UNIVERSITYOF HERTFORDSHIRE
School of Engineering and Computer Science

COURSEWORK ASSIGNMENT

Module Title: Mobile Computing	**Module Code: 6COM1047**
Assignment Title: Mobile App Development Using Xcode	**Individual Assignment**
Tutor: Dr. Xianhui Che	**Internal Moderator**: Guy Saward

Student ID Number **ONLY**:	Year Code:
16052488	6COM1047

Marks Awarded %:	Marks Awarded after Lateness Penalty applied %:

Penalties for Late Submissions

- Late submission of any item of coursework for each day or part thereof (or for hard copy submission only, working day or part thereof) for up to five days after the published deadline, coursework relating to modules at Levels 0, 4, 5, 6 submitted late (including deferred coursework, but with the exception of referred coursework), will have the numeric grade reduced by 10 grade points until or unless the numeric grade reaches or is 40. Where the numeric grade awarded for the assessment is less than 40, no lateness penalty will be applied.
- Late submission of referred coursework will automatically be awarded a grade of zero (0).
- Coursework (including deferred coursework) submitted later than five days (five working days in the case of hard copy submission) after the published deadline will be awarded a grade of zero (0).
- Where genuine serious adverse circumstances apply, you may apply for an extension to the hand-in date, provided the extension is requested a reasonable period in advance of the deadline.

Please refer to your student handbook for details about the grading schemes used by the School when assessing your work. Guidance on assessment will also be given in the Module Guide.

Guidance on avoiding academic assessment offences such as plagiarism and collusion is given at this URL: http://www.studynet.herts.ac.uk/ptl/common/LIS.nsf/lis/citing_menu

UNIVERSITYOF HERTFORDSHIRE
School of Engineering and Computer Science

ASSIGNMENT BRIEF

Students, you should delete this section before submitting your work.

This Assignment assesses the following module Learning Outcomes (Take these from the module DMD):

a. Knowledge and Understanding:
Successful students will typically have a knowledge and understanding of:

[2] Principles of mobile operation and usability
[3] Development and evaluation practices in mobile development

b. Skills and Attributes:
Successful Students will typically be able to:

[4] Write an app using a well-supported mobile platform and development environment
[5] Handle issues of connectivity, user experience, accelerometry and location awareness in mobile programming
[6] Critically evaluate the usability of a mobile app

Assignment Brief:

Please see the attached pages for the detailed description.

Submission Requirements:

Canvas:

- Source code: Zip your Xcode project folder into one file and submit.
- Usability report
- Demo video
- ReadMe file: You may wish to submit a ReadMe text file as an optional document. If you have any special features that cannot be tested using iOS simulator, or if you have put your demo video somewhere else other than Office 365, provide the download link in the ReadMe file.

GitHub: Your repository on GitHub must be private. Please add b.ip@herts.ac.uk as a member or collaborator of your project. The name of your repository should be your **student ID**. Failure to do so may lead to a non-submission of version control.

Viva: This is optional. You may be called to attend a viva session if there are reasons to question your code.

This assignment is worth 90 **%** of the overall assessment for this module.

Please note the quality of your demo video will not be assessed. The demo video will serve as a crucial evidence of your work. Should your app fail to run during the testing for various reasons, your work will not be capped at 40% based on the evidence of demo.

UNIVERSITYOF HERTFORDSHIRE
School of Engineering and Computer Science

A note to the Students:

1. For undergraduate modules, a score above 40% represent a pass performance at honours level.
2. For postgraduate modules, a score of 50% or above represents a pass mark.
3. Modules may have several components of assessment and may require a pass in all elements. For further details, please consult the relevant Module Guide or ask the Module Leader.

Typical (hours) required by the student(s) to complete the assignment: **100** hours

Date Work handed out:	**Date Work to be handed in:**	**Target Date for the return of the marked assignment:**
04/11/2019	15:00 British Time, 13/01/2020	10/02/2020

Type of Feedback to be given for this assignment:

Students will be provided with a score and qualitative comments on their submission via Canvas

UNIVERSITYOF HERTFORDSHIRE
School of Engineering and Computer Science

Assignment Brief

The task of this assignment is to develop an iOS mobile app using Xcode platform and Swift language. The game features a similar concept to the globally well-known title ***Angry Bird***. The basic principle is to shoot a ball to the bird target. Once the game ends, a completion view should be displayed with the facility to restart the game, as shown in Figure 2.

