Sound Feature

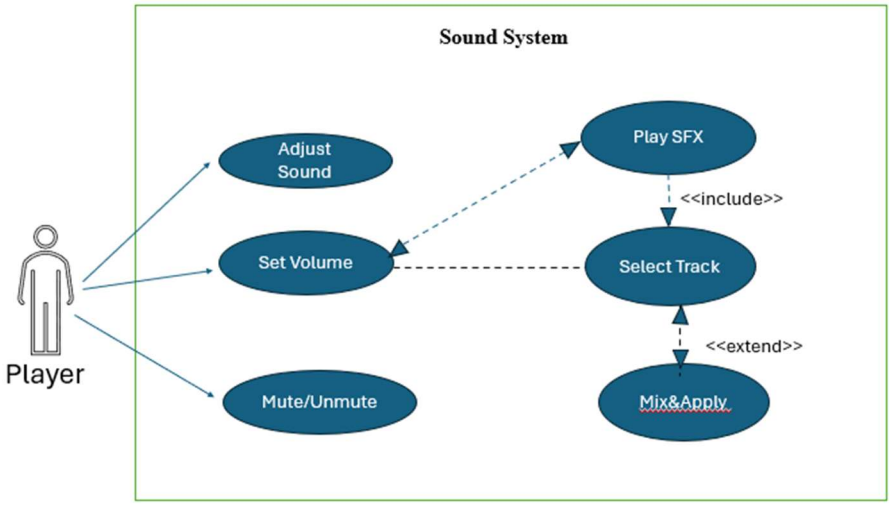
Name: Urvashi Gupta

1. Brief introduction __/3

The **Sound Feature** dynamically adjusts background music, sound effects, and volume based on player progress and in-game events. It maintains immersion and provides appropriate audio cues (tension, calm, reward). Sound scaling includes:

- Increasing music intensity during boss fights or high difficulty.
- Triggering reward jingles for pickups/unlocks.
- Adjusting ambient sounds to match level complexity.

2. Use Case Diagram with Scenario



Scenario: Adjust Sound (ID: S02)

Summary: The system adjusts music, sound effects, and volume when the player progresses or encounters key events.
Actors: Player, Game Engine, Sound System
Preconditions:

- Player has entered or completed a level.

- Sound System is active.

Basic Sequence (numbered so exceptions can reference steps):

1. Players enter a new level or triggers an in-game event (e.g., boss spawn, exploration zone, power-up).
2. System detects current context (performance metrics, event type).
3. System selects the appropriate background track and SFX set (or fallbacks).
4. System applies audio changes with smooth transitions (fade, crossfade, volume mix).
5. Sound System monitors audio playback and listens for new triggers.

Exceptions (numbered and linked to the step they relate to):

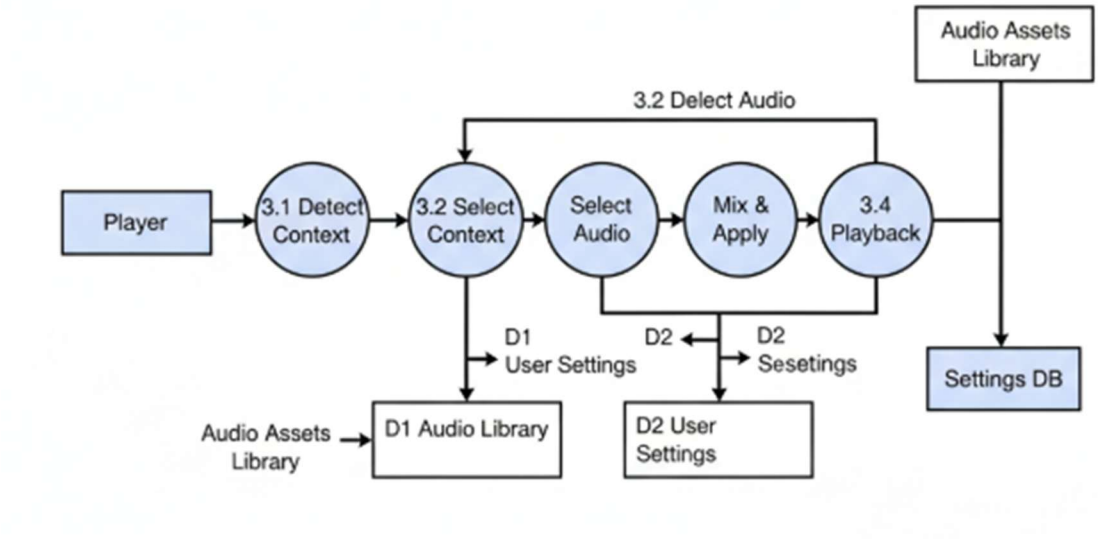
- **2a (relates to step 2):** Context data missing or corrupted → System uses last-known context or defaults to exploration.
- **3a (relates to step 3):** Required audio file missing or invalid → System loads default_track.mp3 and logs an error.
- **4a (relates to step 4):** Player toggled Mute during transition → Automatic transition aborted; manual override persists until unmuted.

