

SIT305

```
> /unitchair "Henry Larkin"  
> /topic "Advanced Mobile App Development"  
> /build iOS, Android
```



SIT305

Table of Contents

Overview	3
Physical Lab Space (Co-working space)	4
Weekly Blackboard - Ask Henry Anything.....	5
What You Can Expect from Me (Henry)	5
What I Expect from You	5
Attitude.....	6
How to Get Help	6
How to Email Henry (template)	7
Banned List.....	8
Feedback to me	8
Assessment	9
Assessment Rules & Standards	9
Effort: Demonstrate 14 hours per week.....	9
Marking Justification	9
University Policy on Late Submissions.....	10
Plagiarism Advice	10
Presentation.....	11
#1: Project Plan	12
Topic	12
Required Sections.....	14
Marking.....	15
#2: Project.....	20
Platform	20
Project Directory	20
Marking.....	21
#3: Project Report.....	24
Required Sections.....	24
Marking.....	25
Code Advice.....	26
BitBucket / GitHub.....	26
Configuring Git	26
Daily GIT Usage	27

Overview

Welcome!! I'm so happy you have chosen to enrol in SIT305.

This unit will run quite differently than other units you've had in your first and second years at Deakin.

This unit is all about **you** developing a complete, mobile app, solving and overcoming unknown factors as you go. There won't be any formal content I convey to you. You'll learn as you go through the process of developing your app, and working out solutions as you go. There are no lessons, no lectures, just you, maybe a team mate, and your IDE of choice.

This is going to be an incredibly rewarding unit, but it does mean you start from Day 1 planning what you have to do, and making it happen asap.

Here is the super-short summary of this unit:

1. I want you to create a real world, published, app. That is the primary purpose of this unit.
2. I would prefer you to do it with one other person (thus forming a team of two), anyone you want, to practice using code repository tools. But if you prefer to work alone, that's also ok, as long as you still interact with your code repository the same as you would if you were working in a team, as this is an essential skill for getting a job.
3. You'll be making a game. The focus will be on programming and data structures (not graphics).
4. Labs: they're a place for you (if on Burwood campus) to work with your teammate. Not compulsory. There are no exercises, no teachings, no tutor, just plain and simple lab space with Macs and devices. You're welcome to come to any times that you want, I've booked 1.5 days of lab space. Don't worry what your timetable says, it doesn't matter in this unit.
5. Assessment:
 - a. First, make a plan. You'll do this individually. You should have done SIT120 with me, and know how to do this. I've also provided a sample template document. Keep in mind the plan should be as professional as possible. Imagine posting this up on a Kickstarter page to get funding to make your app.
 - b. Second, create your app. 50% marks right here. You must show weekly progress, and show evidence of code repository usage and planning / task management. If you're working with another student, then you'll also show evidence of communication. Team communication is critical to success. So critical. My worst projects have been where there hasn't been enough communication. My best projects are where we were communicating several times each week.
 - c. Final assessment item: a reflective report on what worked well and what didn't. 3 items: Plan. Produce. Reflect. Simple as anything.
6. ALL your time will be spent working towards your goal of an app. I'm not here to tell you more theory. No lectures, no classes. You have to discover for yourself what new

theory you need to create your app, and then YOU need to prove to me that you can find the answer yourself. This is because in your real jobs next year, you will have to do this from day 1. You'll often be told on Day 1 of the job, "we use package XYZ", which you'll have never even heard of. Then you'll spend the next week googling and reading how to use it, experimenting with some small code samples and making them work, until you are proficient. This is how most jobs are. Even if you're working on your own entrepreneurial project, you will need to do things you haven't done before, and that means finding your own resources and answers, testing small examples, and figuring out how something works. Most importantly, part of this unit's learning outcomes is you, proving to me, that you can learn new things without my help. Then I know you'll make it in a real job. Then I can pass you.