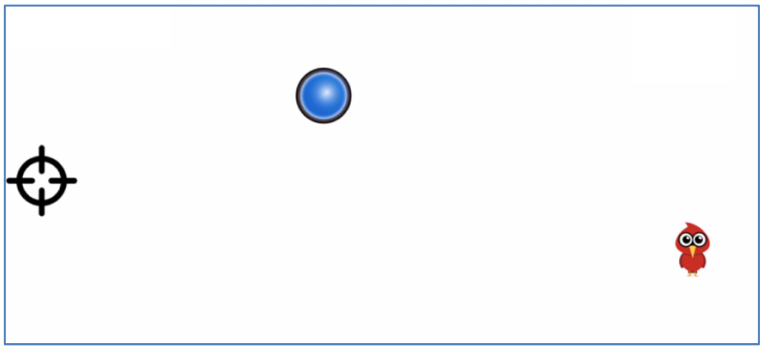

Fig 1. Gaming Scene

Fig 2. Completion Scene

Showcase Video:

You can view the showcase videos of this app on *Canvas/Units/Assignment/Showcase Demo Stage 1-3.mov*, which gives you a brief idea of the basic app design. There are three different stages of development, as demonstrated in the showcase videos:

1. Stage 1 – Initial development
2. Stage 2 – Basic game
3. Stage 3 – Examples of Stretch abilities

Usability Requirements:

Target users are casual game players. The app should be easy enough for the target users to play on their own, and also be appealing enough to attract repeated play. The app should be able to run on any iPhone models of any screen sizes.

Core Features:

Please note no game engine is permitted in this coursework. The core functions contain the features as shown in stage 1 and stage 2 showcase videos, including:

4. The app should be used in the landscape mode only.
5. The **shooter object**:
    a. The user can drag and move it.
    b. The movement is constrained within a square, which is located in the centre of the left screen boundary.
    c. When the user's touch ends, the shooter will reset to the original location and trigger the emergence of a ball object. An angle with coordinates will be calculated.
    d. Please watch Showcase Video Stage 1 for the effect of the shooter.
6. The **ball object**:
    a. The ball is released based on the direction that the shooter object has given.
    b. The left, upper, and lower screen boundaries are the collision boundaries for the ball.
    c. Balls may collide among themselves.
7. The **bird object**:
    a. It randomly appears on the right boundary of the screen.
    b. Once hit by a ball, the bird disappears.
    c. If there are more than one bird appearing, they should not overlay on top of each other.
8. Each game play should last no more than 20 seconds. When the game is over, a completion screen will be displayed with the score information and game reset button.

Development Strategy for Core Features:

Please note any form of game engine is not permitted in the development of this coursework.

There may be many approaches to implement this game. Here is one possible pathway for the essential development, which is highly recommended:

Stage 1.

1. **Position the shooter object.** Its initial position should be at the center of the left screen border, and it can be touched and dragged. The movement should be constrained into a square box area that is centered along the left boundary of the phone screen. The size of this square box should not be too big.
    a. Read Lecture 4 and Lab 4 for how to add a draggable image with constrained movement area.
    b. Start the project with screen-fit programming approach. Refer to Lecture 3 for the technique of screen-fit programming.

2. **Make the shooter object more real.** When the user releases the finger off the phone screen, the shooter should resume to be initial position. This can be coded within the `touchesEnd()` function.
    a. Read Lecture 4 and Lab 4 for how to program the draggable image class.

3. **Create the ball object.** Delegate needs to be used for this feature. When the user releases the finger off the phone screen, a ball object should be created within the main `ViewController` and appear on the phone screen.
    a. Read Lecture 5 for the definition of delegate.
    b. Read Lab 3 for adding an image view programmatically.

4. **Add dynamics to the ball.** Once the balls are created, they will move with linear speed (for the moment, just set up some random directions for the balls to travel). The balls will collide with each other if there are multiple ones on the screen. The balls should also collide with the top, bottom, and left boundaries of the phone screen.
    a. Read Lecture 6 and Lab 6 for using UIKit Dynamics to create motion with linear speed as well as collision with boundaries.

5. **Shoot the ball with a specific angle.** Set up a vector as global variables in the main `ViewController`, based on which the ball will move with linear speed at a certain direction. When user's touch on the shooter object has moved, the vector should be updated on live basis (this feature needs to be implemented using delegate). When user's touch on the shooter object has ended, a ball object should be created and linear speed motion.
    a. Read Lecture 5 for the definition of delegate.
    b. Read Lecture 6 for defining a vector for the direction of linear movement.

