educational / learning.
2. The game must only use: HTML, CSS, and JavaScript. **No other libraries or languages can be used**, as otherwise we cannot assess what you have learned from the content in this unit. The purpose of this project is to demonstrate that you have learned the concepts covered in this unit, so it's important that you demonstrate what has been taught.
3. The game must use only **a single index.html** file. **No** other html files allowed.
  - The index.html file will have a blank <BODY>, and simply load CSS and JS files through the <HEAD>, which then load and run the game.
4. All data should be logically separated in separate JavaScript files. Your game data should never be "hard-wired" into your application code, but should be in separate files so that a non-programmer would be able to change them without really knowing anything about programming. E.g. data/gameLevel1.js would contain a single, large declaration:

```
// ensure the global gameLevels variable exists, or create it
window.gameLevels = window.gameLevels || [];

// now add the data for this level
gameLevels.push( {
  "levelName" : "Basic Verbs 1",
  "wordList" : [
    "run", "eat", "sleep"
  ],
  "pastTense": {
```

```

        "run": "ran",
        "eat": "ate",
        "sleep": "slept"
    }
    "difficulty": 4
} );

```

## Marking

To achieve a particular grade, you must meet **all** criteria for that grade, as per the table below. Each higher grade requires **all** features of the previous grade as well.

Criteria	Pass	Credit	Distinction	HD
Weekly zip uploads	At least 3 weeks of uploads to Cloud Deakin.	At least 4 weeks of uploads to Cloud Deakin.	At least 5 weeks of uploads to Cloud Deakin.	At least 6 weeks of uploads to Cloud Deakin.
Weekly progress	Changelog updated at least once a week.	Changelog lists all new major features added per day-of-work, and a list of "Still working on" for current features in progress.	You include, at the end of each work day, a summary of how you are progressing related to your milestones.	
Code quality	Indentation is perfect. Each major function has an explanation comment.	Each function has a full explanation comment, as well as function signature (the parameters) explanation.	Each function also has several examples of how to call it (where there are parameters), and examples of any returned data.	
Legal	You have a licenses.txt / .xls file, and all materials you use you have legal rights to use for commercial use.		You also have a Legal section / About page with Legal section, where you make all attributions you are legally required to.	Your attributions also include hyperlinks to source pages, as per author wishes.
Playable Scenes	Your app has at least 1 playable scene / level.	Your app has at least 3 playable scenes / levels.	Your app has at least 6 playable scenes / levels.	Your app has at least 12 playable scenes / levels.
Playability	Your game is engaging to play for at least 10 minutes.	Your game is engaging to play for at least 1 hour.	Your game is engaging to play for at least 5 hours.	Your game is engaging to play for at least 10 hours.
UI Design	Your game has all styling within CSS file(s), and uses	Your game also has a consistent, good-looking layout	Your game's design looks visually appealing, and	

	Colour Theory for appropriate colours in your CSS files.	across each screen, where the user can expect where buttons will be placed, and where spacing is consistent between items.	includes occasionally-animating background elements to create a more dynamic application feel.	
UI Layout	Your app works in portrait phone mode.	Your app works in desktop, portrait phone, portrait tablet, and landscape tablet modes.	Your app dynamically adjusts in real-time to changes in orientation & resolution.	
Code Structure	You separate and group related functions into appropriate JavaScript files.	You have also created at least 1 reusable UI component / game component.	You have also created at least 2 reusable UI components / game components.	You have at least 4 reusable UI components / game components.
Data Structures	You do not hardwire any data within program code. You make use of both local and global primitive variables.	You also make use of both objects and arrays.	You make extensive use of complex data structures, e.g. objects/arrays within objects/arrays.	
Bugs	Your code gracefully handles all bugs and invalid user inputs, and restarts or resumes as appropriate.			
Readme.txt / .md	Includes your name, app title, and an overview of your app.	Includes an explanation of major features.		Includes an API reference of your major functions & data structures, should others take over the project and wish to develop / use it further.
Demonstration Video	You demonstrate all main features working in the app that you wish to be graded for.	Your video is still and clear, and audio is clear and easy to understand.	Your video has a professional feel to it, to the level similar to YouTube.	

Additional marking explanations:

- **Weekly zip uploads:** In every week that you upload a zip of a working copy of your app, this counts as “1 week upload”. It doesn’t matter how many times you submit

in a single week, that still counts as “1 week of uploads”. If you submit in say, Week 5, then Week 10, that would count as “2 weeks of uploads”.

