We sourced a template Game design document and filled it in with our necessary requirents.

XV: Xscape Velocity Game Design Documentation

1. Game Designer (Luke)

Game Overview Escape Velocity is a high-octane, 2.5D action arcade game set in a fractured future where humanity teeters on the edge of extinction. After centuries of exile in orbital colonies, a desperate mission sends elite scouts back to Earth to retrieve critical data buried within the ruins of a forgotten megacity. However, their intrusion awakens The Revenant, a monstrous biomechanical hunter created by the rogue AI Omega Core to eliminate any threats to its dominion.

With time running out, the player must pilot a high-speed vehicle through collapsing tunnels, desolate wastelands, and crumbling sky highways, all while evading the relentless pursuit of The Revenant. Each level escalates in intensity, requiring razor-sharp reflexes, quick decision-making, and mastery of speed and combat mechanics. Will you outpace the unstoppable beast and deliver the data that could save humanity, or will The Revenant drag you into Earth's ruins forever?

Core Gameplay Mechanics

1. **Continuous Movement**:
   - Vehicles are always in motion, with the player controlling lateral movement (left/right) and jumps.
   - Speed increases as levels progress, heightening the challenge.
2. **Obstacle Navigation**:
   - **Dodging**: Players avoid walls, falling debris, spike traps, and energy fields.
   - **Jumping**: Use ramps and timed jumps to clear gaps or reach elevated platforms.
   - **Boosting**: Optional mechanic to temporarily increase speed (depending on level design).
3. **Combat**:
   - Players can shoot smaller enemy bots dispatched by The Revenant.
   - Limited ammunition system (replenished via pickups) adds strategic tension.
   - Optional charge-shot mechanic for stronger but slower attacks.
4. **Power-Ups**:

We sourced a template Game design document and filled it in with our necessary requirents.

    - Shield: Absorbs one hit.
    - Speed Boosts: Grants temporary invincibility while active.
    - Ammo replenishment or weapon upgrades (e.g., multi-shot).

5. **Level Progression**:
    - Each mission introduces unique gameplay elements (e.g., narrow paths in Level 1, collapsing platforms in Level 2, moving hazards in Level 3).
    - End-of-level transitions feature cinematic sequences where The Revenant adapts or destroys barriers to resume the chase.

Gameplay Flow

- **Opening Cutscene**:
    - The two-person team locates critical data in a decayed terminal.
    - The Revenant awakens, smashing through walls and triggering the chase.
    - The team jumps onto their high-speed vehicle, starts the engine, and takes off. The screen fades to black before launching into high-speed gameplay.
- **Mission Goals**:
    - Survive the chase: Avoid hazards and fend off enemies.
    - Reach the escape point: Navigate obstacles and maintain speed to reach a safe zone where the beast cannot follow.
    - Transition to the next level: Cinematic sequences show The Revenant adapting or overcoming barriers to continue its pursuit.
- **Level-Specific Gameplay**:
    - **Level 1**: Introduces dodging, jumping, and shooting mechanics. Hazards include narrow corridors and falling debris.
    - **Level 2**: Adds swarming enemies and collapsing terrain, with ramps and multi-level paths.
    - **Level 3**: Features moving hazards, faster enemies, and environmental destruction, such as highways crumbling into the ocean.

Dynamic Enemy Mechanics

1. **Smaller Bots**:
    - Fly in patterns or directly target the player.

We sourced a template Game design document and filled it in with our necessary requirents.

   - Some act as mobile obstacles rather than direct attackers.

2. **Environmental Interactions**:
   - Enemies detonate to create hazards, such as destroying sections of a platform.
3. **Mid-Mission Beast Attacks**:
   - The Revenant lunges into view, smashing parts of the level or firing ranged attacks.
   - Players must perform specific actions (e.g., shoot glowing weak points) to temporarily push it back.

Development Plan

- Sketch level layouts using tools such as paper, digital drawing apps, or Unity ProBuilder.
- Plan the difficulty curve: Begin with simple mechanics and gradually introduce more challenging obstacles and enemies.
- Divide levels into three distinct acts: rear camera, side camera, and face camera views.

