Development Projects

This chapter contains hands-on development tasks designed to help you apply what you've learned and contribute to our church's technical projects. Each page in this chapter outlines a self-contained project, complete with step-by-step instructions, goals, and deliverables. Start with the first project and work through them at your own pace. As new needs arise, we'll continue adding more projects here.

BookStack + Google Apps Script Integration Demo

Overview

This project will help you get familiar with our technical setup by integrating a Google Apps Script web app into our BookStack knowledge base. You will be working with Docker, Google Apps Script, and BookStack.

Objectives

- Set up and run the BookStack app using Docker containers.
- Create and deploy a "Hello World" Google Apps Script web app.
- Embed the web app as an iframe in a BookStack page.
- Record a short demo video to demonstrate the setup.

Prerequisites

Before starting, please review the following documentation to ensure a smooth setup:

- BookStack Official Documentation
- Docker Documentation

> **Note:**
> - The following steps are starting points; additional configuration or troubleshooting may be required.
> - Embedding an iframe might necessitate adjusting BookStack's configuration settings.
> - Deploying the Google Apps Script web app could involve modifying access permissions or deployment settings.
> - Utilize tools like ChatGPT, Grok Deep Research, or other resources to assist with any challenges you encounter during this project.

Tasks

1. Set Up BookStack with Docker

- Use the official BookStack Docker guide or our internal repository if available.
- Ensure the BookStack app is accessible locally, e.g., `http://localhost:8080`.
- Log in using the default or configured credentials.

Expected Outcome: BookStack app is running locally in Docker.

2. Create a Google Apps Script Web App

1. Go to script.google.com.
2. Create a new project and paste the following code:

```
function doGet() {
  return HtmlService.createHtmlOutput("<h1>Hello World</h1>");
}
```

3. Deploy the script as a web app:
   - Click **Deploy > New deployment**.
   - Click **Select type** and choose **Web app**.
   - Enter a description, e.g., "Hello World Deployment".
   - Under **Execute as**, select **Me**.
   - Under **Who has access**, select **Anyone**.
   - Click **Deploy**.
   - Authorize the script if prompted.
   - Copy the deployment URL.

Expected Outcome: A public URL that displays "Hello World".

---

3. Embed the Web App in BookStack

- Log in to your local BookStack instance.
- Create a new Page or edit an existing one.
- Use the following HTML to embed the web app:

```
<iframe src="YOUR_WEB_APP_URL_HERE" width="100%" height="500" style="border: none;"></iframe>
```

> Embedding external content via iframes may require additional configuration in BookStack to allow such content. Refer to the BookStack documentation for guidance on adjusting settings to permit iframe embedding.

Expected Outcome: A page in BookStack that shows the embedded "Hello World" web app.

Final Deliverable

Please record a short demo video showing:

- The BookStack app running in Docker.
- The embedded "Hello World" app functioning in BookStack.
- A brief explanation of your setup.

Upload the video to the following Google Drive folder:

Upload Link

Questions or Issues?

If you run into any problems or have questions, reach out to Thian-Peng directly on Signal.

Digital Signature Form

Overview

This project will help you gain hands-on experience with modern frontend technologies and Google Apps Script integration. You will build a digital Code of Conduct application form for Children's Ministry as a Google Apps Script web app, designed to be embedded into our BookStack knowledge base and to support digital signature and PDF output.

You are encouraged to use AI code assistants (such as Cursor and GitHub Copilot) to accelerate your development and improve code quality.

Objectives

- Build a Code of Conduct application form as a Google Apps Script web app, using HTML5, ES6+, CSS3, and Web Components for the frontend.
- Accept digital signatures via the web app.
- Generate a PDF of the submitted application, and save it to a specified Google Shared Drive.
- Store form submissions in a Google Sheet for tracking and management.
- Embed the web app as an iframe in a BookStack page.
- Consult with your mentor (Thian-Peng) as needed—especially regarding digital signature capture and PDF generation/storage.

Prerequisites

- BookStack Official Documentation
- Docker Documentation
- Google Apps Script Documentation
- Web Components Overview

Tasks

1. Plan & Research

- Study the sample forms at:
  - https://training.saskcac.ca/children-ministry/code-conduct
  - https://training.saskcac.ca/children-ministry/application
- Review how digital signatures are captured in HTML5 (e.g., using `<canvas>` and libraries like Signature Pad).
- Research options for PDF generation and saving files to a Shared Drive via Google Apps Script (e.g., using PDF-LIB, or Apps Script's built-in capabilities).
- Consult with your mentor before final implementation of digital signature and PDF storage.

2. Set Up Your Development Environment

- Set up a new Google Apps Script project.
- Scaffold your frontend using HTML5, CSS, ES6+, and Web Components. (Consider using VS Code, Cursor, or other AI-enhanced IDEs.)
- Use a modular approach and write clean, maintainable code.

3. Build the Application Form Web App

- Recreate the Code of Conduct application form as the first milestone.
  - Include all required fields (name, email, etc.).
  - Add digital signature capture.
- Validate and submit the form data to the backend (Google Apps Script).
- Implement form data handling in Apps Script.
- Set up Google Sheets integration to log all form submissions with timestamp, applicant details, and status.