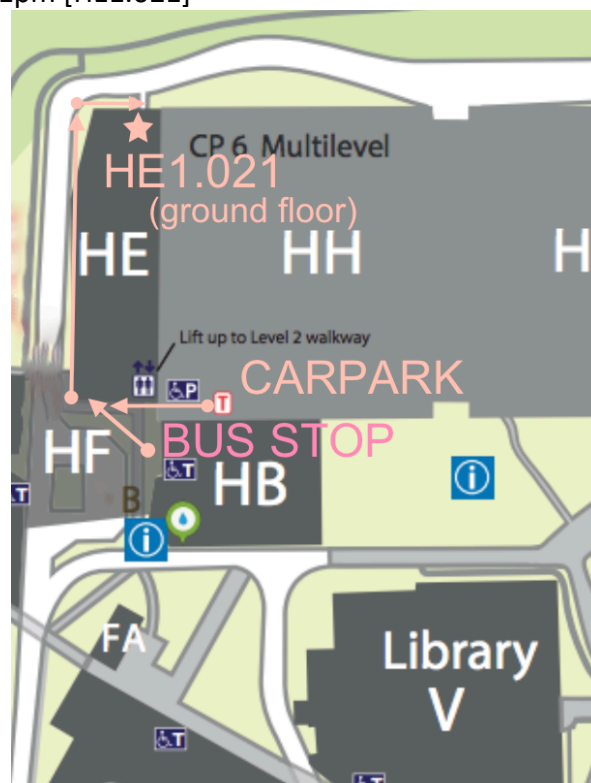
Welcome aboard. And I can't wait to see your apps as they unfold 🤖

Physical Lab Space (Co-working space)

We have 8 hours of physical lab space in HE1.021, for use in working with your partner on your project. You are welcome to use any of this time as you see fit.

There won't be any specific assistance in these labs, as this unit is all about self-directed learning. The room is simply a space to have access to Mac machines (running both XCode and Android Studio) if you're around Burwood campus.

- Monday 9am - 5pm [HE1.021]
- Tuesday 9am - 1pm [HE1.021]



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 3pm-4pm** (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 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"> • Personal issues • Extensions • Questions you don't feel comfortable asking online <p>I'll generally reply within 5 business days (on Mondays). If longer than 5 business days,</p>

	email again, as your email either got lost or couldn't be understood.
Forums	<ul style="list-style-type: none"> • Code questions • Unit query questions • Hints / advice to other students <p>I'll reply to forums on Mondays.</p>
Webinar / Blackboard (Monday 3pm)	<ul style="list-style-type: none"> • General chit-chat • Ask-Henry-anything • Questions on your project. • Quick reviews / short feedback of your work.
Burwood labs	<ul style="list-style-type: none"> • iMac machines to work with • Collaborate with teammate

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:** SIT305 (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:

1	Project Plan	30%	Week 2 , Friday 5pm
2	Project	50%	Week 10 , Friday 5pm
3	Project Report	20%	Week 11 , 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.

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, 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). For the Project, you’ll also have everything in BitBucket / GitHub.

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 (MS Word Doc) <i>Filename: SIDxxxxxxxx A1 Project Plan.docx</i>
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 RPG (Role Playing Game). It can be text-based, or text with static background scenes. I wouldn't recommend trying too graphical beyond that, as there wouldn't be enough time during this short trimester (but I won't place any specific restriction stopping you). But I won't be grading any use of graphics engines, only your algorithmic code in creating your own game engine.

The game should have multiple locations (e.g. "rooms") that the user can navigate to (either through buttons or text commands such as "go north"). Locations should have interactivity, both with passive objects (e.g. "open door", "light candle"), and NPCs (computer characters with dialog, using either text or text-to-speech). A large focus of the project will be on your design of text-based data files for game maps, game logic, and any dialog.

As much as possible, I want you to be creative, use your imagination, and create both a world and a journey for the player, through your dialog scripts and the story you tell.

Some ideas you may want to consider:

- 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. The gaming aspect is to meet mission objectives as issued per level / scenario.

- Detective in a small town, going in any order through clues picked up, for each crime (where a crime is a “level” of gameplay).
- Time traveller. Travel to any period in time, and as interactions with historic events take place, these adjust attributes on the world-timeline-tree of data you’ve created, such that different periods in time then change. The time traveller is given specific mission objectives (like “Prevent World War 1”), to try and achieve without erasing themselves / humans from existence.

As a recent example of a text-based RPG mobile app, take a look at “A Dark Room”, which was very successful in sales. The 2 screenshots below to give you an idea.