- **Weekly Progress** (Changelog.txt / .md): A file within your project directory of the progress you make each day you work.
- **Code quality:** Code quality means both indentation (TAB key) and comments. A method signature means each of the parameters, as well as the return type.
- **Legal:** You must include a licenses.txt / .xls file in your project’s root directory. Every externally-sourced item must be legal for you to use for commercial purposes, and you must have one row per item, recording:
  - **Item name:** image/sound filename you have renamed it to, or method/function/class where the code is found.
  - **License type:** (Public Domain / CC-BY / CC0 / GPL / MIT / Apache / BSD)
  - **Author / Attribution:** (name of author)
  - **Source Website:** (original website of the content, as per the author’s wishes)
- **Playability:** Playability is the length of time an average player would be able to play your game and get enjoyment out of it. Some games can be “finished” (e.g. all levels complete) in only a few hours, whereas others may take weeks or months. Note, this should not involve lots of “grinding”, where the user is doing repetitive (and usually boring) tasks, but where the player is continually engaged.
- **UI Layout:** Each layout and orientation will be tested using Google Chrome when we mark your work.
- **Code Structure:** A new UI component is any function that creates and returns a HTML element (such as a DIV) that contains other elements within it (e.g. createSearchBar() would create a DIV containing a textbox for the search text, AND a button to enable the search).
- **Data Structures:** It is mandatory that all your data is in separate JavaScript files specifically for data (those files do not contain program code). This is to create a logical separation between program code and data, such that a non-programmer could edit and expand the game’s data and your program would adapt automatically. For example, you would never create 3 buttons for 3 levels in program code manually, but would use a loop to generate the number of buttons, based on the number of levels that are in the data. This way levels can be added without any changes to program code.

Note that there is no HD rubric for Weekly Progress, Code Quality, Data Structures, UI Layout, Bugs, and Demonstration Video. If you get a Distinction category, this also counts for High Distinction. Legal does not have a Credit rubric, so a Pass level also counts as a Credit level.

For the video, aim for around 720 x 406 Universal pixel size (not larger), and around 20mb in size (less is better). We will have 150 student videos to download, so giving us anything larger and we may not get time to download and watch it.

Note that, much against my beliefs, the university does enforce a quota of High Distinctions issued, and generally won’t accept more than 20% of a cohort from receiving an overall HD grade in the unit. There’s very little I can do about this, except try to make the HD criteria restrictive to ensure that we remain just under the quota.

## #3: Portfolio

Critical Information Summary	
Weight	30%
Due	Friday Week 9, 5pm
Individual / Group	Individual
Submission via	Cloud Deakin Dropbox <b>Rolling</b> submission (keep uploading the latest copy of your doc each week) (multiple submissions allowed)
Required Files	20-50+ page MS Word Doc <i>Filename: SIDxxxxxxxx-A3-Portfolio.docx</i>

The Portfolio (which I like to refer to as the "Unit Summary Report") is where you create your notes about this entire unit, week by week. You will also demonstrate that you have accomplished all of the practical exercises (with screenshots as evidence). We have this Portfolio instead of an exam, so keep in mind that you need to demonstrate your understanding equivalent to if you were sitting an exam.

In this Portfolio, you will create a reflective summary of the class notes and answers to practical lab exercises for every topic from every week within our unit.

You are required to present a written report of each week's topic, analysing and discussing how the topic relates to developing mobile applications. You will also include a screenshot of the output of each week's practical exercise, along with any significant learning notes. Appropriate formatting required.

Note, this is due before the Project, because you must have covered all concepts to be able to complete the Project. Thus, this earlier due date somewhat forces you to make sure you cover all theory + practical labs before you can finish the Project.

### Purpose

The Portfolio / Unit Summary Report is where you actually learn. It is the foundation of this entire unit.

The science behind this: The purpose of creating a personal summary is to request your brain's neurons to essentially fire connectivity and activate over new information. When listening alone, passive information comes in and will re-inforce or relate to existing information in your brain, but won't form new memories on its own. Basically, your brain says "oh yes, this information relates to what I already know about ABC, so XYZ is important", and the new information is never "saved" into your memory. Passive listening only reinforces existing knowledge. By pressuring your mind to "create" something (e.g. written summaries), this process pressures your mind to create and use new pathways in your brain related to the new information, thus encouraging them to be permanent.

It is the simplest way to get the information into our brains and have it stay there. So, this exercise is for you. For many topics, you'll find you'll remember the topic for the entire trimester without having to revisit your notes.

## Required Sections

- **Marking Justification:** A cover page detailing the grade you are aiming for, and evidence for each individual rubric.
- Week 1:
  - Notes
  - Lab Exercise Answers
- Week 2:
  - Notes
  - Lab Exercise Answers
- Week 3:
  - Notes
  - Lab Exercise Answers
  - Project Progress
- Week 4:
  - Notes
  - Lab Exercise Answers
  - Project Progress
- etc...
- Week 9:
  - Project Progress


## Marking

To achieve a particular grade, you must meet **all** criteria for that grade, as per the table below. Each higher grade requires **all** features of the previous grade as well.