4. Set Up Google Sheets Integration

- Create a Google Sheet to store form submissions and track application status.
- Use Apps Script's SpreadsheetApp service to append form data automatically.
- Include columns for: timestamp, name, email, signature status, PDF file ID, and drive link.
- Implement error handling for sheet operations to ensure data integrity.
- Consider adding a simple dashboard view using Google Sheets' built-in charts for application tracking.

5. Generate and Store PDF

- On submit, generate a PDF version of the completed application form, including the digital signature.
- Store the generated PDF in the designated Google Shared Drive folder.
- Update the corresponding Google Sheets row with PDF file ID and drive link.
- Mark submission status as "completed" in the sheet.
- Ensure only authorized users can access the PDF (discuss with mentor).

6. Deploy and Embed

- Deploy the web app publicly or with appropriate permissions.
- Embed it into a BookStack page using an iframe:

```
<iframe src="YOUR_WEB_APP_URL_HERE" width="100%" height="800"
style="border:none;"></iframe>
```

- Test the full workflow.

Final Deliverable

- A working Google Apps Script web app that replicates the Code of Conduct form, accepts digital signatures, generates a PDF, and stores it in a Google Shared Drive.
- Demo video (2-5 minutes) showing:
  - The web app running (form fill + signature + PDF output)
  - The Google Sheet being updated with submission data
  - The PDF saved on Google Drive with tracking in the sheet
  - The form embedded in BookStack

- Upload the video to this folder: Upload Link

Questions or Issues?

- If you get stuck or have any questions (especially about digital signatures or PDF storage), reach out to Thian-Peng directly via Signal or your usual channel.

Tip: Don’t reinvent the wheel—use AI code assistants like Cursor and GitHub Copilot to help you learn and move faster.

Church Offering Deposit Form

Overview

This project builds upon the digital signature form project to create a mobile-friendly offering deposit entry form. The form will streamline the process of recording multiple donations collected during church offerings, with digital signatures from two counting staff members.

Due Date: August 31, 2025

Objectives

- Build a mobile-optimized offering deposit form as a Google Apps Script web app
- Support rapid entry of multiple deposits with minimal friction
- Implement dynamic data loading from Google Sheets for staff names and fund types
- Capture digital signatures from two counting staff members
- Generate timestamped PDF records with signatures
- Store all data in Google Sheets for tracking and reporting
- Embed the application in BookStack for easy access

Prerequisites

Before starting, ensure you have completed the digital signature project and are familiar with:

- Google Apps Script web app development
- HTML5, CSS3, ES6+, and Web Components
- Digital signature capture using canvas/Signature Pad
- PDF generation and Google Drive integration
- Google Sheets API operations
- Mobile-responsive design principles

Database Structure Setup

1. Create Required Google Sheets

Create three Google Sheets in your project folder:

Sheet 1: "Offering Deposits" (Main Data Sheet)

Columns: Date, Name, Payment Form, Amount, Offering, Note, Counter 1, Counter 2, Timestamp, PDF File ID, Drive Link, Status

Sheet 2: "Offering Counting Staff"

Columns: Name, Active Status Pre-populate with:

- Xiaoying Shen
- Weihong Xue
- Sinnie Lee
- Lisa Hsieh

Sheet 3: "Funds"

Columns: Fund Name, Active Status, Display Order Pre-populate with:

- General
- Church Building
- Benevolent
- Missions
- Short-Term Mission
- Capital Projects
- Next Generation Leadership

Development Tasks

1. Create Efficient Deposit Entry Interface
   Design for Speed:
   - Use a repeatable row component for deposits
   - Implement "Add Another" functionality
   - Auto-calculate running totals
   - Validate data as user enters data
2. Implement Dual Digital Signature
3. Backend Data Processing
4. PDF Generation with Signatures
5. Data Validation & Error Handling
6. Optimize
   - When entering the person who makes the donation, provide the ability to select people who make donations previously to allow rapid selection.

Technical Specifications

Mobile Breakpoints:

- Ensure usability on both portrait and landscape
- Test on iOS Safari and Android Chrome

User Experience Flow

1. **Setup Phase:** Select date and two counting staff members
2. **Data Entry:** Add deposits one by one with quick workflow
3. **Review:** Show summary with totals by fund type
4. **Signatures:** Capture both staff signatures
5. **Submission:** Generate PDF and save to sheets
6. **Confirmation:** Show success message with PDF link

Final Deliverable

Required Outputs:

- Fully functional mobile-optimized web app
- Integration with three Google Sheets
- PDF generation with digital signatures
- BookStack embedded implementation
- Demo video (3-5 minutes) showing:
  - Mobile form usage workflow
  - Multiple deposit entries
  - Signature capture process
  - Generated PDF output
  - Google Sheets data updates

Testing Checklist:

- ☐ Mobile responsiveness across devices
- ☐ Data validation and error handling
- ☐ Signature capture functionality
- ☐ PDF generation with proper formatting
- ☐ Google Sheets integration
- ☐ Performance under typical usage loads

Questions or Issues?

If you encounter challenges during development, especially with mobile optimization or performance issues, reach out to Thian-Peng directly via Signal.

Recommended Tools:

- Use Cursor or GitHub Copilot for accelerated development
- Test on actual mobile devices, not just browser dev tools
- Consider using Chrome DevTools' mobile simulation for initial testing

---

This project emphasizes practical mobile UX for rapid data entry while maintaining the robust backend architecture from the digital signature project.