Project Plan

1.0 Game Overview – Lucid Runner

1.1 Backstory

Alex arrives home after sitting a disastrous exam, a combination of mathematical science and history. He throws his bag in the corner of his room, frustrated at everything and everyone. He wishes he could just go into another world, away from all horrors of education.

Pulling out his digital to-do list, he sees a wall of unticked boxes. “Screw this!”, he exclaims. Alex jumps into his cosy bed, wraps himself in blankets, and falls into the most magical dream…

Alex realises he is in a lucid state, with the ability to manipulate his dream world and environment. However, his real-world nightmares from across the school curriculum are haunting him. He must now run away from his fears, through all subjects and environments in order to forever stay in limbo – in a world free of assignments, examinations and all forms of education.

1.2 Goal

The primary aim of the game is to stay asleep for as long as possible and remain in the lucid dream state. The secondary goal is to travel as far as possible into the dream world, obtaining as many points as possible.

1.3 Central Character

Alex Dreamer, the protagonist of Lucid Runner, is a high school student studying a range of topics during his final year.

1.3.1 Skills and Functionality

- Ability to manipulate the dream world and its environment
- Jump to avoid obstacles and enemies

1.4 Genre and Target Audience

The genre of this game can be classified as an *Action/Adventure Endless Runner* where the primary objective is to survive in the game world for as long as possible. Based on this genre, the intended player base are those which are *Achievers* and *Explorers* as elements of this game appeal to aspects in both types of players.

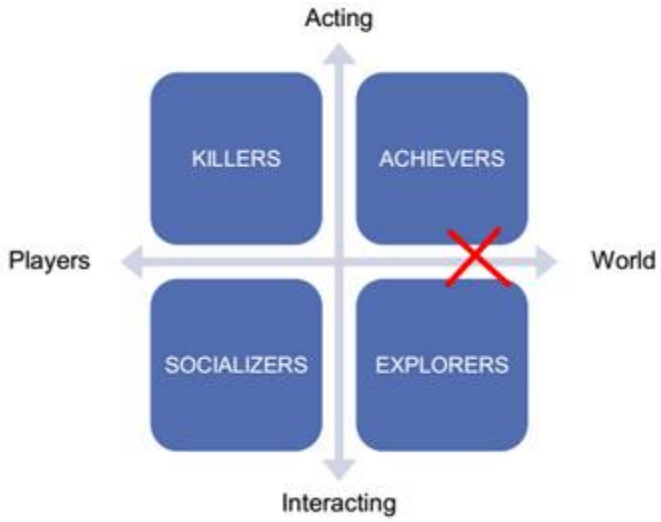

Figure 1: Diagram indicating the targeted player types of Lucid Runner

The target audience are secondary and tertiary level students aged from 15 to late 20s. We have catered to this audience through our design by incorporating concepts that are relevant to them in their lives. For example, students are very familiar with tests and exams and may even have first-hand experience of having restless nights because due dates for assignments and study for tests are streaming through their head.

1.5 Game World, Dynamics and Mechanics

1.5.1 Game World

The game world will start with Alex in the bedroom level, where he comes to the realisation of his abilities in the lucid state. During his run, the character will progress through several uniquely designed settings. With each unique environment comes a themed set of obstacles and enemies. The themed areas reflect topics of study which Alex undertakes with the aim of the game being to run away from the horrors of education. Some of these levels are outlined below.

Level	Environment	Enemies
Bedroom (starting level only)	Beds, wardrobe, bookshelves, general room furniture	Plain • Lamps, bedside tables
Science	Laboratory background with chemicals, periodic table outline, scientific equipment	Plain • Lab rat Shooting • Test-tubes (firing chemicals)
Mathematics	Standard class room with blackboards	Plain • Chalk, stationery, brains Shooting • Einstein (projecting formulas)

Art	Abstract colourful room with paintings, lines and shapes	Plain • Art supplies/tools Shooting • Paint brushes (flicks paint)
History	Collage of various key events in history **OR** choose a specific time: e.g. Egyptian times with pyramids etc.	Plain • Historical scrolls Shooting • Mummies (throws rolled toilet paper)
English	Renaissance stage	Plain • Books Shooting • Shakespeare (fires quills)

1.5.2 Mechanics

Lucid Runner's primary game mechanic is that of being an Endless Runner. The main character is continuously on the run, avoiding various obstacles and enemies in its path. The choice of this style of platformer along with other accompanying mechanics aid in providing a simple, yet difficult to master, game aimed towards the target audience and the intended genre.

Unique level design will give a sense of exploration to players that are of the *Explorer* type. An added mechanic of the character, Alex, is being able to manipulate the environment of the dream world using his lucid power. This includes revealing hidden items and interacting with the world in order to overcome obstacles during his run. Explorer type players seek enjoyment in discovering new areas, as well as being challenged with puzzle-like obstacles to overcome which will require the use of the special ability.