Stage 2.

6. **Create the bird object.** Only create one bird in a fixed location for now. When any ball object intersects with the bird object, the bird object should be removed, and one score should be gained.
    a. Read Lab 3 for adding an image view programmatically.
    b. Read Lab 6 for adding actions after collision.
    c. Read Lab 3 for removing a subview.

7. **Randomly create the birds.** The emergence of the birds should be random, and all position along the right border of the phone screen. The birds should not overlap with each other.
    a. Read Lab 5 for the use of timer.
    b. Read Lab 3 for the intersection of image views.
    c. Read Lab 2.2 for the use of arrays.
    d. Read Lab 2.3 for generating random numbers.

8. **Set a time-out for the game.**
    a. Read Lab 5 for the use of timer.

9. **Implement a Game-Over screen.** Enable replay.
    a. Read Lab 3 for how to make a view show/hidden.

Stage 3.

10. **Stretch abilities.** If you are feeling keen, confident, and devoted, you may wish to stretch your abilities and add some additional features to make your app more publish-worthy (but you will need to commit yourself with a decent amount of research efforts). Here are some potential ideas that you may wish to consider.
    - Create different levels of games. In each level, there is a random obstacle block placed somewhere in the screen, as shown in Figure 3. The ball will bounce off the blocks, and therefore need to bypass it in order to hit the birds. Watch the showcase video stage 3 for the effects.

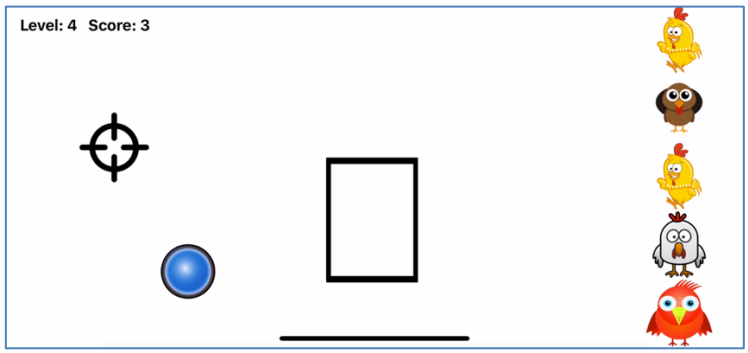

Figure 3. Obstacle Block in the Middle

- Add sound effects to increase the game appeal.
- Add more animation effects to create visual impact.
- Use of the built-in sensors and vibrators.

Production:

Your production should be managed by GitHub version control, as explained in Lecture 2. And refer to Tutorial 2 for how to create private repository. Upon completing the coursework, you need to produce a demonstration video of your app and submit it along with your source code. Please read *Canvas/Units/Assignment/Video Recording* for guidance of recording videos.

Usability Report:

You need to submit a written report containing the self-evaluation of the app you designed, reflecting the usability and future improvements. The word limit is 1000 words excluding references, and marginal excess is permitted. Please note this is a usability report, not a user manual. Refer to Lecture 8 for usability analysis.

Resources & Copyright:

This coursework aims to simulate a corporate environment for all students in order for them to gain valuable industrial experience. Imagine yourself work in a company where there is a dedicated graphic team to provide you with the image resources; and your task is to develop the app and to achieve the visual impact from the technical side, using the given resources.

The images used in the showcase video are available on *Canvas/Units/Assignment/images.zip*. Please feel free to use them. You do not have to implement the app that looks identical to the one in the showcase video, and you are welcome to use other graphic themes. Using images and audio files from other online sources is acceptable, but you should acknowledge the sources in the completion page. For example, using a small print in the corner of the screen to reference the source. Please bear in mind that this is NOT an art module, so please do NOT spend much energy on graphic design. The aesthetics of the app will NOT be judged.

Also note that your apps are strictly used for the internal study only, as some apps may incur potential copyright issues with images or sound sources. Under no circumstance should you be permitted to publish your apps in the public domain under the name of the University. (Please feel free to demonstrate your app during job interviews.)

