US1 – Movie Registration

User Story

As a cinema administrator
I want to register a movie with title, genre, and duration,
So that it is available in the system

Acceptance Criteria

1. The system must allow entering title, genre, and duration (in minutes).
2. Genre must be chosen from a predefined list.
3. Duration must only accept numeric positive values.
4. Title field cannot be left empty.
5. Confirmation message displayed after successful registration.
6. Data is stored in the database.

Agile Process – Product Discovery

Problem Discovery

- **Identified Problem:** Administrators currently rely on manual spreadsheets or informal notes to register movies, leading to data errors and inconsistencies.
- **Impact:** Time wasted on duplicate work, difficulty maintaining a clean catalog, higher probability of human error, and delays in publishing new content for users.

User Need

- A simple, intuitive form for registering movies with validations.

Proposed Solution

- Frontend form with validation for mandatory fields.
- Backend API to persist new movies.
- Immediate confirmation.

Expected Prototype

- **Main Panel:** Admin panel form with fields (title, genre dropdown, duration numeric input).
- "Save" button with validation feedback.

Validation

- Success: Register 10 movies in < 3 minutes.
- Error rate < 5%.
- Movies update immediately in movies view.

- 100% conflict detection.

Business Value

- Faster onboarding of titles.
- Clean data.
- Reduced administrative errors.

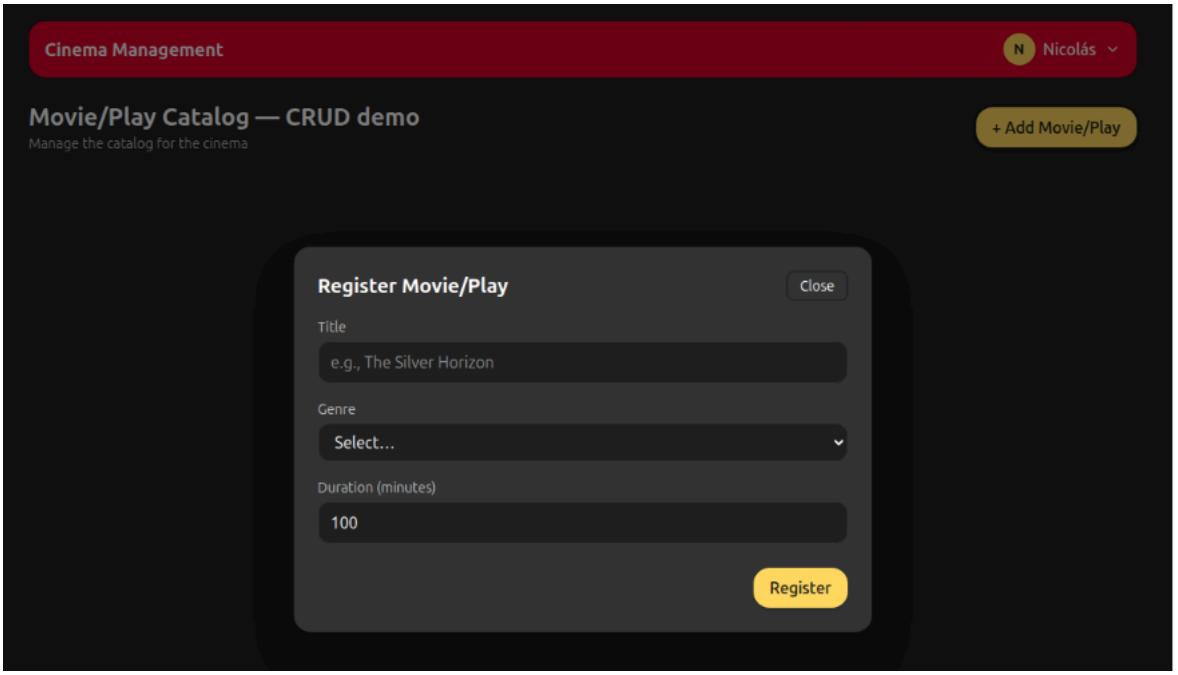

Figure 1. Mockup Function 1

Definition of Done (DoD)

- Code reviewed and merged.
- Unit/integration tests (>80% coverage).
- Frontend/backend validations.
- API documentation updated.
- Usability test passed.
- Deployed to staging.