Criteria	Pass	Credit	Distinction	HD
<b>Presentation</b>	Your doc has consistent styling, title page, headers/footers, heading styles, and has been spell checked & grammar checked.	Your doc uses Styles for all main formatting.	Your work would pass as a professional document, both when read online and when printed.	Your work easily passes for a professional pitch, where professionals and venture capitalists would not be able to tell the difference.

<b>Rich text</b>	You write in mostly plain text.	You include some bullets / lists / tables / images to explain your ideas.	Your doc has an equal mix of text (~30%), rich text (~30%), and images (~30%).	Your doc uses Style padding and margins to maximise clarity through white space.
<b>Explanation</b>	You have accurate spelling & grammar, have appropriate paragraph structure.	Your explanation and analysis are easy to understand, such that you could give your doc to any generic IT student, and they would learn from your report.	Your ideas are both easy to understand (E.g. ELI5), while simultaneously cover enough detail for a team of developers to learn from it.	Your report is at a level where you could sell it as a complete printed book on developing mobile web apps.
<b>Note Summary</b>	You've completed every week's lecture note summary (at least 100 words), summarising all main points.	Each week's notes would be at least 500 words in length, and additionally usually contain at least 5 code snippets. Your notes link each week's points back to previous weeks' learning.  Your notes also ask questions or reflects when you see something that isn't answered, or that you've become curious about. E.g. "I wonder if pixel density relates to those blocky pixels I see in videos when the compression is bad?". This demonstrates that you are critically thinking about the material, and sometimes finding and including answers.	Your notes include images, facts and information you searched for to answer questions or seek more understanding that wasn't included in the unit content.  Your notes also include your own curiosity questions (based on the weekly topics), and also shows you've done some research to get answers.	Your resulting lecture note summary reads like a professional mini book. It flows so coherently that you could give your Portfolio / Unit Summary Guide to a new student, who doesn't have access to the unit, and they would be able to gain a solid understanding of designing apps. This means your document is instructional, informative, and stand-alone.
<b>Lab Exercises</b>	You have made a decent attempt (75% correct) of every week's exercises.	You have complete answers for every week's exercises, mostly correct (90% correct).	Your lab answers also include a "What I Found Useful" sub-section, with a list of learning points that you think you will apply to your Project.	Your lab answers also include brief code snippets / API guide of the overall key code lines that you found important.
<b>Project Progress</b>		You write a brief list of tasks you've	You also include a list of what you plan to do	





		accomplished this week for your project.	next week, and an estimate of man-hours required for each task.	
--	--	--	---	--

Additional marking explanations:

- **Explanation:**
  - Paragraph structure means: topic sentence, supporting sentences, conclusion sentence.
  - ELI5 means “Explain Like I’m 5”, as in, imagine you are explaining an introductory sentence or paragraph of a section to a 5-year-old child. How would you use language, analogies, and generally express your idea so that a 5-year-old would understand the gist of your point?
- **Note Summary:**
  - Distinction: E.g. let's say you're going through the material, and you hear/read and learn a little about pixel density. But then you think "wait, what's the pixel density on android devices. Are these consistent or are they different for every device?", because personally (based on your own interests) this is something you're curious about. Then you go search for answers, and write it up: "Something else about Pixel Density [reference #] is ....". Note: You always write in your own words. Never copy and paste, unless it's a short, direct quote, in quotation marks and with a direct reference following it.

Note that, much against my beliefs, the university does enforce a quota of High Distinctions issued, and generally won't accept more than 20% of a cohort from receiving an overall HD grade in the unit. There's very little I can do about this, except try to make the HD criteria restrictive to ensure that we remain just under the quota.

## Submission

This task is different from others, in that you must complete it weekly, and upload your latest copy of the entire document each week. You'll keep a file: “SIDxxxxxxxx A3 Portfolio.docx”, containing everything you've written so far. And each week, you'll upload that same file again (your document will grow larger each week you write in it). You are also welcome to edit previous weeks' work at any stage.

Note: keep one single entire document (append each week's new content into the existing single document you keep for this Portfolio). **Do not** separate each week into its own document. Every time you upload the document, it should contain **everything**.

# Game Ideas

---

Here is a random list of topic areas that you should consider, ranging from kid's games, to school games, to adult education. All of these ideas are allowable and meet the criteria for the Project.

