Project Portal Documentation

Overview

Project Portal is a centralized web application that displays real-time project information from your organization's Google Sheets data. It provides engineers, stakeholders, and developers with instant visibility into project status, budget, progress, and other critical metrics through an intuitive web interface embedded in Google Sites.

Key Benefits:

- Real-time project data synchronization from Google Sheets
- Quick assessment of project health and status
- Budget tracking and progress monitoring
- Accessible from any device through Google Sites

Prerequisites

Before setting up or using Project Portal, ensure you have:

For Setup (Administrators):

- Google Workspace account with access to Google Sheets and Google Sites
- Supabase account (free tier available at supabase.com)
- Basic knowledge of Google Apps Script
- Understanding of HTML, CSS, and JavaScript
- Editor permissions for the source Google Sheet

For Users (Engineers/Stakeholders):

- Access credentials to your organization's Google Sites
- Modern web browser (Chrome, Firefox, Safari, or Edge)
- Network connection to access the portal

Architecture Overview

Project Portal uses a three-tier architecture:

1. **Data Source:** Google Sheets containing project information
2. **API Layer:** Supabase database with automatic synchronization
3. **Presentation Layer:** Web interface (HTML/CSS/JS) embedded in Google Sites

[Google Sheets] → [Apps Script Sync] → [Supabase Database] → [Web Interface] → [Google Sites]

Setup Instructions

Step 1: Create Supabase Project and Table

1. Log in to your Supabase account at supabase.com
2. Click "New Project" and configure:
   - Project name (e.g., "ProjectPortal")
   - Database password (save securely)
   - Region (select closest to your users)
3. Wait for project provisioning (2-3 minutes)
4. Navigate to the "Table Editor" in the left sidebar
5. Create a new table matching your Google Sheet structure:
   - Click "Create a new table"
   - Name it appropriately (e.g., `projects`)
   - Add columns matching your Sheet headers (sl, project_name, status, budget, progress, etc.)
   - Enable "Enable Row Level Security" for data protection

Important: Note your table name—you'll need it for configuration.

Step 2: Configure Google Sheets Synchronization

1. Open your source Google Sheet containing project data
2. Access Apps Script:
   - Click "Extensions" → "Apps Script"
3. Delete any existing code in the editor
4. Paste the synchronization script (request from your admin or see Additional Resources)

Configure the script variables:
const SUPABASE_URL = 'your-project-url.supabase.co';
const SUPABASE_KEY = 'your-anon-public-key';
const TABLE_NAME = 'projects';

5. Save the script (Ctrl+S or Cmd+S)
6. Set up automatic triggers:
   - Click the clock icon (Triggers)
   - Add trigger: `onEdit` for real-time sync
   - Add trigger: `syncToSupabase` time-driven (hourly) for reliability

Step 3: Retrieve Supabase API Credentials

1. In Supabase, navigate to "Settings" → "API"
2. Copy the following values:
   - **Project URL** (e.g., `https://xxxxx.supabase.co`)
   - **anon public API key** (starts with `eyJ...`)
3. Store these securely—you'll need them for the web interface

Step 4: Build the Web Interface

1. Create a new HTML file for your portal

Add the fetch configuration:

```
const SUPABASE_CONFIG = {
  URL: 'https://your-project.supabase.co',
  API_KEY: 'your-anon-public-key'
};
const TABLE_NAME = 'projects';
```

2. Implement the data fetching function:

```
      async function fetchProjects() {

  try {
    const response = await fetch(
      `${SUPABASE_CONFIG.URL}/rest/v1/${TABLE_NAME}?select=*&order=sl.asc`,
      {
        headers: {
          'apikey': SUPABASE_CONFIG.API_KEY,
          'Authorization': `Bearer ${SUPABASE_CONFIG.API_KEY}`,
          'Content-Type': 'application/json'
        }
      }
    );

    if (!response.ok) {
      throw new Error(`HTTP error! status: ${response.status}`);
    }

    const data = await response.json();
    return data;
  } catch (error) {
    console.error('Error fetching projects:', error);
    return [];
  }
}
```

