TDD (Technical Design Document)

GeeksInstitute - LaStartupStation

Project Name: Tbourida

Name	Role
Abdellah Aoukrad	Gameplay Programmer/Game Designer/Producer
Zakaria Rezki	Graphic Designer / 3D / 2D
Soufiane Mjadi	Graphic Designer / 3D / 2D
Yassine Ait HMAD	

Version Table

Version No.	Date	Author(s)	Description
1.0	2025-12-18	Yassine Ait Hmad	Draft TDD with same sections as the game idea docs.
2.0	2025-12-23	Abdellah Aoukrad	Final TDD with proper TDD sections.
3.0	2026-01-08	Abdellah Aoukrad	Alpha version TDD.

Version No.	Date	Author(s)	Description
4.0	2026-01-25	Abdellah Aoukrad	Gold version TDD.

---

1. Game Presentation

Tbourida is a 3D rhythm-simulation game featuring a stylized low-poly aesthetic. The technical scope has evolved from a basic prototype to a complex coordination system.

- **Core Focus:** "Sorba synchronization," where AI agents follow a player-leader using sophisticated damping and formation logic.
- **Optimization:** Highly optimized low-poly assets ensure high visual clarity and stable performance (10+ riders plus dense crowds) without degradation.

---

2. Platforms and Hardware Specifications

- **Target Platforms:** PC (Windows 10/11) and Mobile (Android API 26+, iOS 13+).
- **Performance Targets:** Strictly optimized for **60 FPS** on entry-level mobile devices via aggressive polygon reduction and optimized URP shaders.
- **Input Handling:** Unity Input System (WASD/Arrows for steering; Spacebar for shooting).

---

3. Development Environment

- **Game Engine:** Unity 6.3 (6000.3.1f1 LTS).
- **Rendering Pipeline:** Universal Render Pipeline (URP).
- **Version Control:** Unity Version Control.
- **3D Workflow:** Blender 4.5.5 with custom automation scripts.

---

4. Software Architecture & Game Systems

The game utilizes a decoupled, manager-based architecture coordinated by a central state machine:

Core Managers

- **GameManager:** Drives high-level GameState (Lineup, Countdown, Playing, Shop, etc.). Handles session transitions and game mode management.
- **SessionManager:** Manages persistence (JSON/PlayerPrefs) for credits, equipment, and career progress.
- **RoundManager:** Defines track/scoring zones. Calculates scores based on alignment, speed, and distance from the optimal line.
- **ArenaManager:** Handles loading of `ArenaData` ScriptableObjects and environment transitions.

Gameplay Engines

- **MovementEngine:** Physics-based controller for acceleration, braking, and steering clamp logic.
- **SorbaManager:** Handles AI spawning and formation offsets. Uses smooth damping for natural following and captures high-precision timestamps for scoring.
- **ShootingEngine:** Manages the "Baroud" phase, muzzle flash VFX, and synchronization recording.
- **AnimationEngine:** Drives horse/rider states (Idle, Run, Brake, Shoot) with randomized idle variations.
- **CameraEngine:** Manages four distinct modes: Follow, Shooting (side-view), Shop (fixed), and Lineup (cinematic).

