Submit Button Fix for Initiator Role

Date: October 29, 2025
Status: ☑ COMPLETE
Impact: Critical UX Improvement

Problem Identified

Issue: Missing "Submit for Approval" Workflow

User Report:

> "The submit button in initiator role is not available. We only have save as draft and cancel. It's important that initiator submits req after completing the form. Saving it as a draft is an option also."

The Problem:

- Initiators could only create requisitions that went directly to `pending_hod` status
- No "Save as Draft" option for incomplete requisitions
- No clear workflow distinction between:
  - Draft (work in progress, can edit)
  - Submitted (locked, sent to HOD for approval)

Expected Workflow:

1. **Save as Draft** → Status: `draft` (initiator can edit later)
2. **Submit for Approval** → Status: `pending_hod` (locked, goes to HOD)

Solution Implemented

Backend Support (Already Existed)

The backend already had the correct endpoints:

1. Create Requisition (as draft):

```
POST /api/requisitions/simple
Creates requisition with status = 'draft'
Returns: { requisition_id, req_number }
```

2. Submit for Approval:

```
PUT /api/requisitions/:id/submit
Changes status: 'draft' → 'pending_hod'
Logs audit: 'Submitted for HOD approval'
```

Frontend Changes Made

1. Split Single Handler into Two Functions

Before: Single `handleSubmit()` function

- Always created with status `pending_hod`
- No draft option

After: Two separate functions

Function 1: `handleSaveAsDraft()`

```
const handleSaveAsDraft = async () => {
  // Validates required fields
  // Creates requisition via POST /api/requisitions/simple
  // Status remains 'draft'
  // Shows: "Requisition saved as draft successfully!"
  // Returns to dashboard
};
```

Function 2: `handleSubmitForApproval()`

```
const handleSubmitForApproval = async () => {
  // Validates required fields
  // Creates requisition via POST /api/requisitions/simple (as draft)
  // Immediately submits via PUT /api/requisitions/:id/submit
  // Status changes: 'draft' → 'pending_hod'
  // Shows: "Requisition submitted for approval successfully!"
  // Returns to dashboard
};
```

2. Updated Form Buttons

Before:

```
[Submit Requisition] [Cancel]
```

After:

```
[Submit for Approval] [Save as Draft] [Cancel]
```

Button Details:

1. **Submit for Approval** (Primary action)
    - Blue background (`bg-blue-600`)
    - Creates + immediately submits
    - Goes to HOD for approval
    - Status: `pending_hod`
2. **Save as Draft** (Secondary action)
    - Gray background (`bg-gray-600`)
    - Saves without submitting
    - Can edit later
    - Status: `draft`
3. **Cancel**
    - Gray border (`border-gray-300`)
    - Returns to dashboard
    - No changes saved

User Workflow Now

Scenario 1: Complete Requisition - Submit Immediately

1. Initiator fills out requisition form completely
2. Clicks **"Submit for Approval"**
3. System:
    - Creates requisition (status: `draft`)
    - Immediately submits (status: `pending_hod`)
    - Logs audit trail
4. Success message: "Requisition submitted for approval successfully!"
5. Goes to HOD's pending approvals
6. ☑ Initiator cannot edit (locked)

Scenario 2: Incomplete Requisition - Save as Draft

1. Initiator starts filling requisition form
2. Not ready to submit yet (missing info, need to check something)
3. Clicks **"Save as Draft"**
4. System:
    - Creates requisition (status: `draft`)
    - Does NOT submit
5. Success message: "Requisition saved as draft successfully!"
6. Remains in initiator's draft list
7. ☑ Initiator can edit anytime

Scenario 3: Edit Draft and Submit Later

1. Initiator views dashboard
2. Sees draft requisitions listed
3. Clicks "Edit" on a draft
4. Completes missing information
5. Clicks **"Submit for Approval"**
6. Status changes: `draft` → `pending_hod`
7. Goes to HOD for approval

Technical Details

API Calls Made

Save as Draft:

```
POST /api/requisitions/simple
Body: {
  description: "...",
  delivery_location: "Office",
  urgency: "standard",
  required_date: "2025-11-15",
  initiatorId: userId,
  items: [{ item_name, quantity, specifications }]
}
Response: { requisition_id: 123, req_number: "KSB-IT-JB-20251029..." }
Status in DB: 'draft'
```