US2 – Movie Consultation

User Story

As a user
I want to view the list of available movies or plays,
So that I can decide which show to attend.

Acceptance Criteria

1. The system displays all registered movies in a list.

2. Each item shows title, genre, and duration.

3. The user can search by title.

4. The user can filter by genre.

5. The list updates dynamically without reloading.

6. Public catalog is read-only.

Agile Process – Product Discovery

Problem Discovery

- **Identified Problem:** Users lack a reliable and up-to-date way to view available titles. Information is often inconsistent or requires asking staff directly.
- **Impact:** Frustration for users, increased workload for cinema staff answering queries, higher chance of abandoned visits, and lower overall satisfaction.

User Need

- Quick and clear access to available movies.

Proposed Solution

- Public-facing catalog view.
- Search box and genre filter.
- Responsive design for desktop and mobile.

Expected Prototype

- Public catalog showing cards with title, genre, duration, and “View Details”.

Validation

- Success: Retrieve desired movie in < 20 seconds.
- > 90% success rate in user test.
- 100% conflict detection.

Business Value

- Improved user experience.
- Fewer queries to staff.
- Increased engagement.

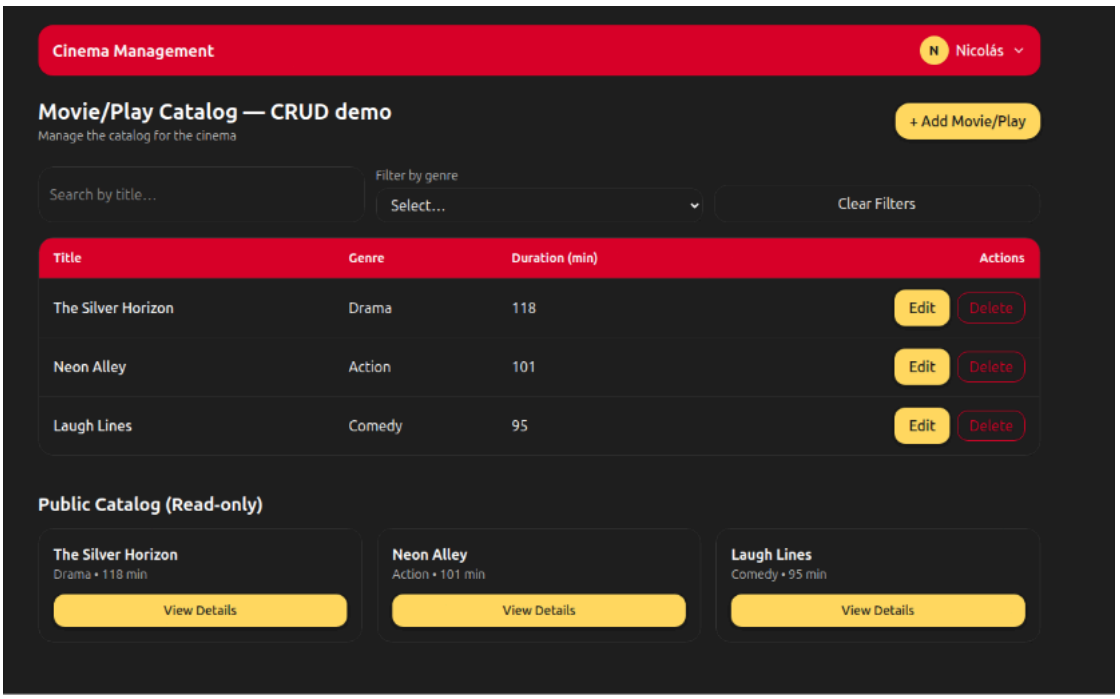

Figure 2. Mockup Function 2

Definition of Done (DoD)

- Code reviewed and merged.
- List view connected to database.
- Search and filter functions implemented.
- UI/UX validated with user testing.
- Unit/integration tests > 80%.
- Deployed to staging.