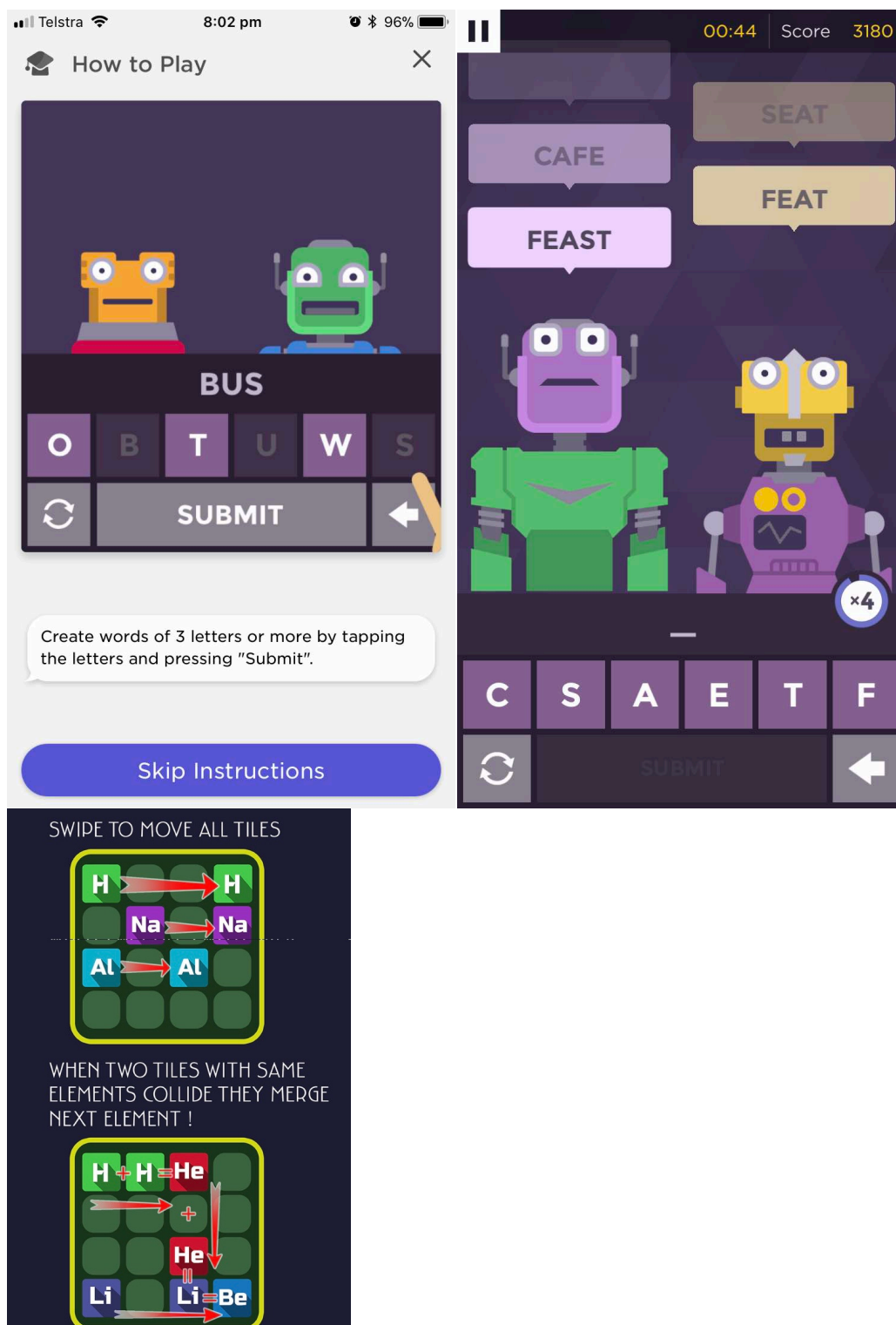
1. Sorting objects into buckets as quickly as possible. E.g. dragging a list of items into relatable circles, such as colours, shades of colour, word types (e.g. past tense / present tense), object types (modes of transport such as car/bus/bicycle vs types of buildings vs types of people), etc. Sorting objects into a sequence. Sequences can be numerical, size-based objects, side-based (e.g. number of sides on a shape), math-based, word-based (e.g. from lightest weighting of a word, such as "kind" to the heaviest weighting, such as "benevolent"), etc. Could also sort patterns involving multiple objects.
2. Listen to word (Text-to-Speech API) and then select which image it is, and pick alternative (incorrect) image options that are very similar in their sound. The learning language can be English or something else.
3. Identify the incorrectly spelled words in a sentence. Have a small data-bank of common incorrect sequences, to find and replace in any word from the word list (e.g. "ou" and "oo" sound pairs, "au" and "uo" sound pairs).
4. Identify incorrect grammar (e.g. tense) sentence within a randomly-generated paragraph.
5. Convert sentences between tenses or pronouns (from first-person to third-person).
6. Banking transactions / maths / loans / credit card simulation to learn about credit / banking / interest rates / supply & demand.
7. Diagnose simple health conditions on a body image, selecting which test to perform and see what output comes out (based on a start-of-game computed scenario).
8. Travel the world in a plane by calculating fuel required for each leg of the trip (while simultaneously covering geography topics).
9. Learn-to-type exercises, measuring speed and accuracy of each set of exercises.
10. Memory match: tap to reveal up to two cards at any time, and when they are a match then they stay revealed. The player will remember positions of cards to try and match all cards in the fewest steps possible. The board size varies automatically per level (e.g. 2x2 tiles, 3x3 tiles, 4x4 tiles, etc.), each game randomly selecting a different set/topic of tile pieces.
11. Crossword puzzles (computer-generated each time).
12. Any of hundreds of math-type lessons (where one taps the answer, not types it, to make it mobile-friendly).
13. Be given (computer-randomised) weather reports, and match what clothes will minimise discomfort (e.g. if a raincoat but no rain pants, and it's raining with wind, then discomfort level of legs would lower player's score).
14. Be given text set of directions for memorisation, then be shown a computer-generated map and have to remember the journey, from the player's position (different each game), to find some location on the map. E.g. player is given "Turn left at first intersection, right at 4<sup>th</sup> intersection, travel past 3 shops, and click the shop on the left". The user can keep clicking location boxes if they get it incorrect, but their