3. Build your HTML structure to display project cards or tables

4. Style with CSS to match your organization's branding

5. Add JavaScript for interactivity (filtering, searching, sorting)

Step 5: Embed in Google Sites

1. Navigate to your Google Site
2. Edit the page where you want the portal
3. Click "Insert" → "Embed"
4. Choose "Embed code"
5. Paste your complete HTML file (including `<style>` and `<script>` tags)
6. Adjust the embed size as needed
7. Click "Insert" and publish your site

Using Project Portal

Viewing Project Information

Dashboard Overview: The portal displays all projects in a structured format showing:

- Project serial number (SL)
- Project name and description
- Current status (Active, On Hold, Completed, etc.)
- Budget allocation and spent amount
- Progress percentage
- Key milestones and deadlines

Navigation:

- Projects are automatically sorted by serial number
- Use search functionality to find specific projects
- Filter by status, department, or other criteria
- Click on project cards for detailed information

Understanding Project Status

Common status indicators:

- **Active:** Project is currently in progress
- **On Track:** Meeting milestones and budget
- **At Risk:** Behind schedule or over budget
- **On Hold:** Temporarily paused
- **Completed:** Successfully delivered
- **Cancelled:** Discontinued

Interpreting Progress Metrics

- **Progress Bar:** Visual representation of completion percentage
- **Budget Status:** Shows allocated vs. spent amounts
- **Timeline:** Displays start date, current phase, and deadline

---

Best Practices

For Data Managers

Maintain Data Quality:

- Update the Google Sheet regularly with accurate information
- Use consistent formatting for dates (YYYY-MM-DD)
- Standardize status values (create dropdown lists in Sheets)
- Include all required fields for each project

Optimize Sync Performance:

- Limit sheet size to active projects (archive completed ones)
- Avoid excessive formatting in source sheets
- Test synchronization after major sheet changes

For Portal Users

Daily Workflow:

- Check portal at the start of your day for updates
- Bookmark the portal page for quick access
- Use filters to focus on your relevant projects
- Report data discrepancies to your project manager immediately

Collaboration Tips:

- Share direct links to specific project views with team members
- Reference project serial numbers in communications
- Use portal data in status meetings for consistency

For Administrators

Security Considerations:

- Never commit API keys to version control
- Use environment variables for sensitive credentials
- Regularly rotate Supabase API keys (quarterly recommended)
- Enable Row Level Security policies in Supabase
- Restrict Google Sheet edit access to authorized personnel

Performance Optimization:

- Implement caching for frequently accessed data
- Use pagination for large project lists (100+ projects)
- Compress images and assets
- Monitor Supabase usage to stay within plan limits

Troubleshooting

Projects Not Displaying

Symptoms: Portal shows empty or loading indefinitely

Solutions:

1. Check browser console for error messages (F12)
2. Verify Supabase API credentials are correct
3. Confirm table name matches configuration
4. Check Supabase project is active (not paused)
5. Test API endpoint directly in browser:
   `https://your-project.supabase.co/rest/v1/projects?select=*`

Data Not Syncing from Google Sheets

Symptoms: Changes in Sheet don't appear in portal

Solutions:

1. Check Apps Script execution logs:
   - Open Apps Script editor
   - View → "Executions"
   - Look for errors in recent runs
2. Verify trigger is active and not disabled
3. Manually run `syncToSupabase` function to test
4. Check Sheet column names match Supabase table structure exactly
5. Ensure Apps Script has necessary permissions

API Key Errors (401 Unauthorized)

Symptoms: "Invalid API key" or authorization failures

Solutions:

1. Verify you're using the **anon public** key, not service_role key
2. Check for extra spaces or characters in API key
3. Confirm Supabase project is not paused
4. Regenerate API key in Supabase if compromised

Slow Loading Performance

Symptoms: Portal takes 5+ seconds to load

Solutions:

1. Check number of records—consider pagination for 500+ projects
2. Reduce query complexity (limit selected columns)

Implement client-side caching:
 const CACHE_DURATION = 5 * 60 * 1000; // 5 minuteslet cachedData = null;let cacheTime = null;