US3 – Movie Modification

User Story

As a cinema administrator
I want to edit the information of a movie or play,
So that the catalog remains up to date.

Acceptance Criteria

1. The system must allow selecting a registered movie for editing.
2. Title, genre, and duration can be modified.
3. Validation rules identical to registration apply.
4. Confirmation message shown after update.
5. Data persisted in the database.

6. Changes immediately reflected in the catalog.

Agile Process – Product Discovery

Problem Discovery

- **Identified Problem:** Once registered, movies cannot be updated efficiently, forcing administrators to create duplicate entries or leave outdated information.
- **Impact:** Catalog contains errors and redundant records, confusion in schedules, wasted storage, and reduced trust in system data.

User Need

- Update titles quickly without duplicating records.

Proposed Solution

- "Edit" button in the admin catalog.
- Prefilled form with current values.
- Save changes with validation.

Expected Prototype

- Editable form with "Update" button and real-time validation feedback.

Validation

- Success: Update 5 records in < 2 minutes.
- Zero duplicate records.
- 100% conflict detection.

Business Value

- Ensures catalog accuracy.
- Reduces manual corrections.
- Improves trust in system data.

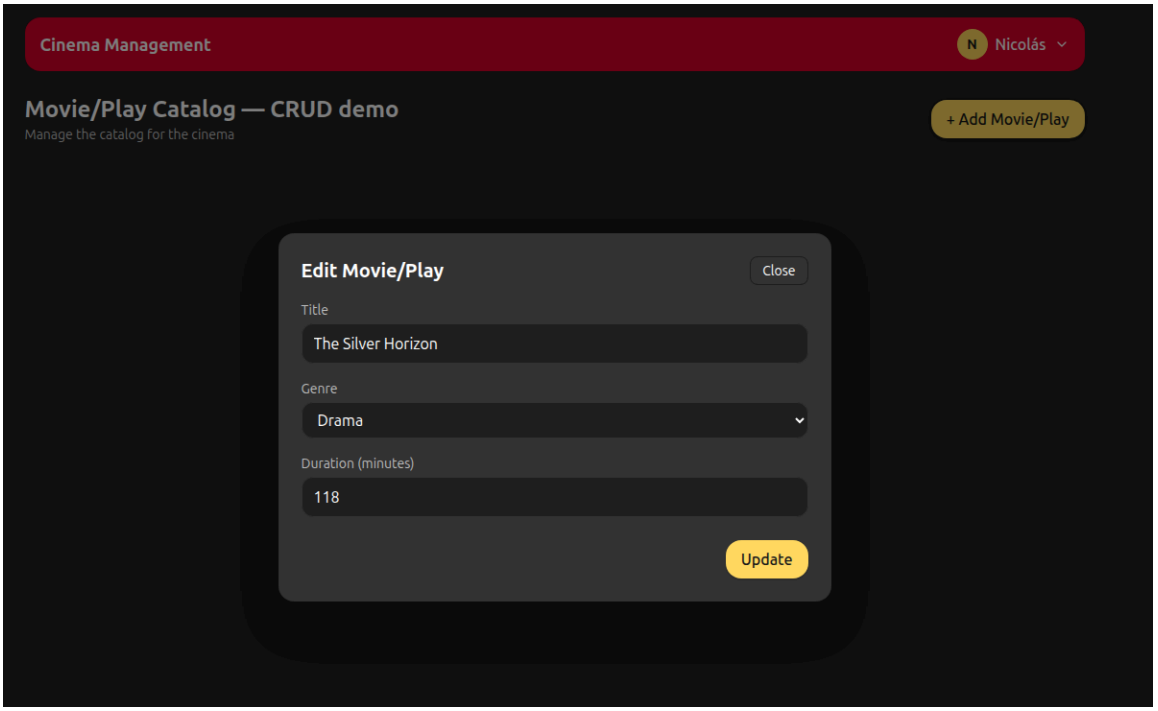

Figure 3. Mockup Function 3

Definition of Done (DoD)