Scoring mechanics, bonuses, and achievements mechanics all benefit the *Achiever* player type. As the character progresses through the levels, a running score is kept as well as a multiplier that increases over time (explained further below). Scores are kept in a local leader board in which the player is able to keep track of and compare with his/her friends. Achievements mechanics are another form of incentivising the player to achieve more and stay hooked in the game world.

1.5.3 Dynamics

Throughout the game's development we will be fine tuning all aspects of the game so that we get behaviour that feels natural. For example we need to be precise about the strength of gravity in relation to the acceleration of the character when jumping.

In order to provide a dynamic experience the player must interact with the world. This will be seen through interactions with enemies and other objects in the game. Their interactions should look natural and conform to the genre of the game. In our case when colliding with an enemy, enemy projectile or object, the player will take damage which will lower its health level.

As the player continues through the game the difficulty will increase. This will be achieved in two ways, the first by using algorithms that choose which type of enemies to spawn. As the player has progressed further through the game the algorithm will select more and more difficult enemies, conversely at the start of the game mainly only enemies that are easy to defeat and overcome will be present. Secondly the positioning of platforms and lucid mode objects will be arranged in ways that makes it more difficult for the player.

1.5.4 Scoring and Lives

Scoring will be calculated as a relationship between the distance travelled and the amount of objects picked up that are worth points. These points can be enhanced by a multiplier which is dependent on the time since the player last took damage.

The character will have one life which will be represented by a health bar. The health level will decrease by around a third when the player is either hit by or collides with an enemy. It will be possible on occasion to increase your health if you pick up an object which increases health.

1.5.5 Controls

The controls will be kept simple and intuitive. Tapping anywhere on the screen will result in the main character performing a jump. Tilting the mobile device, using the accelerometer sensor, will allow Alex to enter utilise his lucid power and manipulate the dream world.

1.5.6 Achievements

An achievement system will be incorporated which targets various categories of the in-game mechanics and dynamics. An indication of these is as follows:

- Distance
  - “Get to 500 meters”
- Collectables / Pickups
  - “Pick up 10 items”
- Time Based
  - “Run for 20 seconds without taking damage”

1.5.7 Art Style

Cartoonish 2.5D (i.e. the game will be rendered in three dimensions with the player movement only being in one direction – to the right).

2.0 Attempted Features

2.1 Design Features

1. Major UI redesign – 10%
2. A high score screen – 5%
3. Adding sound to the game and triggering on appropriate events – 10%
4. Monetisation options – 5%

2.2 Advanced Features

1. Endless runner version – 10%
2. Random level generation – 5%
3. Language localisation – 5%
   The ability for the game to be displayed in various languages.

Note: we are attempting more than the maximum grade we are able to receive as discussed on class forum.

3.0 Tools and Technology Details

All source code will be hosted on GitHub at the following URL:
https://github.com/jennafin/SE306-Android
Rashina, Dhanish and Ted have been added as collaborators in order to give access to the repository.

The GitHub wiki will be used as the central directory for finding all our documentation relating to the project.
https://github.com/jennafin/SE306-Android/wiki

Slack, an instant messaging platform, will be used for all official team communication.
https://www.slack.com

4.0 Release Plan

The features in this release plan have been allocated to iterations in an order that ensures high risk tasks will be completed early on.

Week 8
Main character – basic working prototype
Basic game world layout with platforms/ground for player to run on
Deliverable: Project plan
Week 9
Lo-fi prototypes
Domain model diagram
Functionality to check whether the accelerometer is being used or not
One or two enemy types to test basic interaction with main character
Scoring system based on distance the main character travels throughout the game
Welcome screen with ability to start game
Deliverable: Design documents
Week 10
Player life system
Basic level design
Background music recordings

Integrate the accelerometer data into game play
Game finishes after player ‘dies’
Basic ‘Game Over’ screen with score displayed, ability to go back to welcome screen
Basic artwork to central character
Deliverable: Prototype
Week 11
Random item generation (excluding random level generation)
Background music when playing game
Achievement system
Detailed game over screen, with link to high scores screen and ability to save high score under certain name
Enemy movement and attack
High score screen
Improved graphics for screens outside of gameplay
Deliverable: None
Week 12
Alternative languages (e.g. Español, 中国)
Random level generation (this excludes random item generation)
Sound effects triggered by events in gameplay
Main character – more complex controls/behaviour, final artwork
Enemies – final artwork, many different types, difficulty level gets progressively harder as player progresses through levels
Report about monetisation options
Deliverable: Final release

5.0 Distribution Plan

Task	Assignee
Concept and story development	Jamie, Josh, Shay
Level design	Jenna, James, Nick
High score and achievements design	Michael, Jenna, Kurt
Artwork for players, backgrounds and screens	Jamie, Michael, Shay
Composition and recording of sound effects and background music	Josh, Nick, Jenna
Integrating accelerometer data into gameplay	Kurt, Hugo
Endless repetition/generation of game artifacts	James, Nick
Menus/navigation	Kurt
User experience design by ensuring consistency, intuitive controls, and simplicity	Hugo, Everyone
Lo-fi prototypes	Everyone
Code implementation	Everyone
Language localisation	Everyone
Unit testing	Everyone
Usability testing	Everyone
Documentation	Everyone