Submit for Approval:

```
// Step 1: Create as draft (same as above)
POST /api/requisitions/simple

// Step 2: Submit immediately
PUT /api/requisitions/123/submit
Body: { user_id: userId }
Response: { success: true }
Status in DB: 'pending_hod'
```

Data Flow

Save as Draft:

```
User fills form
  ↓
Click "Save as Draft"
  ↓
Validate fields
  ↓
POST /api/requisitions/simple
  ↓
Backend: INSERT with status='draft'
  ↓
Response: { requisition_id }
  ↓
Reload dashboard
  ↓
Draft appears in "My Drafts" section
```

Submit for Approval:

```
User fills form
  ↓
Click "Submit for Approval"
  ↓
Validate fields
  ↓
POST /api/requisitions/simple (creates as draft)
  ↓
Response: { requisition_id }
  ↓
PUT /api/requisitions/{id}/submit
  ↓
Backend: UPDATE status='pending_hod'
Backend: INSERT audit_log 'submitted'
  ↓
Reload dashboard
  ↓
Requisition appears in HOD's pending list
```

Files Modified

frontend/app.js:

- Lines ~1268-1340: Split `handleSubmit()` into two functions:
  - `handleSaveAsDraft()`
  - `handleSubmitForApproval()`
- Lines ~1435-1450: Updated button section with 3 buttons
  - Submit for Approval (primary - blue)
  - Save as Draft (secondary - gray)
  - Cancel (tertiary - border)

No backend changes needed - endpoints already existed!

Testing Checklist

☑ Save as Draft

- [ ] Fill form partially
- [ ] Click "Save as Draft"
- [ ] Check dashboard - draft appears
- [ ] Check database - status = 'draft'
- [ ] Edit draft - form pre-populated
- [ ] Verify can edit and save again

☑ Submit for Approval

- [ ] Fill form completely
- [ ] Click "Submit for Approval"
- [ ] Check dashboard - requisition gone from drafts
- [ ] Check HOD dashboard - requisition appears in pending
- [ ] Check database - status = 'pending_hod'
- [ ] Verify initiator cannot edit (locked)

☑ Validation

- [ ] Try to submit with empty description - should show error
- [ ] Try to submit with empty quantity - should show error
- [ ] Try to submit with empty date - should show error
- [ ] Try to submit with empty justification - should show error

☑ UI/UX

- [ ] Three buttons visible and clearly labeled
- [ ] "Submit for Approval" is primary (blue, prominent)
- [ ] "Save as Draft" is secondary (gray)
- [ ] Buttons disabled during loading
- [ ] Loading text appears ("Submitting..." / "Saving...")
- [ ] Success messages display correctly
- [ ] Returns to dashboard after save/submit

Status Lifecycle

States:

- `draft` - Saved but not submitted, editable by initiator
- `pending_hod` - Submitted, locked, awaiting HOD approval
- `pending_md` - HOD approved, awaiting MD approval
- `approved` - Fully approved
- `rejected` - Rejected at any stage

Benefits

For Initiators

☑ Can save incomplete work without submitting
☑ Can come back and edit drafts later
☑ Clear distinction between draft and submitted
☑ Less pressure to complete everything in one session
☑ Can verify information before final submission

For HODs

☑ Only receive complete, submitted requisitions
☑ No confusion about draft vs. submitted status
☑ Clean pending approvals list

For System

☑ Clear audit trail (created vs. submitted)
☑ Proper status workflow
☑ Better data integrity
☑ Follows industry best practices

Related Endpoints

Endpoints Used:

- `POST /api/requisitions/simple` - Create requisition (default: draft)
- `PUT /api/requisitions/:id/submit` - Submit for approval (draft → pending_hod)
- `GET /api/requisitions` - Get all requisitions (includes drafts)
- `PUT /api/requisitions/:id` - Update requisition (for editing drafts)

Future Enhancements:

- Add "Continue Editing" button on draft list
- Add draft count badge in sidebar
- Add auto-save for drafts
- Add "Delete Draft" option
- Add draft age indicator (created 2 days ago)

Summary

Problem: Only one "Submit" action, no draft workflow
Solution: Split into "Save as Draft" + "Submit for Approval"
Impact: Better UX, clear workflow, less pressure on initiators

Files Changed: 1 (frontend/app.js)
Lines Added: ~70
Backend Changes: None (already supported)
User Impact: Significant improvement ☑

Fix Completed: October 29, 2025, 1:30 PM
Status: ☑ Ready for Testing
Breaking Changes: None
Migration Required: None