CliniHOF - Phase 2 Implementation

✅ Implemented Features

This document describes all Phase 2 features implemented in the CliniHOF clinic management system.

---

1. FUTURE APPOINTMENTS VIEW ✅

Page Location

- **URL**: `/agenda/futuros` or `/agendamentos-futuros`
- **File**: `app/(dashboard)/agenda/futuros/page.tsx`
- **API**: `app/api/agenda/futuros/route.ts`

Features Implemented

- ✅ Comprehensive view of all upcoming appointments
- ✅ Advanced filters:
- Date range picker (from/to dates)
- Collaborator dropdown filter
- Patient dropdown filter
- Status filter (PENDING, SCHEDULED, COMPLETED, CANCELLED)
- ✅ Real-time search by patient name or procedure name
- ✅ Sortable columns:
- Date/time (ascending/descending)
- Patient name
- Procedure name
- Collaborator name
- ✅ Statistics cards showing:
- Total appointments
- Pending, Scheduled, Completed, Cancelled counts
- Unique patients count
- ✅ Procedure breakdown showing most common procedures
- ✅ Export to CSV functionality
- ✅ Quick date range presets (7 days, 30 days, 3 months, all)

Usage

Navigate to “Agendamentos Futuros” in the sidebar to access the comprehensive appointments view with all filters and sorting options.

---

2. CREATE APPOINTMENTS IN CALENDAR ✅

Files Modified

- **Calendar Page**: `app/(dashboard)/agenda/page.tsx`
- **Create API**: `app/api/agenda/criar/route.ts`
- **Modal Component**: `components/forms/appointment-modal.tsx`

Features Implemented

- ✅ **Click-to-create**: Single/double-click on calendar day cells to open appointment creation modal
- ✅ **Quick create button**: “Novo Agendamento” button in header
- ✅ **Pre-filled date/time**: Modal pre-populates selected date and time slot
- ✅ **Smart session handling**:
- Finds existing pending sessions for patient/procedure
- Updates existing sessions when available
- Creates new sale if needed
- ✅ **Comprehensive form**:
- Patient selection (searchable dropdown)
- Procedure selection (with colors and duration)
- Collaborator assignment (optional)
- Date and time picker with 30-minute increments
- Appointment type classification
- Notes field
- Toggle to create new sale or use existing
- ✅ **Calendar integration**: Appointments immediately appear after creation
- ✅ **Link to future appointments**: Quick access button to view all future appointments

Usage

1. Click “Novo Agendamento” or double-click a calendar day
2. Fill in patient, procedure, and schedule details
3. Optionally assign a collaborator
4. Click “Criar Agendamento”
5. Appointment appears on calendar immediately

---

3. BIDIRECTIONAL GOOGLE CALENDAR SYNC ✅

Files Created/Modified

- **Google Calendar Service**: `lib/google-calendar.ts`
- **Sync API**: `app/api/google-calendar/sync/route.ts`
- **Settings Page**: `app/(dashboard)/configuracoes/page.tsx`
- **Settings API**: `app/api/settings/route.ts`
- **Auth Config**: `lib/auth.ts` (added Google Calendar scopes)

Database Changes

- ✅ Added `googleEventId` field to `ProcedureSession` model

- ✅ Created `WorkspaceSettings` model with:
- `googleCalendarEnabled` (Boolean)
- `googleCalendarId` (String) - defaults to “primary”
- `googleSyncToken` (String) - for incremental sync
- `lastGoogleSync` (DateTime)