score decreases each incorrect response. Could also do the same for trains, “Take the Lilydale for 4 stops, then switch to platform 3 and take the next train for 2 stops, then platform 8 for 1 stop, and exit via the North exit and find me there (tap the correct person to meet).

15. Randomised messy house of items (as images/icons), and need to, as quickly as possible (timed), drag items to their correct location (cupboard, kitchen counter, bathroom, etc.).
16. Age of discovery sailing game. Can navigate between locations (and learn about them), where each location may have some, but not all, required items for the continuation of a journey (e.g. food / water / fuel). Each item may have different bartering amounts, or may not be tradable at all, at different ports.
17. History: give clues as to what time period a time traveller is in, and the player has to guess the period from a given list of periods, e.g. 1600AD London, 1300AD (no location given), etc.

## Sample Apps

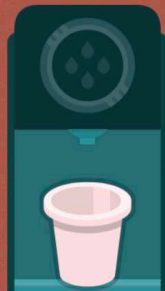
This section contains a selection of screenshots from educational games (mostly from Peak, Elevate and Lumosity). You don’t need to specifically know how any of these games operate. Rather, use these as inspiration for the kinds of educational games you can create. Especially focus on how the interface is responsive, and is extremely easy to use on a touch-based device.



Skip Tutorial



This order asks for sugar. Tap the SUGAR JAR below to add sugar to the cup.



TIME 0:41 SCORE 15930



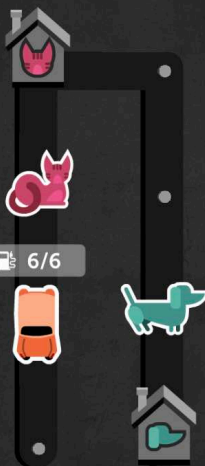
Skip Tutorial



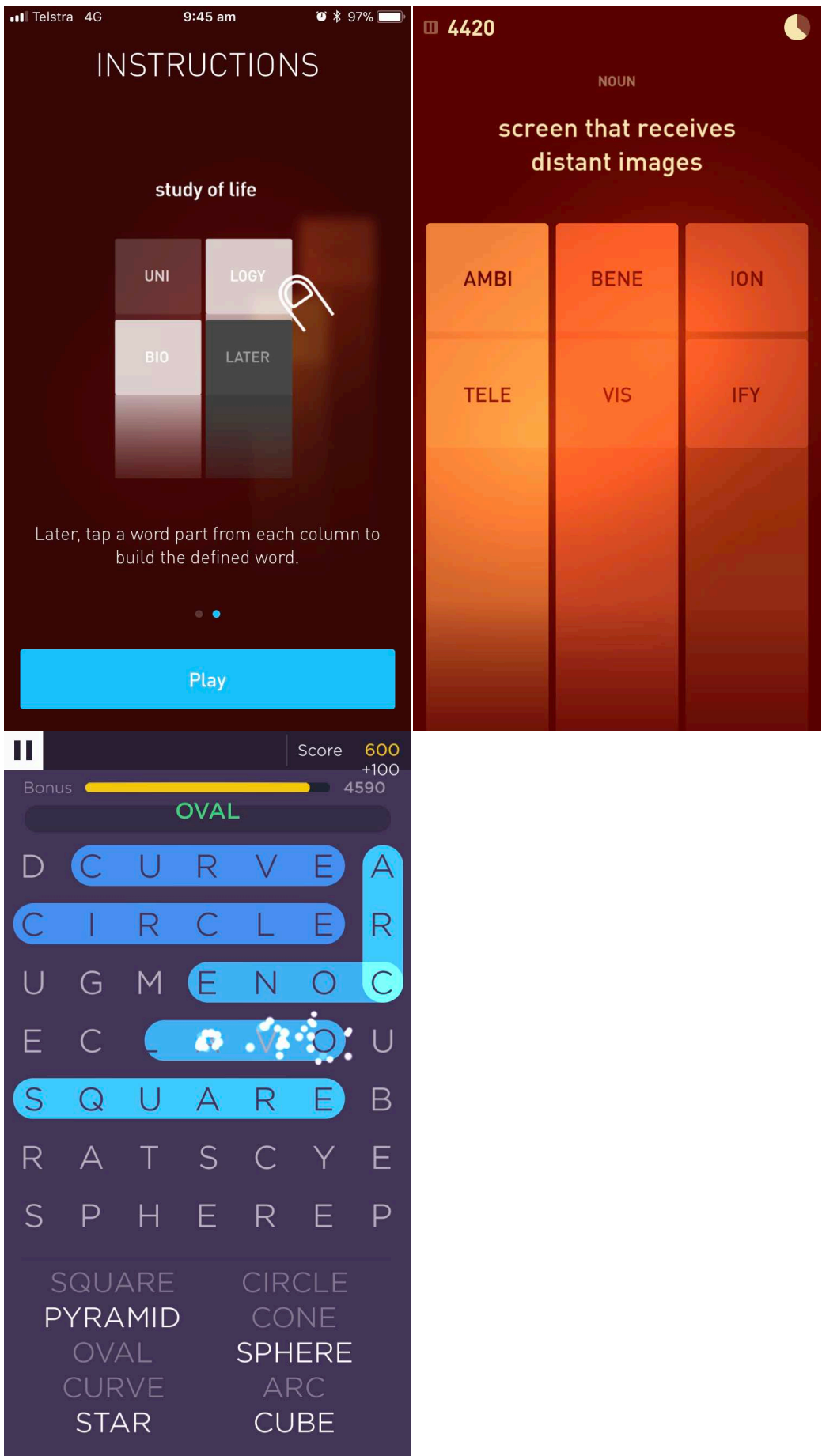
Tap the BUTTON to start filling the cup.