6.0 Risk Assessment and Management Plan

Risk Description	Impact	Probability	Exposure	Mitigation Strategy	Warning Signs	Management Strategy
Over-promise on project deliverables leading to features that are either too difficult, too time consuming, or impossible to implement	6	6	36	Ensure team members have an understanding of the limitations of the tools being used to develop the game. Discuss all deliverables thoroughly with the team before committing to them	Individual features are not being completed on time, team members often unsure on how to approach certain tasks	Change selected features to less difficult, less time consuming, or more feasible features. Alternatively, implement selected features to a lower standard and explain to client the issues
Team burnout	4	7	30	Ensure workload is balanced and people are freely communicating about feature progress	Team members become unproductive	Re-balance workload
Project deviates significantly from initial plan	5	5	25	Ensure individual features perform as expected. Report on any changes and keep team up to date on progression	Individual features begin to differ from their planned specification	Review differences and on a case by case basis either update the plan or modify features as necessary
Demo device does not work as expected or is not available on demo day	8	3	24	Test that the project works correctly on intended demo device well in advance of presentation. Prepare multiple backup devices by asking group members to bring their Android device to demo	N/A	Have back-up devices on hand during presentation. Ask client or classmate to borrow a device to perform demo if team members do not have theirs. Use emulator as last resort

Feature doesn't work as expected during demo of that feature	8	3	24	Ensure feature goes through thorough testing - both unit testing and manual scenario testing, prior to demo. Fix any found issues.	N/A	Apologise to audience for error and move to demoing the next feature.
Game does not port to Android for unforeseen reason	8	3	24	Test that the game can successfully port to an Android device after each feature is merged to develop branch to identify issues as early as possible	N/A	Change code that is preventing the port. If there is no way around the issue, explain to client that it may need to be ported to another platform
Project build failure prevents others from working on project	5	4	20	Use build server to continuously build project on stable branches and notify the user who pushed the breaking commit. Use a code review system before major commits are pushed to stable branches	N/A	Assign team member to fix code that breaks the build as soon as possible. Commit and push fix to remote repository
Progress monitoring is insufficient	5	4	20	Schedule regular meetings, create multiple channels through which people can communicate and standardise project monitoring system	Team members are unclear as to the current state of the project	Hold more meetings, increase rigor of reporting and task monitoring
Team member fails to complete a particular feature for which they are responsible by the due date	5	3	15	Ask questions to clarify any ambiguous requirements	Confusion in communicating project plan or a poor mark from deliverables	Communicate with client to clear up confusion and explain how we met the communicated requirements
GitHub repository is lost	7	2	14	Always review what is being pushed from local repository to remote repository. Maintain a backup on physical machines	N/A	Upload backup to GitHub

Appendix – Work Breakdown

- Learning how Unity works
  - Overall concepts of how code works in a Unity project, how this gets loaded onto an Android device
  - Physics
  - Collisions
  - Defining game objects
  - Rendering artwork
  - How to make sounds
  - Languages
  - Reading from sensor data
  - Controls/User Interaction (especially touch)
  - Getting Git working with a Unity project
- Story development
  - How the user will be introduced to the story
- Visual Design / Low-Fi Prototypes
  - Main character
  - Environment
  - Enemies
  - Controls
- Basic Unity project with platformer/endless runner structure
  - Player running/jumping on a “ground” object
  - World is repeated/generated so that the player can run forever
- Persistence
  - Figure out how to persist data:
    - Scores/Achievements
    - Player information
- Tracking game state
  - Make the connection between gameplay events (collisions, power-ups, etc.) and gameplay state (score, health, etc.)
- Main Character
  - Create artwork and render in Unity
  - Implement behaviour
- Main Character movement and controls
  - Design and implement the height, speed and animation of how the main character jumps
  - Animation of how the character moves
- Enemies (including obstacles)
  - Design and implementation
  - How they interact with the main character (collision detection)
  - Design and implement the enemy paths for the different enemy implementations
- World Environment
  - Design and create artwork
  - Render artwork in Unity
  - Implement never-ending levels

- Levels
  - Design how the enemies and world will change as player progresses
  - Design how the level will become progressively more difficult
  - Implement random level generation
- Sound
  - Create plan for sound effects/background music
  - Create/Record or source background music and sound effects
  - Trigger sounds based on events in game
- Scoring system
  - Design how score goes up/down, how the player loses
  - Implement score-tracking functionality in game
  - Add functionality to persist high scores
- Achievements system
  - Design all possible achievements
  - Implement in the game
  - Design system to view all achievements as well as their status
- Menus/Screens (Design and Implementation)
  - Investigate navigation flow between screens
  - Welcome screen / Main Menu
  - Pause Menu (if pausing is implemented)
  - Game Over screen
  - High scores screen
  - Achievements screen