Postconditions:

- Player experiences audio appropriate to context; any exceptions have fallback behavior applied.

Priority: 2 (Should have)

3. Data Flow Diagram(s) (Level 0) + Process descriptions



Process 3.1 — Detect Context

- Input: player event stream, performance metrics (score, health, time)
- Output: context object (mode: boss/explore/combat, performance rating)
- Activity: validate incoming data, tag event type, forward context to 3.2

- **Process 3.2 — Select Audio**
  - Input: context, Audio Library (D1), User Settings (D2)
  - Output: selectedTrack, sfxSet, target volume levels
  - Activity: apply selection rules (e.g., if boss → choose intense track), fallback to defaults if missing
- **Process 3.3 — Mix & Apply**
  - Input: selectedTrack, sfxSet, user volume
  - Output: final audio mix (channels, levels), transition plan (fade durations)
  - Activity: compute crossfade curves, apply ducking for voice/sfx as needed
- **Process 3.4 — Playback**
  - Input: audio mix stream
  - Output: audio to output device / speakers
  - Activity: stream to audio API; monitor playback health and stalls

Pseudocode (compact)

```
WHILE game_running:
  context = DetectContext(playerEvents, metrics)   # 3.1
  if userSettings.muted: continue
  track, sfx = SelectAudio(context, AudioLibrary)  # 3.2
  mix = MixAndApply(track, sfx, userSettings)      # 3.3
  Playback(mix)                                    # 3.4
END WHILE
```

---

4. Acceptance Tests __9

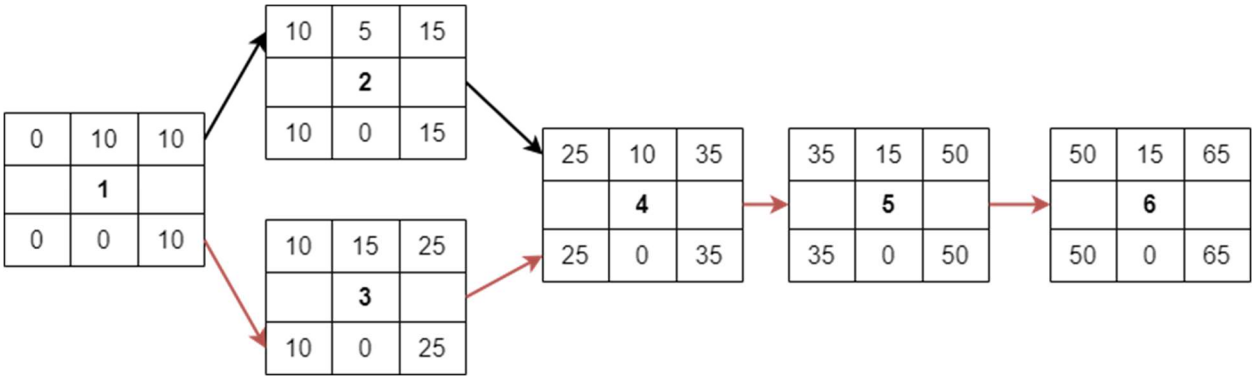
Test ID	Input (Player/Event Data)	Expected Output	Notes
T01	Boss fight triggered	Intense music fades in; combat SFX amplified	Stress test
T02	Player in idle zone	Calm/ambient track plays softly	Normal case
T03	Player collects power-up	Short reward jingle plays immediately	Feedback cue
T04	Player mutes audio	No sound plays; manual override persists	User setting
T05	Invalid file path for music	Default soundtrack loaded; error logged	Exception handling