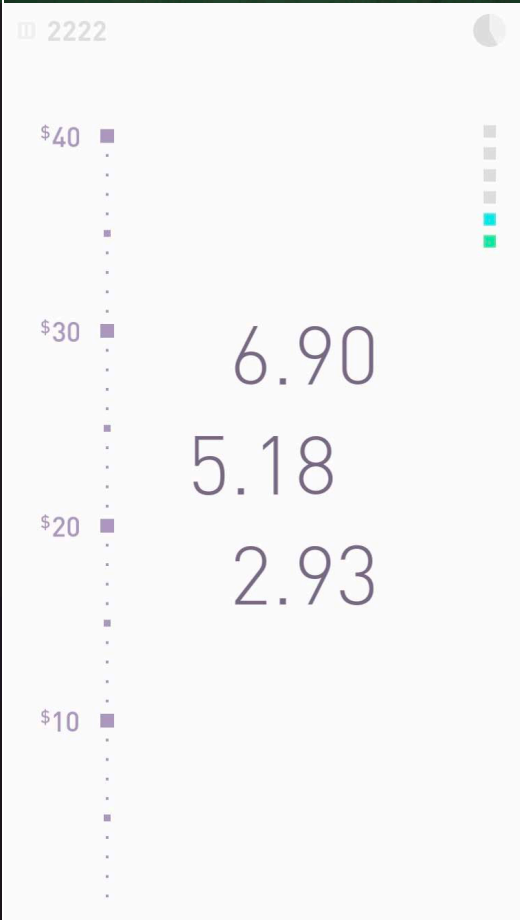
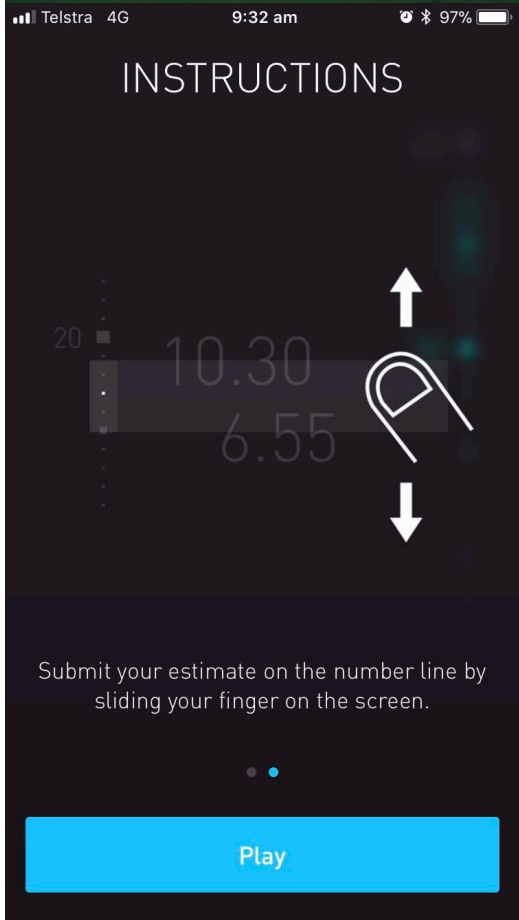
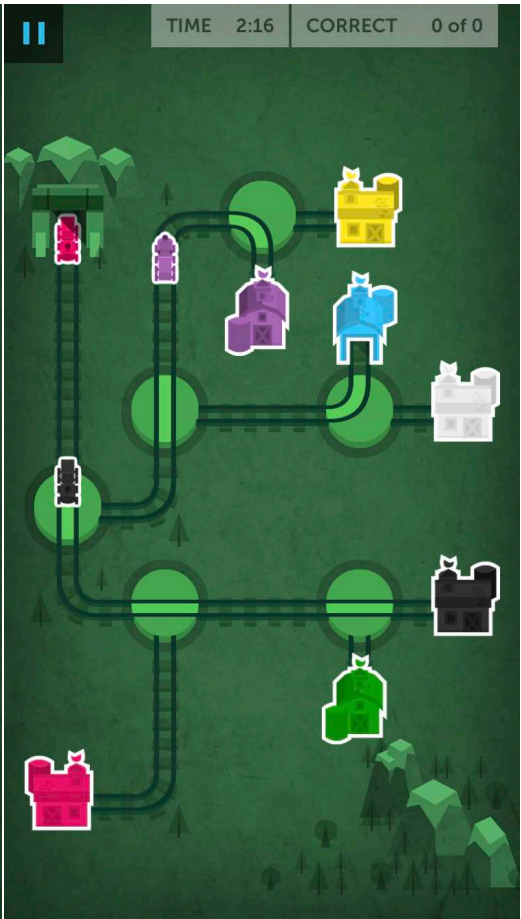
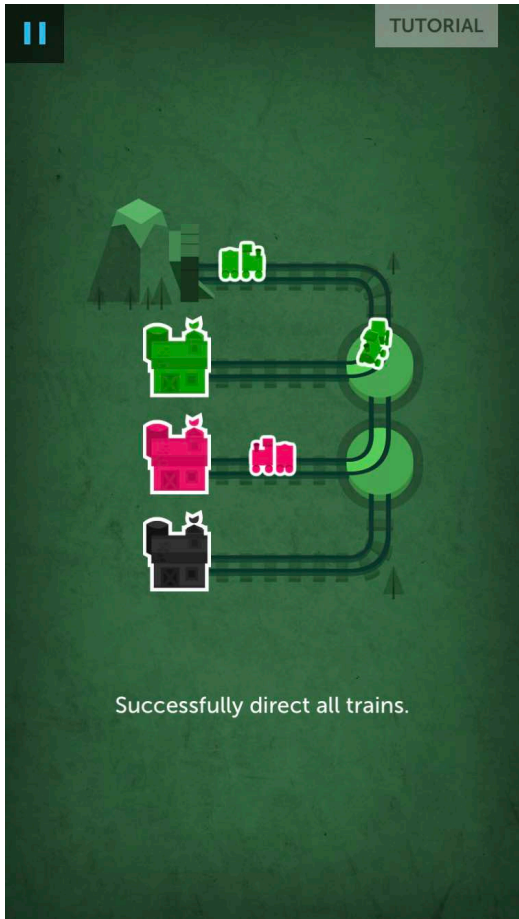
Skip Tutorial