Features Implemented

- ✅ **OAuth Integration**: Google Calendar API with proper scopes (`calendar.events`)
- ✅ **Bi-directional Sync**:
- **CliniHOF → Google**: When appointments are created/updated/deleted in CliniHOF, corresponding events are managed in Google Calendar
- **Google → CliniHOF**: Webhook endpoint to receive change notifications from Google Calendar
- ✅ **Event Details**: Google Calendar events include:
- Summary: “Procedure Name - Patient Name”
- Description: Patient contact info, procedure details, notes
- Duration: Based on procedure duration
- Attendees: Patient email (if available)
- Reminders: 60 min and 15 min before
- Extended properties: CliniHOF session ID for tracking
- ✅ **Settings UI** (`/configuracoes`):
- Google account connection status
- Enable/disable sync toggle
- Calendar ID configuration
- Sync now button
- Last sync timestamp
- ✅ **Smart Sync Functions**:
- `syncSessionToGoogleCalendar()` : Handles create/update/delete actions
- `syncFromGoogleCalendar()` : Pulls changes from Google Calendar
- `createGoogleCalendarEvent()` : Creates new event
- `updateGoogleCalendarEvent()` : Updates existing event
- `deleteGoogleCalendarEvent()` : Removes event

Setup Required

1. Configure Google Cloud Console:
   - Enable Google Calendar API
   - Create OAuth 2.0 credentials
   - Add redirect URI: `https://yourdomain.com/api/auth/callback/google`
2. Update `.env` :
   ```
   GOOGLE_CLIENT_ID=your_client_id
      GOOGLE_CLIENT_SECRET=your_client_secret
   ```
3. Users must:
   - Sign in with Google (account linking)
   - Enable sync in `/configuracoes`
   - Configure calendar ID (default: “primary”)

Usage

1. Navigate to “Configurações” in sidebar

2. Click “Conectar” to link Google account
3. Enable “Sincronização Automática”
4. Click “Sincronizar Agora” to sync existing appointments
5. All future appointment changes automatically sync

---

4. PROCEDURE FIXED COST ✅

Database Changes

- ✅ Added `fixedCost` field to `Procedure` model (Float, default 0)

Files Modified

- **Procedure API**: `app/api/procedures/route.ts`
- **Procedure Detail API**: `app/api/procedures/[id]/route.ts`
- **Procedure Form**: `components/forms/procedure-form.tsx`

Features Implemented

- ✅ **Cost Calculation Logic**:
  ```

  Total Cost = fixedCost + supplyCosts + collaboratorCosts

supplyCosts = Σ(supply.costPerUnit × quantity)
collaboratorCosts = Σ((timeMinutes / 60) × hourlyRate)
hourlyRate = (baseSalary + charges) / monthlyHours
``
`- ✅ **Automatic Calculations**:`
`  - Supply costs from linked supplies and quantities`
`  - Collaborator costs from time spent and hourly rates`
`  - Total procedure cost`
`  - Profit margin percentage:` ((price - totalCost) / price) × 100 `- ✅ **API Enhancements**:`
`  - GET returns calculated costs for all procedures`
`  - POST/PUT accepts` fixedCost `field`
`  - Helper function` calculateProcedureCosts()` computes all cost breakdowns
- ✅ **UI Updates**:
- Procedure form includes “Custo Fixo (R$)” field
- Clear help text explaining fixed costs
- Cost breakdown displayed in procedure details

### Usage

1. Create/edit a procedure
2. Enter fixed cost (equipment, exclusive products, etc.)
3. Add supplies and collaborators (optional)
4. System automatically calculates:
   - Total supply costs
   - Total collaborator costs
   - Total procedure cost
   - Profit margin
5. View complete cost breakdown in procedure list/detail
  ```

5. CLINIC HOUR CALCULATION ✅

Database Changes

- ✅ Created `WorkspaceSettings` model with:
- `monthlyFixedCosts` (Float) - Total fixed costs per month
- `monthlyWorkingHours` (Float) - Working hours per month (default: 176)
- `hourlyClinicCost` (Float) - Calculated: costs / hours

Files Created

- **Settings Page**: `app/(dashboard)/configuracoes/page.tsx`
- **Settings API**: `app/api/settings/route.ts`