3.
4. Optimize images and reduce file sizes
5. Use CDN for static assets

Embed Not Working in Google Sites

Symptoms: Blank space or "Unable to load" message

Solutions:

1. Verify HTML is properly formatted (closed tags)
2. Check for JavaScript errors in console
3. Ensure external resources (fonts, libraries) use HTTPS
4. Remove any blocked content (iframes, external scripts)
5. Try embedding in a test page first
6. Check Google Sites doesn't block your Supabase domain

FAQ

Q: How often does data sync from Google Sheets?
A: By default, data syncs on every edit and hourly via time-based trigger. You can adjust the hourly trigger frequency in Apps Script.

Q: Can I customize the portal appearance?
A: Yes. Modify the HTML, CSS, and JavaScript files to match your branding. Ensure changes don't break data fetching logic.

Q: Who can view the Project Portal?
A: Anyone with access to the Google Site where it's embedded. Control access through Google Sites sharing settings.

Q: How do I add new project fields?
A: Add columns to Google Sheet → Add corresponding columns to Supabase table → Update web interface to display new fields.

Q: Is my project data secure?
A: Data security depends on your configuration. Use Row Level Security in Supabase, keep API keys private, and restrict Sheet access appropriately.

Q: Can I export portal data?
A: Portal displays Google Sheets data—export directly from the source Sheet. Alternatively, implement export functionality in the web interface.

Q: What happens if Supabase is down?
A: Portal will show an error or fail to load. Implement fallback messaging and monitor Supabase status at status.supabase.com.

Q: Can multiple people edit the Google Sheet simultaneously?
A: Yes. Google Sheets supports collaborative editing. Sync triggers will fire for each change, though Supabase may rate-limit rapid updates.

Additional Resources

Documentation Links

- Supabase JavaScript Client Documentation
- Google Apps Script Reference
- Google Sites Embed Guide

Code Repositories

- Request complete Apps Script code from your system administrator
- Access HTML/CSS/JS templates from your organization's development team

Support Channels

- **Technical Issues:** Contact your IT department or project administrator
- **Data Questions:** Reach out to project managers or data owners
- **Feature Requests:** Submit feedback through your organization's channels

Training Materials

- Schedule a walkthrough session with your team lead
- Review recorded demo sessions (if available)
- Consult organization-specific style guides

Next Steps & Recommendations

For Immediate Implementation

Clarifications Needed:

1. **Data Structure:** What specific columns exist in your Google Sheet? (project name, status, budget, progress, owner, dates, etc.)
2. **Status Values:** What are the exact status options used in your organization? (helps create filtering logic)
3. **User Roles:** Do different users need different views or permissions?
4. **Visual Requirements:** Are there specific branding guidelines, color schemes, or layouts required?
5. **Integration Points:** Does this need to integrate with other tools (Jira, Slack, etc.)?

Missing Documentation Sections:

- **Sample Apps Script Code:** Complete synchronization script with error handling
- **Complete HTML/CSS/JS Template:** Full working example with best practices
- **Security Configuration Guide:** Detailed Supabase RLS policies
- **Backup and Recovery:** Procedures for data loss scenarios
- **Version Control:** Guidelines for managing code updates

For Future Iterations

Enhancement Opportunities:

- Add real-time notifications for project status changes
- Add data visualization (charts, graphs for budgets and timelines)
- Enable commenting and collaboration features
- Integrate with calendar for milestone tracking
- Implement advanced search with saved filters

Feedback Collection:

- Survey users after 2-4 weeks of usage
- Track most-used features via analytics
- Identify pain points and feature requests
- Monitor performance metrics and optimize

Maintenance Plan:

- Schedule quarterly reviews of documentation accuracy
- Update screenshots and examples as UI evolves
- Archive outdated versions with changelog
- Train new team members on portal usage

Document Metadata

Version: 1.0
Last Updated: January 4, 2026
Owner: Sheikh Sayed
Reviewers Needed: Product Manager, Lead Developer, End Users
Next Review Date: 3 months from publication