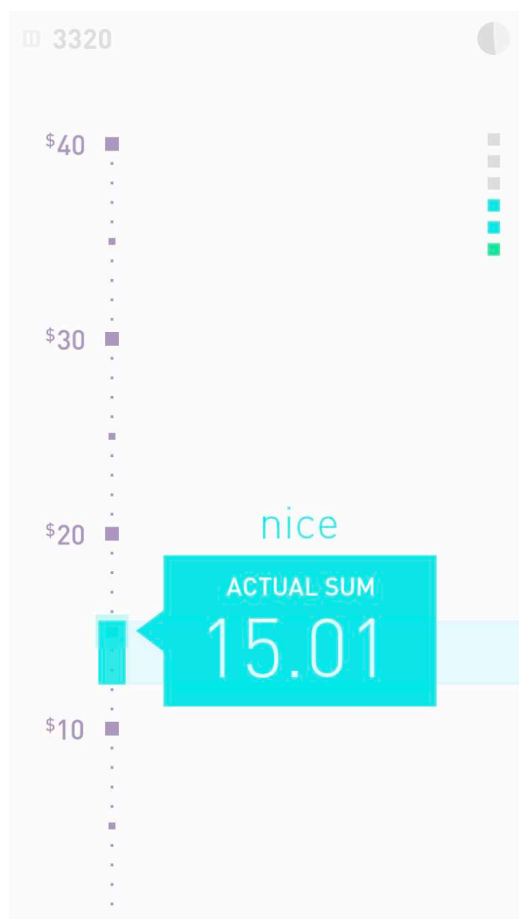


You have enough fuel to drive 6 spaces. Return both pets before your fuel runs out.