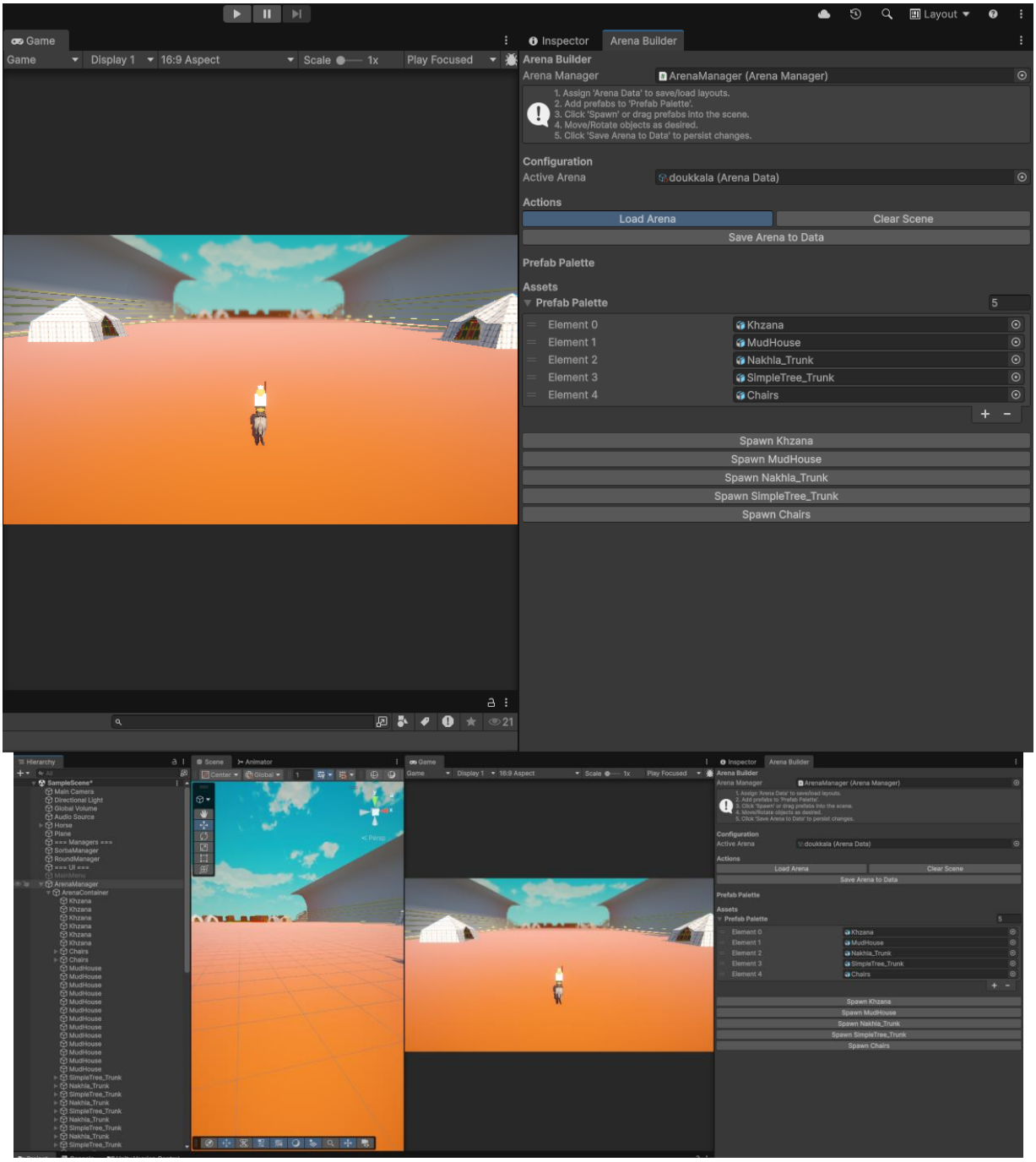
Progression & Minigames

- **ProgressionManager:** Tracks career mode across five regions, including opponent generation and region unlocking.
- **FormationRecoveryMinigame:** A timing-based challenge triggered when formation breaks, featuring adjustable difficulty scaling.

---

5. Custom Tools Developed

- **ArenaBuilderWindow:** Editor tool for spawning prefabs from a palette and saving track layouts.
- **Blender Export Script:** Python automation for exporting FBXs directly to Unity with auto-mapped materials.
- **ForwardAxisFollower:** Utility for particle alignment, ensuring VFX stay focused on the horse’s forward velocity.

- 

6. Graphic Rendering & Optimization

- **Asset Counts:** Models optimized for low-poly counts (e.g., houses < 100 triangles).
- **Crowd System:** Optimized "billboard" and low-poly models with alpha textures to simulate thousands of spectators.
- **Materials:** Custom textures for the **Khzana** (tent) and local vegetation (Nakhla trees).

---

7. Menus & HUD

- **UI Technology:** Built with **Unity UI Toolkit (UIDocument)**.
- **HUD Elements:** Real-time alignment indicator, speedometer, and recovery minigame overlay.
- **In-World UI:** Start, Optimal, and Finish lines rendered as 3D meshes to reduce overlay clutter.

---

8. Physics & Gameplay Logic

- **Lineup & Countdown:** Cinematic transition into a 3-second countdown triggered by player input.
- **The Drift:** Perlin Noise function generating lateral forces that require active steering.
- **Scoring Math:** Total rating (0–9 points) based on:
    1. **Shot Accuracy** (Line distance)
    2. **Alignment Quality** (Standard deviation)
    3. **Speed Rating**
- **Synchronization:** AI agents fire with a randomized delay (10ms–50ms) to simulate human variation.

---

9. Audio System

- **Dynamic Audio:** Velocity-mapped hoofbeat loops.
- **Triggers:** High-impact "Baroud" gunshot samples and crowd "Zaghroutas" (ululations) triggered by performance success.