And an older PC game with static background scenes, Carmen Sandiego:



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 (first page of your Word Doc).
2. **Overview:** A brief, well-explained overview of the app you'll develop.
3. **Story:** Explain the story / type of world you will be creating, the main characters, and the overall objectives of the game (note this section doesn't have a specific rubric).
4. **Competitor Analysis:** An analysis of 5+ competitors, either mobile app, traditional game, or and other physical game. 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: name, price / monetisation, basic story, core game 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 \$15 for your app. What does your app need to contain, and how professional does it need to look & feel, for you to justify a \$15 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, UI colour choices, and any UX decision (user experience, e.g. the flow of common actions).
8. **Data:** A top-level overview of the data tables / structures you will store in your app. This will include both:
 - a. **User Variables** (in-memory variables, such as user score, current level), and
 - b. **Game Data** (fixed data from disk storage) used for your app (e.g. each game level map layout, conversation dialog flow, etc.). This will be loaded into memory when your game loads, but won't be user-editable (usually).

You may need to skill up your knowledge of data if you didn't cover it enough in previous units, as this is a significant part of the project. Generally, all data structures should be text-based files, so that you can easily modify them, and/or add levels or game content.
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. Ensure you have license to use them for commercial purposes (CC-BY / CC0 / Public Domain licenses).

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
----------	------	--------	-------------	----

Presentation	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.
Rich text	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.
Explanation	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.
Competitor Analysis	5 competitors, each with a minimum 150 words, identifying the strengths and weaknesses of each.	5 competitors, each also with a relevant in-app screenshot.	8 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.
Features	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.
Milestones	A list of milestones you plan on making, and the estimated man-	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	You also identify a breakdown of most tasks for each milestone (can be a

	hour durations multiplied by two.		justifiable (based on the dependencies).	simple sub bullet point list)
Design	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 RGB values), using any colour theory, and display the colours on one of your wireframes.	You include a subsection identifying a bullet list of most common user actions, and the number of taps to achieve the result. (e.g. Button 1 -> Action 4 -> Scroll)
Data	You briefly explain the type of storage system(s) you'll use, and a bullet list of the major files/tables.	You include a table for each major data object, outlining the field names and types.	You include a subsection containing a bullet-point list of actions that will be taken if certain data is missing or corrupt.	You include short example snippets of each data object / table (all text-based file formats).
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. 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 Structures:**

- You **cannot** hardwire any game data (levels / characters) inside code, but are in files for a non-programmer to edit/update your levels without programming. Thus, your game data **must** be text-based, each in its own separate file within your app. Most likely you'll use JSON, but you may have some files in CSV, or even your own custom line-based text format. You may even have your own escape characters for defining variable usage within text, e.g. "\$playerName\$, you are lacking magic."
- You should usually use JSON for any complex data structure, as it is the easiest text format to use for things like character/NPC attributes.
- You may create your own syntax, not just in whole text files, but within Strings in any JSON data. For example, you may have conditions on an enemy NPC taking a hit, with your own syntax "attributeName: change/value", and allowing multiple value pairs separated by semi-colons:

```
{
  "name": "Thief",
  "onAttack": {
    "punch": "health:-1",
    "knife": "health:-1; clothingDamage:-1"
  }
}
```

- Usually you want to provide options from certain player choices, even if you've created your own text file format (e.g. for dialog):

```
# these lines are comments in my file format
player: How are you?
# intents mean that this is part of a conversation that only triggers
# if the above dialog option is chosen.
shopkeeper:
  # Any { } is a line condition, it will only trigger
  # if the condition is met.
  # Anything in [ ] are options, with the selected option
  # being randomised.
  {shopkeeper.mood > 0.5} [I'm pretty happy. | I'm feeling great!
| Isn't today just wonderful.]
  {shopkeeper.mood <= 0.5} [Not so great. | I'm feeling great! |
Isn't today just wonderful.]

player: How's the weather today?
# Anything inside a pair of $ $ symbols is a variable lookup inside
# either (1) the global gameData object, (2) the currentNPCs object
# of NPC(s) we are interacting with.
shopkeeper: Today is $weather.today$.
```

```

        {weather.today == sunny|bright|good} Isn't it great to be
outdoors.
        {weather.today == rainy|cloudy|overcast} It doesn't look like
today is going so well.
    # The player is now presented with 2 options to choose from,
    # as both next lines are "player:" lines
    # with indented lines following, meaning they start a
    # mini conversation within the indent
    player: I'd like to buy some fish.
        {{player.inventory.fish: +4}}
        {{exit}}
    player: I'm interested in learning more about this town.
    ...