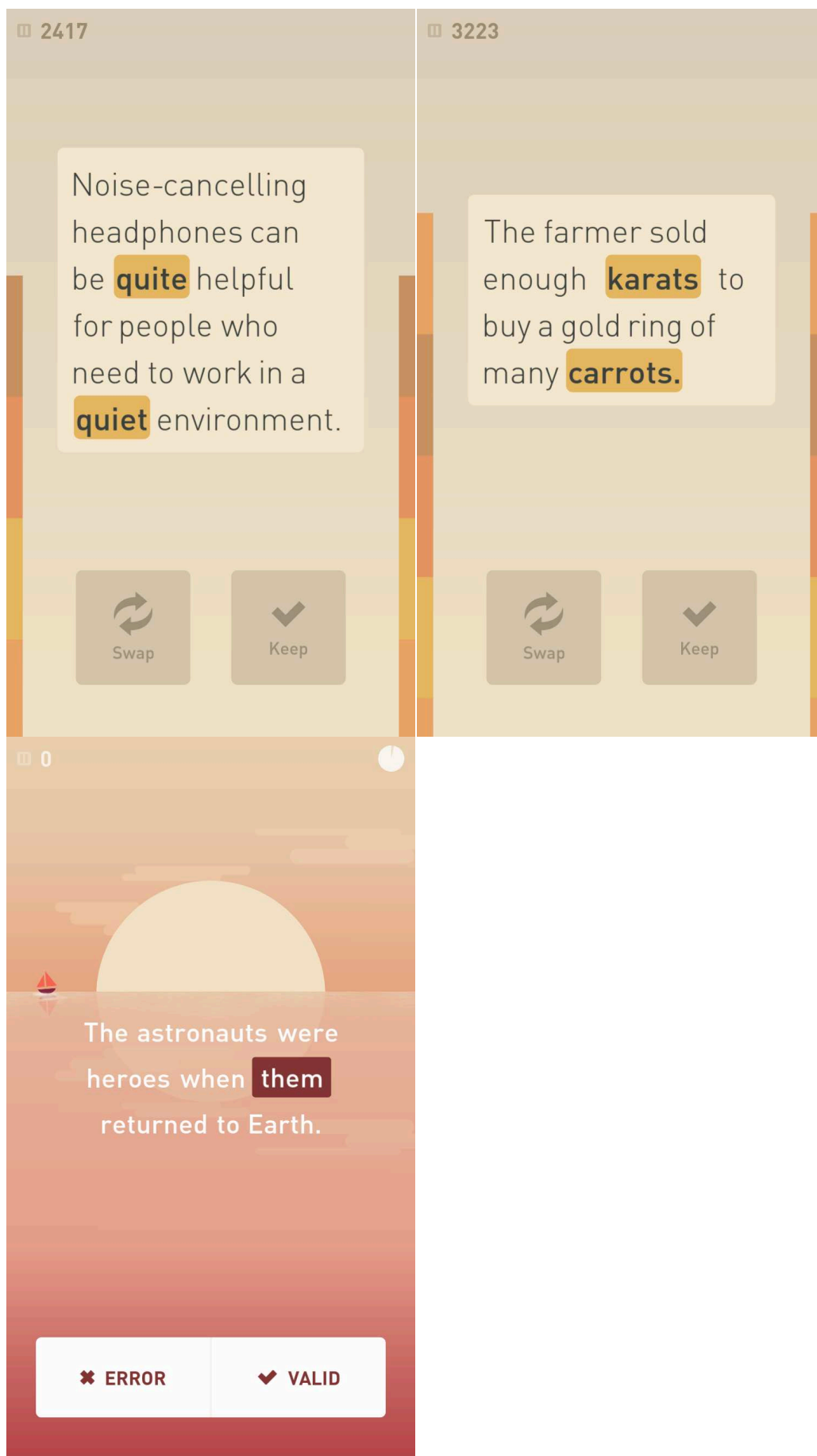












0

♥♥

When I came to the edge of the gorge, it felt as if my whole body had been dipped in liquid nitrogen. The wind tugged at my legs and hair, taunting me, wanting me to fall. Closing my eyes, I counted to five and stepped forward. When I opened

INFERENCE

The author would most likely have the same reaction while


riding in a hot air balloon


being enclosed in a small space

taking an elevator

Skip Tutorial

Tomato must be in position 2.





1

2

In this game, you follow rules to plant seeds in a garden.

SUBMIT

5200

♥♥

When I came to the edge of the gorge, it felt as if my whole body had been dipped in liquid nitrogen. The wind tugged at my legs and hair, taunting me, wanting me to fall. Closing my eyes, I counted to five and stepped forward. When I opened

OVERVIEW

The author's tone in the passage evolves from


terror to wonderment


anxious to tranquil

cowardly to underwhelmed

Skip Tutorial

Tomato must be in position 2.





DRAG the seeds down to the numbered slots, and then tap SUBMIT.

SUBMIT

Document Version: 2018-02-16

34 / 40