---

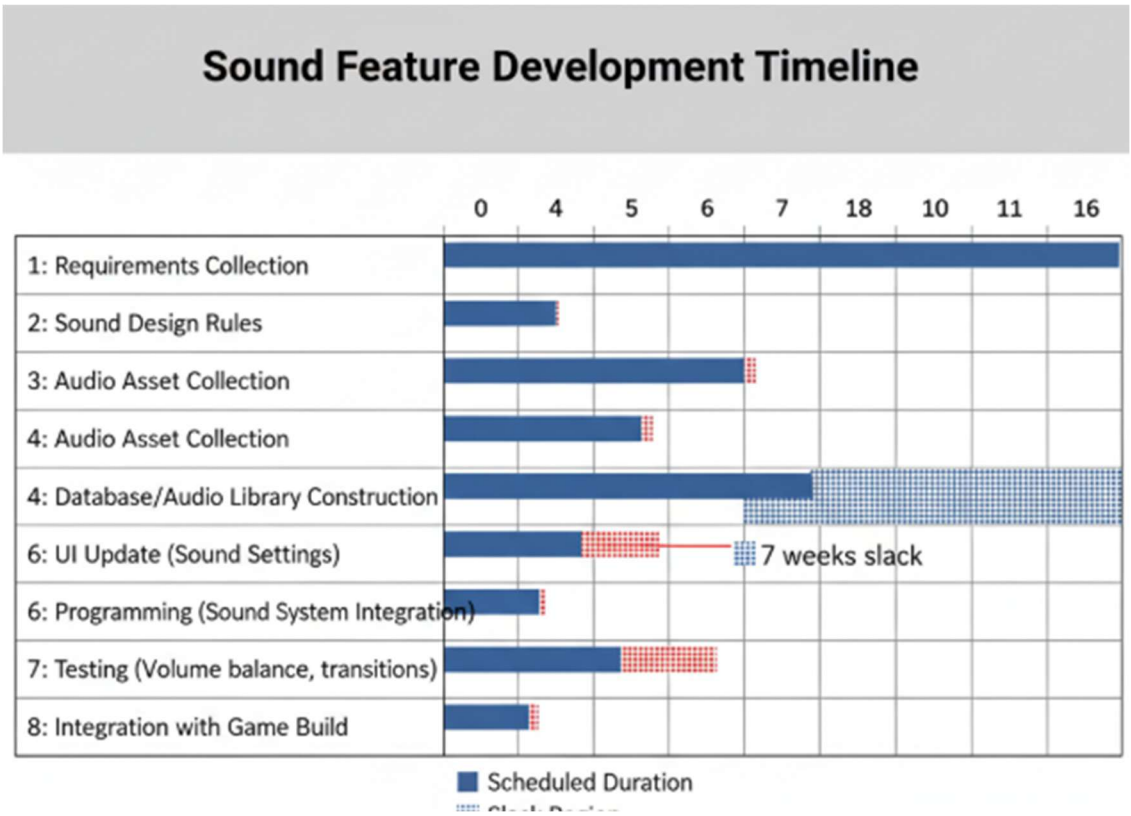
Work items

ID	Task	Duration (weeks)	Predecessor(s)
1	Requirements Collection	2	-
2	Sound Design Rules	3	1
3	Audio Asset Collection	3	1
4	Database/Audio Library Construction	2	2, 3
5	UI Update (Sound Settings)	2	4
6	Programming (Sound System Integration)	4	4
7	Testing (Volume balance, transitions)	3	6
8	Integration with Game Build	2	7

Pert Diagram



Gantt Chart


Sound Feature Development Timeline
0
4
5
6
7
18
10
11
16
1: Requirements Collection
2: Sound Design Rules
3: Audio Asset Collection
4: Audio Asset Collection
4: Database/Audio Library Construction
6: UI Update (Sound Settings)
7 weeks slack
6: Programming (Sound System Integration)
7: Testing (Volume balance, transitions)
8: Integration with Game Build
Scheduled Duration