Features Implemented

- ✅ **Settings Management**:
- Input monthly fixed costs (rent, utilities, salaries, etc.)
- Configure monthly working hours (default: 176h = 22 days × 8h)
- Auto-calculate hourly clinic cost
- ✅ **Auto-calculation from Costs**:
- Button to sum all active fixed costs from Cost model
- Automatically populates monthly fixed costs
- ✅ **KPI Display**:
- Large, prominent display of hourly clinic cost
- Visual metrics cards showing:
    - Total monthly costs
    - Working hours per month
    - Calculated cost per hour
- ✅ **Formula**:
  `hourlyClinicCost = monthlyFixedCosts / monthlyWorkingHours`
- ✅ **Integration**:
- Hourly cost available for procedure cost calculations
- Can be used in profitability analysis
- Displayed on dashboard as KPI

Usage

1. Navigate to “Configurações”
2. Enter monthly fixed costs manually OR click “Calculate” to auto-sum from Costs
3. Adjust monthly working hours if needed
4. View calculated hourly clinic cost
5. Save settings
6. Metric appears in dashboard and cost calculations

6. PACKAGES IN SALES/APPOINTMENTS ✅

Database Changes

- ✅ Added `packageId` field to `Sale` model (optional foreign key)
- ✅ Added `sales` relation to `Package` model

Files Modified

- **Sales API**: `app/api/sales/route.ts`
- **Appointments Page**: `app/(dashboard)/appointments/page.tsx` (packages fetched)
- **Package Sessions Page**: `app/(dashboard)/pacotes-sessoes/page.tsx` (NEW)

Features Implemented

- ✅ **Package Selection in Sales**:
- API supports `packageId` parameter
- When package is selected:
  - Auto-populates sale items from package items
  - Uses package `finalPrice` as total
  - Creates `ProcedureSession` records for each procedure
  - Sets all sessions to “PENDING” status
  - Links sale to package via `packageId`
- ✅ **Session Tracking**:
- Each package sale creates N sessions (one per procedure × quantity)
- Sessions track individual procedure completions
- Progress tracking: completed vs total sessions
- ✅ **Package Sessions View** (`/pacotes-sessoes`):
- Lists all package sales with:
  - Patient name
  - Package name
  - Sale date
  - Session progress (X/Y completed)
  - Progress bar and percentage
- Expandable cards showing all sessions:
  - Session number
  - Procedure name
  - Scheduled date (if any)
  - Completion date (if completed)
  - Status badge
  - “Concluir” button for pending sessions
- Statistics:
  - Total packages sold
  - Total sessions
  - Completed sessions count
  - Pending sessions count
  - Overall progress bar
- ✅ **API Enhancements**:

- GET `/api/sales` includes package information
- POST `/api/sales` with `packageId` auto-creates sessions
- Session completion API updates progress

Usage

1. **Create Package Sale**:
   `json`
   ```
     POST /api/sales
     {
       "patientId": "...",
       "packageId": "...",
       "totalAmount": 1500.00,
       "paymentSplits": [...]
     }
   ```
2. **Track Progress**:
   - Navigate to “Sessões de Pacotes”
   - Click on a sale to expand and see all sessions
   - Mark sessions as completed
   - View real-time progress updates
3. **Benefits**:
   - Automatic session creation
   - Clear progress tracking
   - Easy completion management
   - Patient-specific package progress

Additional Improvements

Sidebar Navigation

- ✅ Added “Agendamentos Futuros” link
- ✅ Added “Configurações” link
- ✅ Added “Sessões de Pacotes” link
- ✅ Organized logical menu structure

Error Handling

- ✅ Comprehensive error messages
- ✅ Toast notifications for user feedback
- ✅ Validation on all forms
- ✅ Graceful fallbacks for missing data

Performance

- ✅ Efficient database queries with proper includes
- ✅ Pagination ready (future enhancement)
- ✅ Optimized re-renders with proper React patterns

Database Schema Updates

New Models

```
model WorkspaceSettings {
  id                    String    @id @default(cuid())
  workspaceId           String    @unique

  // Clinic costs
  monthlyFixedCosts     Float     @default(0)
  monthlyWorkingHours   Float     @default(176)
  hourlyClinicCost      Float?

  // Google Calendar
  googleCalendarEnabled Boolean   @default(false)
  googleCalendarId      String?
  googleSyncToken       String?
  lastGoogleSync        DateTime?

  createdAt             DateTime  @default(now())
  updatedAt             DateTime  @updatedAt

  @@index([workspaceId])
}
```