- Editable form functional.
- API supports update operations (PUT/PATCH).
- Validation tested on both frontend and backend.
- Unit/integration tests > 80%.
- Deployed to staging.

US4 – Movie Deletion

User Story

As a cinema administrator
I want to delete a movie that is no longer available,
So that the database remains consistent, and people don't get confused.

Acceptance Criteria

1. Delete option available from admin catalog.
2. Confirmation required before deletion.
3. Deleted items no longer visible in the public catalog.
4. Soft delete implemented (record not erased, but hidden).
5. Cannot delete movies linked to active screenings without warning.

Agile Process – Product Discovery

Problem Discovery

- **Identified Problem:** Old, cancelled, or invalid movies remain in the catalog with no options for safe deletion.
- **Impact:** Outdated entries confuse users, clutter the catalog, generate inconsistency with screenings, and increase operational overhead for administrators.

User Need

- Ability to safely remove outdated titles while keeping history.

Proposed Solution

- Delete button with confirmation modal.
- Soft delete in database.
- Integration with screening relationship to prevent inconsistencies.

Expected Prototype

- Admin panes list with “Delete” button, after pressing it, there is a confirmation popup.

Validation

- Success: Safely delete records in < 30 seconds.
- No orphan data created.
- 100% conflict detection.

Business Value

- Keeps the catalog clean.
- Prevents user confusion.
- Ensures data consistency.