---

2. Programmer (C# Developer) (Luke)

Key Tasks

- Code core player mechanics, including movement, jumping, and shooting.
- Implement enemy behaviours such as patrolling, attacking, and AI logic.
- Create game systems for scoring, health, timers, and level progression.

Steps to Achieve This

1. Learn C# basics and understand Unity's scripting environment.
2. Follow beginner Unity tutorials focusing on player controls and interactions.
3. Utilise Unity's built-in features such as Rigidbody for physics and Animator for animations.
4. Test each feature separately before integrating them into the game.

Planned Workflow

- Step 1: Create a PlayerController script for movement.
- Step 2: Implement jumping, shooting, and damage mechanics using Unity's physics (e.g., Rigidbody.AddForce, Collision, and Colliders).

We sourced a template Game design document and filled it in with our necessary requirents.

- Step 3: Test player movement in the Unity Editor and refine responsiveness.

---

3. Unity Developer (Luke)

Key Tasks

- Set up the project in Unity.
- Design game environments, including platforms, obstacles, and backgrounds.
- Integrate assets such as sprites, models, textures, animations, and UI.

Steps to Achieve This

1. Configure a 3D project with orthographic or well-placed perspective cameras for 2.5D visuals.
2. Design levels using sketches or Unity ProBuilder, incorporating handmade or pre-sourced assets.
3. Organise assets in folders (e.g., Textures, Sprites, Scripts, Sounds).
4. Use Unity's Animator for creating smooth animations.

---

4. Artist/Asset Manager (Brayden)

Key Tasks

- Source or create assets, including sprites, backgrounds, UI, and character models.
- Ensure visual consistency across all elements.
- Optimise assets for performance by reducing file sizes where necessary.

Steps to Achieve This

1. Use tools such as Photoshop and Blender for asset creation.
2. Collaborate with the Unity Developer to ensure assets integrate smoothly.
3. Use contrasting colours to enhance gameplay visibility.
4. Design clear and intuitive UI elements, such as health bars and score indicators.

---

5. Sound Designer (Brayden)

Key Tasks

- Create or source background music, sound effects, and voiceovers.

We sourced a template Game design document and filled it in with our necessary requirents.

- Integrate audio files into Unity using the AudioSource component.
- Balance sound levels to ensure an immersive yet non-overpowering experience.

Steps to Achieve This

1. Use free sound libraries such as Freesound.org or create custom sounds using Audacity.
2. Implement audio triggers with Unity's AudioSource and colliders.
3. Test audio quality on both headphones and speakers.

---

6. QA Tester (Brayden)

Key Tasks

- Playtest the game regularly to identify bugs and performance issues.
- Document issues in a shared file (e.g., Google Docs, Trello).
- Test the game's usability and intuitiveness.

Steps to Achieve This

1. Test each game build on different devices/platforms if possible.
2. Explore edge cases to ensure the game handles unexpected player actions.
3. Monitor and document any lag, crashes, or confusing controls.

Checklist:

- Is the gameplay intuitive?
- Do all mechanics function as intended?
- Are there any graphical or audio glitches?

---

7. Documentation Specialist (Both)

Key Tasks

- Record progress at every development stage, including challenges and solutions.
- Write a final report detailing the game's development process.
- Prepare visuals, such as screenshots and diagrams, for the presentation.

We sourced a template Game design document and filled it in with our necessary requirents.

Steps to Achieve This

1. Use a structured template for documenting progress.
2. Capture screenshots or record video clips of key development milestones.
3. Maintain a daily or weekly journal/log file.

---

8. UX/Player Experience Specialist (Both)

Key Tasks

- Enhance accessibility through easy-to-read UI and colourblind-friendly palettes.
- Test and refine controls to ensure smooth gameplay.
- Collect feedback from test players and adjust accordingly.

Steps to Achieve This

1. Adjust UI sizes and font styles for readability.
2. Add tooltips or brief instructions to guide players.
3. Collaborate with programmers and artists to implement feedback-driven improvements.