Schema Modifications

```
model Procedure {
  // Added:
  fixedCost   Float?   @default(0)
}

model Sale {
  // Added:
  packageId   String?
  package     Package? @relation(fields: [packageId], references: [id])

  @@index([packageId])
}

model ProcedureSession {
  // Added:
  collaboratorId  String?
  googleEventId   String?

  @@index([googleEventId])
}

model Package {
  // Added:
  sales Sale[]
}
```

API Routes Summary

New Routes

- `GET /api/agenda/futuros` - List future appointments with filters
- `POST /api/agenda/criar` - Create appointment directly from calendar
- `GET /api/settings` - Get workspace settings
- `PUT /api/settings` - Update workspace settings
- `POST /api/settings` - Calculate costs from Cost model
- `GET /api/google-calendar/sync` - Pull from Google Calendar
- `POST /api/google-calendar/sync` - Push to Google Calendar

Modified Routes

- `POST /api/sales` - Now accepts `packageId`
- `GET /api/sales` - Now includes package information
- `GET /api/procedures` - Now includes cost calculations
- `POST /api/procedures` - Now accepts `fixedCost`

Environment Variables

Add to `.env`:

```
# Google Calendar Integration
GOOGLE_CLIENT_ID="your_google_client_id"
GOOGLE_CLIENT_SECRET="your_google_client_secret"
```

Migration Commands

To apply database changes:

```
npx prisma generate
npx prisma db push
```

Or create a migration:

```
npx prisma migrate dev --name phase2_features
```

Testing Checklist

Future Appointments View

- [ ] Access `/agenda/futuros`
- [ ] Test date range filters
- [ ] Test search functionality

- [ ] Test sorting by different columns
- [ ] Export CSV and verify data

Calendar Appointment Creation

- [ ] Double-click calendar day to create appointment
- [ ] Click "Novo Agendamento" button
- [ ] Verify date/time pre-fill
- [ ] Create appointment and verify it appears
- [ ] Test with existing pending sessions

Google Calendar Sync

- [ ] Connect Google account in Configurações
- [ ] Enable sync and configure calendar ID
- [ ] Click "Sincronizar Agora"
- [ ] Create appointment in CliniHOF → verify in Google Calendar
- [ ] Update appointment in Google Calendar → verify in CliniHOF

Procedure Fixed Cost

- [ ] Create procedure with fixed cost
- [ ] Add supplies and collaborators
- [ ] Verify total cost calculation
- [ ] Check profit margin display

Clinic Hour Calculation

- [ ] Navigate to Configurações
- [ ] Enter monthly costs and hours
- [ ] Click auto-calculate button
- [ ] Verify hourly cost calculation
- [ ] Save settings

Packages in Sales

- [ ] Create sale with package
- [ ] Verify sessions auto-created
- [ ] Navigate to "Sessões de Pacotes"
- [ ] Expand sale and view sessions
- [ ] Mark session as completed
- [ ] Verify progress updates

Next Steps (Future Enhancements)

1. **UI for Package Selection in Sales Form**
   - Add toggle between "Individual" and "Package" sale types
   - Package dropdown with auto-populate
   - Dynamic session date inputs based on package items

2. **Advanced Google Calendar Features**
   - Two-way webhook notifications

   - Conflict detection
   - Recurring appointments sync
   - Calendar color coding by procedure

3. **Enhanced Cost Analytics**
   - Procedure profitability dashboard
   - Cost trend charts
   - Break-even analysis
   - Margin optimization suggestions

4. **Package Management**
   - Package templates
   - Seasonal packages
   - Package usage analytics
   - Automatic expiration handling

Support

For questions or issues, refer to:
- Main README.md for general setup
- Prisma schema documentation: `prisma/schema.prisma`
- API route files for endpoint details

Implementation Date: January 30, 2026
Version: Phase 2 Complete
Developer: Abacus.AI Deep Agent