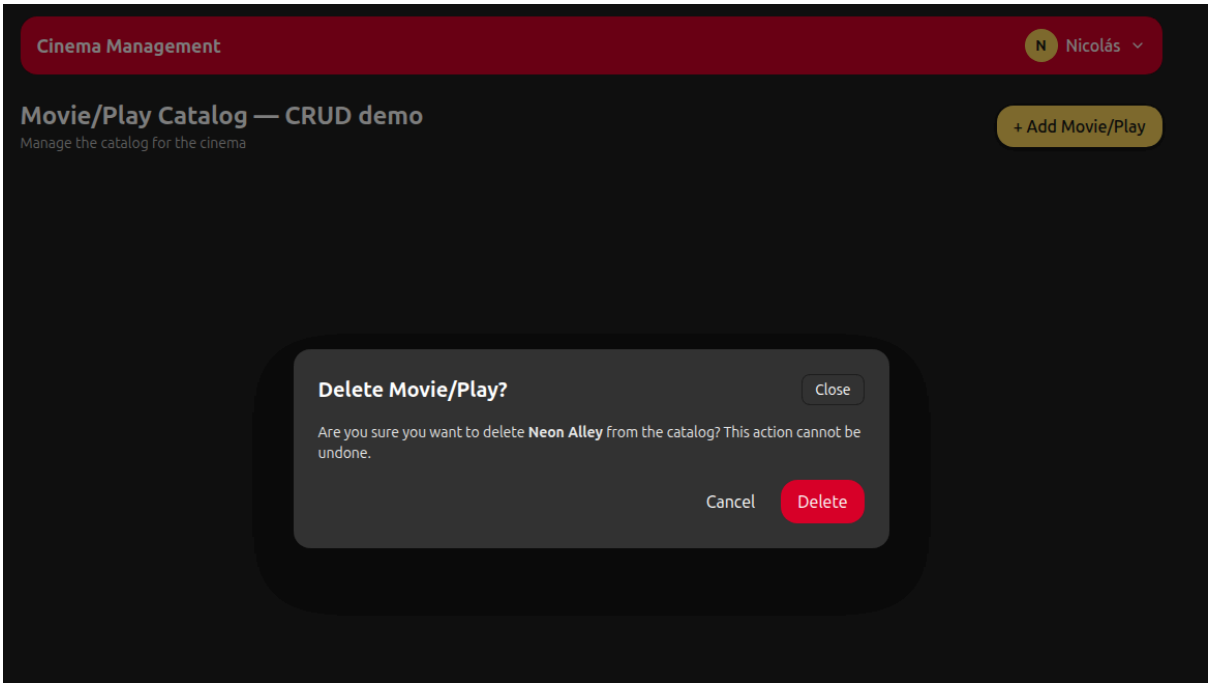

Figure 4. Mockup Function 4

Definition of Done (DoD)

- Delete functionality implemented with confirmation.
- Soft delete in DB.
- Usability test validated.
- Unit/integration tests > 80%.
- Deployed to staging.

US5 – Screening/Theater Room Registration

User Story

As a cinema administrator
I want to register screenings by specifying date, time, theater room, and movie
So that I can schedule and organize movie projections in the system

Acceptance Criteria

1. The system must allow selection of a previously registered movie.
2. The administrator must be able to choose an available theater room from the catalog.
3. Date (DD/MM/YYYY) and time (HH:MM) must be enterable.
4. The system must validate that no scheduling conflicts exist in the same theater room.
5. A visual confirmation must be displayed when the screening is successfully registered.
6. Data must be persisted in the Cinema Management microservice database.
7. The system must prevent registration of screenings on past dates.

Agile Process – Product Discovery

Problem Discovery

- **Identified problem:** The administrator currently lacks an in-system tool to plan screenings, which forces manual management and generates inconsistencies.
- **Impact:** Time wasted, errors due to duplicate scheduling, outdated information, and no automation for user notifications.

User Need

- Intuitive interface for scheduling.
- Automatic conflict validation.
- Ability to schedule multiple screenings efficiently.
- Clear visualization of room availability.

Proposed Solution

A **frontend form** connected to the Cinema Management microservice:

- Dropdown selectors for movie and theater room.
- Date/time pickers with validation.
- Real-time conflict detection.
- Preview before saving.
- Clear error messages.

Expected Prototype

- **Main panel:** Screening registration form (movie, room, date, time, optional price).
- **Secondary panel:** Daily view of screenings with timeline.
- **Interaction flow:** Fill → validate → confirm → update schedule.

Validation

- Success: Register 5 screenings in < 3 minutes.
- Error rate < 5%.
- Screenings update immediately in schedule view.
- 100% conflict detection.

Business Value

- 70% faster scheduling.
- Eliminates schedule conflicts.
- Updated, reliable schedule.
- Traceability and scalability.

figure5_mockup function 5

Definition of Done (DoD)

- Code reviewed.
- Unit/integration tests (>80% coverage).
- Frontend/backend validations.
- API documentation updated.
- Usability testing passed.
- Deployment in staging environment.

US6 – Movie–Screening Relationship

User Story

As a cinema administrator
I want to associate a movie with multiple screenings
So that the system correctly manages the schedule and users see all options available

Acceptance Criteria

1. A movie can be linked to multiple screenings (1:N).
2. Each screening must belong to one movie.
3. Deleting a movie warns about its screenings.
4. The system allows querying all screenings per movie.
5. Relationship persisted in DB.
6. Possible to modify the movie of an existing screening.
7. Visual summary of screenings per movie.

Agile Process – Product Discovery

- **Problem:** Currently, movies and screenings exist independently, causing orphan data and inconsistent schedules.
- **User Needs:** Admins need quick assignment and overview; users need all showtimes in one place.
- **Solution:**
    - Data model with FK (screening → movie).
    - REST endpoints (assign, bulk assign, query).
    - Batch creation for weekly templates.
- **Prototype:** Admin view with quick scheduling, accordion view of screenings, filters by date/room/status.