```

- **API:** You can achieve a pass without this.
 - 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	2-member team, or individual
Submission via	Cloud Deakin Dropbox (multiple submissions allowed) and Code repository (BitBucket / GitHub)
Required Files	Zip file <i>Filename: SIDxxxxxxxxx A2 Code.zip</i>

The core of this unit, is for you to develop an entire app, either yourself or in a team of two. Even if you are working alone, **you must** use a code repository every day that you work, so that you practice this essential skill for your careers. If you want to work together with someone, please post a message on our discussion forums, with: (1) App title, (2) App platform, (3) Grade you want, (4) Hours you will promise each week, (5) Brief description of the app. Then replies to that post would follow on with experience + hours promising.

Platform

The platform of your project is your choice. It can be iOS or Android. Also, you don't have to use native code, but you also have the option of Cordova, Unity, etc., depending on what skills you wish to spend time advancing in this unit.

Project Directory

Your Project Folder/Directory must contain at least the following structure:

- **data/** (all game data is in here, e.g. json/csv/text files)
- **images/** (all your images are in here)
- **sounds/** (all your sound files go in here)
- **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 any team member), and an overview of your project, BitBucket/GitHub link, etc.)
- **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"

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
Code commits (bitbucket / github)	At least 10 unique-day commits (per person). If in a team, also an equal number of fetch/pulls.	At least 15 unique-day commits.	At least 20 unique-day commits.	At least 25 unique-day commits.
Weekly progress	Changelog updated at least once a week (per person)	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. Every class, and each major method/function has a brief comment.	Every class, and each major method/function has an explanation comment, as well as method/function signature explanation.	Each major method/function also has several examples of how to call it (where there are parameters), and snippet examples of 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.	
Playability	Your game is engaging to play for at least 30 minutes.	Your game is engaging to play for at least 2 hours.	Your game is engaging to play for at least 5 hours.	Your game is engaging to play for at least 10 hours.
Data Handling	You load all your game data through local text files.	You also have all app constants in local text files.	You also load & save user settings to an external file / DB / resource.	You also load & save user/player state from/to external file / DB / resource.
Layout	Your app works in portrait phone mode.	Your app works in both portrait phone, tablet	Your app dynamically adjusts in real-time to	

		portrait, and tablet landscape mode.	changes in orientation & resolution.	
Bugs	Your code gracefully handles all bugs, and restarts or resumes as appropriate.	Your code also keeps all data safe and saved, as well as captures and details and reports bugs to the end-user at a high level.	Your code also keeps a log page, that the end user can access, listing all bugs captured.	
Readme.txt / .md	Includes your name, app title, app platform, link to github/bitbucket, and an overview of your app.	Includes an explanation of major features.		Includes an API reference of your major public classes / functions / methods, should others take over the project and wish to develop / use it further.
Publishing	Your app meets all App Store Guidelines.	Your app has all icons and splash screens appropriate for the devices you are targeting.	Your app has a directory/folder of all information required to publish your app (including all promo text and screenshots per device, as required)	Your app is published.
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 company YouTube videos.	

Further detail / supplementary information:

- **Code commits:** A unique-day commit means any day you have at least one commit (although that day may have many commits), this will count as “1 unique-day commit”.
- **Weekly Progress** (Changelog.txt / .md): A file within your project directory (and updated on GitHub/BitBucket of course), of the progress you make each day you work.
- **Code quality:** 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. A good measure is thinking about if a gameplay video was made of a typical walk-through, would a non-playing viewer of the video feel compelled to watch the entire video, engaged in the storyline and wanting to know how it finishes? The length of playability is one of the biggest determinants of your grade.
- **Data Handling:** The data of your app **cannot** be hard-wired into the app, but rather you must use external text files (JSON / CSV / your own custom line-based format / etc.). The files you create for your game data must be easy to edit / update / create new additions with, such that you wouldn't need a programmer to be involved in story/world updates or fixes.
- **Readme:** You can get an idea of good readme files by browsing open-source libraries on GitHub. Note there is no Distinction category for Readme.

Note that there is no HD rubric for Weekly Progress, Code Quality, Legal, Layout, Bugs, and Demonstration Video. If you get a Distinction category, this also counts for High Distinction.

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: Project Report

Critical Information Summary	
Weight	20%
Due	Friday Week 11, 5pm
Individual / Group	Individual
Submission via	Cloud Deakin Dropbox (multiple submissions allowed)
Required Files	1000 – 3000+ word MS Word Doc <i>Filename: SIDxxxxxxxxx A3 Report.docx</i>

The Project Report is where you'll reflect on how you went about achieving the Project, the processes, skills, systems, and software you used, and how effective they were at achieving the completed project as efficiently as possible.