Marks Breakdown:

Components	Marks
Functionalities and Reliability	65%
Usability Practice	20%
Development Practice	5%
Usability Analysis	10%
Total	100%

- **Functionalities and Reliability:** the programming exercise to achieve the required functionalities; reliability refers to the smooth and error-free running of the app
- **Usability Practice:** friendly user interface for the targeted user group; a highly-usable app for any iPhone models
- **Development Practice:** production management with version control
- **Usability Analysis:** self-reflection of the app design and implementation; discussion of future improvement

Detailed mark breakdown is shown below:

Components		Weight
Functionalities and Reliability (65%)	Shooter Movement	5%
	Shooting Balls	5%
	Ball Movement and Collisions	5%
	Random Emerging of Birds	5%
	Collision between Balls and Birds	5%
	Game-Over Enabled and Finishing View	5%
	Replay Enabled	5%
	Score Keeping	5%
	Smoothing Running of App	5%
	Stretch Features	20%
Usability Practice (20%)	User-Friendly Interface	5%
	Visual Impact	5%
	Engaging Factors of the Product	5%
	Screen Fit for All iPhone Models	5%
Development Practice (5%)	Regular Version Control	3%
	Well Documented Commits	2%
Usability Analysis (10%)	In-Depth Evaluation	2%
	Supporting Statements with Evidence	2%
	Future Work Suggestions	2%
	References	2%
	Academic Language and Formatting	2%
Total		100%

Please bear in mind this is a production process – if you do not have a product in the end, you will fail the project. You should rather have a simple app that works, than a sophisticated app that does not work. Failure to deliver a working product will fail the assessment:

- An app that fails to run due to poor file management (such as missing files) will receive a penalty of 5 marks. Excuses such as last-minute rush in submission or problems with file-zipping will not be accepted.
- An app that fails to run for various other reasons will be capped at 40% for marks.

UNIVERSITYOF HERTFORDSHIRE
School of Engineering and Computer Science

Grading Criteria

	Functionalities and Reliability (65%)	Usability Practice (20%)	Development Practice (5%)	Usability Analysis (10%)
1st	Outstanding use of appropriate technologies. Consistent and accurate execution. Full functions as required, with steps beyond expectations using sophisticated solutions. The app can run smoothly. No crashes or errors after repeated testing.	Visual impact. Flawless operations. Model compatibility. Proportionally sized displays.	Effective, frequent, and sustained use of management skills. Well-planned task cards on the Kanban. Version control is efficiently and effectively used. Agile management helps with the project. Regular management behaviours are evident.	Critical analysis. Evidence-based research. In-depth evaluation. Statements supported by evidence. Sensible suggestions for future improvements. Background research effort indicated by solid reference list. Proper academic writing and report formatting.
2:1	Appropriate use of technologies. Minor errors in technique and/or application with little impact on deliverables. Required components are functional with accuracy. The app can run smoothly for the first or second time, but there may be some errors or crashes after repeated testing.	Attempt on the visual impact. User-friendly and compatible. Minor setback on display. May work well on certain screens but not others.	Good efforts, with occasionally minor setback. Well managed project on regular basis overall. May be lack of a few activities here or there.	Effective analysis. Can benefit from minor tweaks. Commendable quality of work containing all relevant analysis and discussions. May be lack of critical analysis. A minor effort of improvements still required.
2:2	Appropriate use of technologies to the problem domain. Key functionalities implemented. Execution with occasional errors, giving minor impact on intended operations. Occasional crashes.	Lack of visual impact. Relatively easy to use. Minor compatibility and display issues.	Some attempt, but needs major improvement. Relatively large time gap found.	Shortage of research efforts. Satisfying writing overall. Satisfactory level of understanding. Proper concepts are demonstrated with evidence.
3rd	Genuine attempt on the technologies. Required functions are not partially completed. Delivery of a working product, but with notable errors that require major improvements.	No effort on the appeal. Not intuitive to operate. Inadequate visual display.	Claimed attempt with insufficient evidence. Only a couple of backup and project management attempts during the whole development process. Mismatched or doubtful submissions made.	Verbose and redundant. Lack of concrete content. Brief discussion on the subjects. Familiarity with the usability concept, but unable to form an evaluation.
Fail	Limited use of technologies. A non-working product. Fatal errors. Missing core components. Significant improvements are necessary. Academic misconduct.	Major problems with visual display and compatibility issues.	Low or little attempt on the management practice. No exercise towards agile management and version control. Fabrication of the process.	Inadequate content. Very brief text on the subject. Academic dishonesty.

UNIVERSITYOF HERTFORDSHIRE
School of Engineering and Computer Science

Interpretation of Grades (Newly Proposed for 2019/2020)

10-point numeric grade	Grade descriptor
95	Outstanding
85	Excellent
75	Very good
65	Good
55	Clear pass
45	Marginal pass
35	Marginal fail
25	Clear fail
10	Little or nothing of merit
0	Little or nothing of merit