Validation

- Test: Create "Avengers 5" → 42 screenings (2 rooms × 3 times × 7 days).
- Verify schedule, deletion warning, and queries.
- Metrics: < 5 min weekly scheduling, no orphan screenings, queries < 200ms.

Business Value

- Weekly scheduling 10x faster.
- Data consistency.

- User-friendly showtimes.
- Supports analysis by title.

DoD

- Data model updated.
- Endpoints implemented and documented.
- Unit/integration tests (>85% coverage).
- Frontend quick scheduling functional.
- Load tests (100 movies × 50 screenings).
- Documentation delivered.

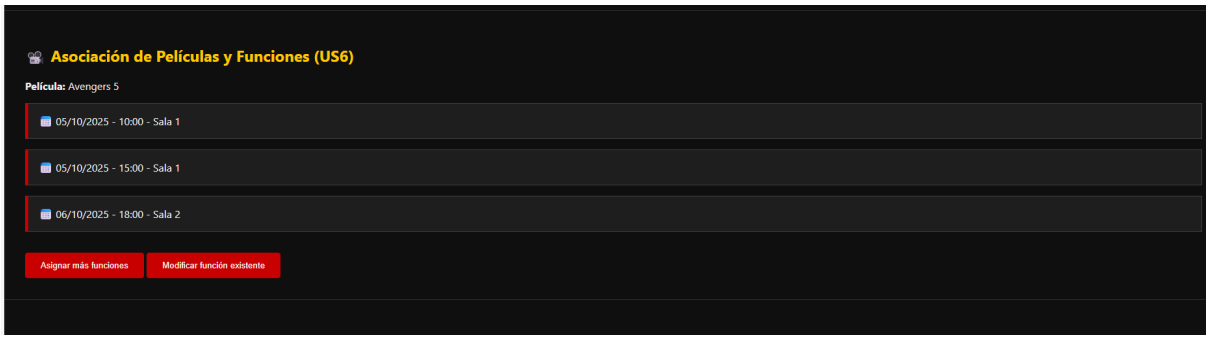

figure6_mockup function 6

US7 – Screening Query (End User)

User Story

As a cinema user
I want to see all screenings for a movie with times, rooms, and availability
So that I can choose the most convenient option

Acceptance Criteria

1. Accessible from movie detail view.
2. Screenings grouped by date.
3. Show time, room, seat availability, price.
4. No past screenings shown.
5. Low availability (<20%) highlighted.
6. Filters: date, time slot, room.
7. Loading < 1s.
8. Responsive design.

Agile Process – Product Discovery

- **Problem:** Users abandon purchases if showtimes aren't clear.
- **Needs:** Fast, filtered access to all screenings, with real-time availability.
- **Solution:** Optimized GET endpoint with caching + filters.
- **Prototype:** Movie detail view with tabs for dates, cards with showtime info, responsive design.

Validation

- User test: Buy 2 tickets for Saturday evening screening.
- Metrics: Task time < 45s, >95% success, SUS > 80/100.
- A/B test on card vs table design.

Business Value

- +25% screening views.
- +15% conversion to purchase.
- –60% support calls.

DoD

- Endpoint implemented and optimized.
- Cache with invalidation.
- Responsive frontend with filters.
- Tests and analytics metrics.
- Product Owner approval.