For example, did you use task lists (such as Trello boards), and how did you find them? Were they time-consuming, were they useful for only certain tasks? Did you give yourself reflective sessions, where you sat without a computer for 15 minutes, thinking about the big picture of how your app is progressing, and did this give you insights or ideas that you then implemented?

The Project Report is a reflection on **how** you worked, not on what was achieved. I want you to think about your actions during the Project, and if you would want to go about things differently if you did the Project again. Would you spend more time planning? Would you use different tools, and why? Would you want to set time limits on your work periods, and for what reason? Etc. Etc.

Required Sections

1. **Marking Justification:** A cover page detailing the grade you are aiming for, and evidence for each individual rubric.
2. **What worked well:** What processes / habits / approaches / systems / software that you used were helpful, and why were they helpful.
3. **What didn't work well:** What processes / habits / approaches / systems / software that you used were helpful, and why were they helpful.
4. **Readiness:** Do you consider yourself ready, right now, to work in a larger team in the real world, either in a company and/or as a private entrepreneurial group. Or do you think you aren't ready for real world employment yet (and if so, what do you independently need to do to get to that level).
5. **Overall:** Overall how did you find the experience.

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
Presentation	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.
Rich text	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.
Explanation	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 take your idea and develop it.	Your report is at a level where you could sell it as a small printed book, for advice for programmers on achieving small projects.

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 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.

Code Advice

BitBucket / GitHub

It is an essential requirement that you use a code repository for your Project in this unit. The reason behind this is fairly easy – almost every single workplace for programming jobs uses a code repository. However, they're also incredibly cumbersome to learn initially. So, it's much safer to practice this skill as a student, rather than in your careers where you may risk destroying a company's entire code directory.

While you should learn all unknown concepts during this unit independently, I will give you some hints to get started on using code repositories. My current preference is BitBucket. My rough notes follow. Note, these are very rough.

Configuring Git


GIT is a code repository protocol, a place where we can store our code, and all access it / share it / upload it without conflicts. It also stores all versions of uploads, so we can roll back to a previous version in case anyone messes up a file. The "server" not only stores all versions, but every single user also has a complete local copy on their own machines.

The particular server we'll be using is called BitBucket (as it's free with university email accounts).

1. Create an account on bitbucket (<https://bitbucket.org>). Write down your username + email address (you **must** use your Deakin email address). You'll also include these details in your project readme.txt/.md file.
2. Install Tools
 - a. For Windows: <https://git-for-windows.github.io/>
 - b. For Mac: Tools are already installed on command-line.
 - c. Alternatively, try a GUI client, and Clone the existing Bitbucket repository:
<https://tortoisegit.org/docs/tortoisegit/tgit-dug-clone.html>
3. Configure your username + email on your computer (use deakin email). If you aren't using a GUI client, then you do this on the command line:


```
# Open Command-Line/Terminal
# Use username + email you use on bitbucket
# Type the following two commands:
> git config --global user.name "Firstname Lastname"
> git config --global user.email YOURNAME@deakin.edu.au
```
4. Create a directory for your web-app projects. It could just be in your home directory/Sites.


```
# Mine on OSX is in:
# /Home/USERNAME/Documents/Projects/
# Open Command-Line/Terminal
```



```
# Change to your web-app project directory
> cd Sites/Web-Apps/
# Download the existing project
> git clone
https://YOUR_USERNAME_HERE@bitbucket.org/YOUR_USERNAME_HERE/PROJECTNAME.git PROJECTNAME
```

5. You're now done, and you can use the tools in either a GUI client (better), or the IDE. Note that your IDE usually has poorer quality GIT tools compared to stand-alone clients. But that's for you to experiment with. Good luck!

Daily GIT Usage

1. Whenever you start your daily session, download the latest updates from the server, and merge them with your local copy. This is called a "git pull" operation (and does a "git fetch" + a "git merge" combined). This will take about a minute. Then you're up-to-date.
2. When you are finished for the day, there are several steps to perform.
 - a. FIRST, do another "git pull", to make sure you get any changes that happened while you were working. This is because your teammate may have been working on files at the same time, and if you don't have the latest version then you'll get an error that your copy is behind.
 - b. Second, you then want to do a "git commit", which will take your saved changes for the day, and save them to your local copy of the GIT database.
 - c. Finally, you do a "git push", which will then save all committed changes (in your local copy of the GIT database) to the server.
 - d. Now your files will be up on the server. You can confirm that by accessing the bitbucket website and looking at the "commits" tab.