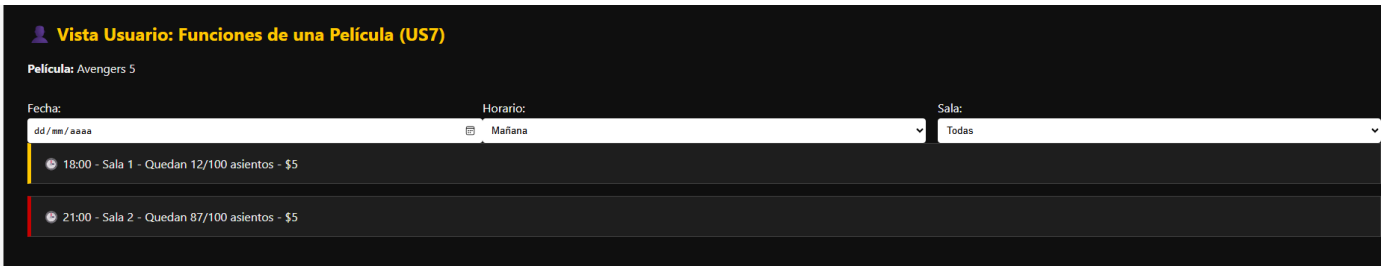

figure7_mockup function 7

US8 – Screening Deletion

User Story

As a cinema administrator
I want to delete screenings with confirmation and ticket handling
So that the schedule stays accurate and users aren't misled

Acceptance Criteria

1. Deletion from admin panel.
2. Confirmation required.
3. If tickets sold:
    - Show number of affected tickets.
    - Require cancellation reason.
    - Notify affected users.
4. Deleted screening disappears from public view.
5. Soft delete maintained in DB (historical record).
6. Audit trail (who/when).
7. Cannot delete past/ongoing screenings.

8. Batch deletion option.

Agile Process – Product Discovery

- **Problem:** Cancelled screenings remain public, creating complaints, refunds, and legal risks.
- **Needs:** Fast deletion, impact transparency, user notifications, auditability.
- **Solution:** Admin panel with deletion flow, soft delete, notification system, role-based restrictions.

Validation

- Scenario: Projector failure → delete all Room 3 screenings same day.
- Metrics: Deletion < 30s, 100% user notifications, audit log generated.

Business Value

- Preserves user trust.
- Reduces operational complaints.
- Ensures legal compliance.
- Provides complete traceability.

DoD

- Soft delete implemented.
- Notifications integrated (email/SMS).
- Role-based restrictions.
- Tests (unit, integration, usability).
- Logs/audit trail available.
- Approved by Product Owner.

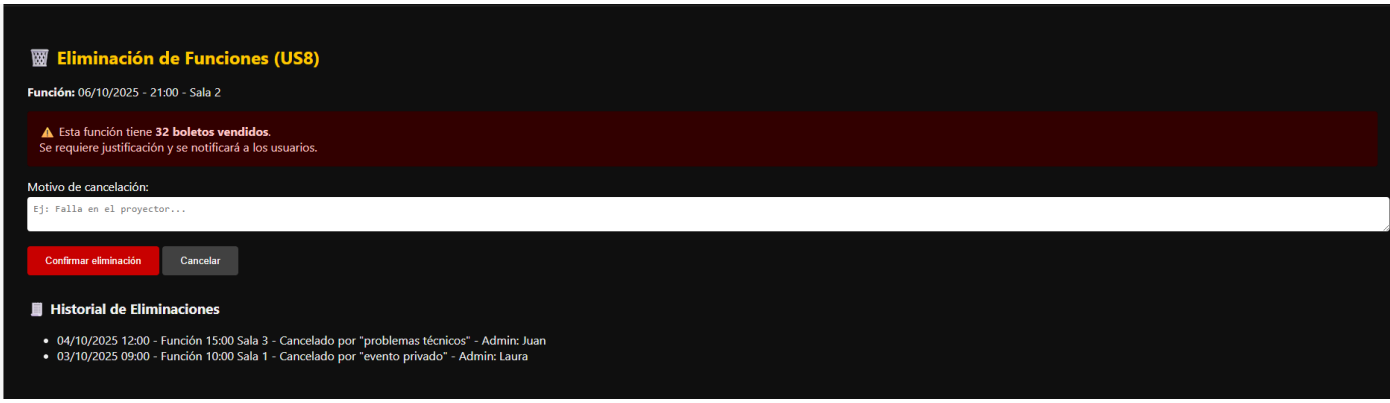

figure8_mockup function 8