AccuFund Migration Steps

Mr. MoneyBags v1.x — macOS Local Installation

Step-by-Step Migration Checklist

Mr. MoneyBags Logo

Last updated: August 2025

Table of Contents

> *Use this document as a punch-list while you migrate.*
> *All steps assume macOS (Apple Silicon or Intel).*

Legend

☐ = action not started ☑ = completed 📂 = file or folder path

💻 = command shell

---

1 Pre-Migration Checklist (≈ ½ day)

#	Task	Who	Done
1.1	□ Confirm executive approval & budget	Sponsor	
1.2	□ Freeze non-essential AccuFund customization changes	IT	
1.3	□ Verify latest AccuFund license & version 9.x	IT	
1.4	□ Schedule migration window (ideally Fri 18:00 → Sun 12:00)	Finance	
1.5	□ Email downtime notice to users (template in `📂 communications/`)	Comms	
1.6	□ Snapshot AccuFund VM or run **Full Backup** utility	IT	
1.7	□ Verify backup SHA-256 matches log	IT	
1.8	□ Confirm local Mac target installation is ready (Mr. MoneyBags runs on this Mac)	IT	

Common pitfalls

- Forgetting to lock end-users ➜ results in delta data loss.
- Skipping checksum validation ➜ corrupt backup undetected.

2 Local Mac Setup (≈ 30–45 min)

1. **Prerequisites (Homebrew)**
    - Homebrew – install from https://brew.sh/
    - Node.js LTS 18 or 20:

      ```
      💻 brew install node
      ```

    - Postman ≥ 10 for API testing

2. **Clone repository**

    ```
    💻 git clone https://github.com/org/mr-moneybags.git ~/mr-moneybags
    ```

3. **Configure environment**

    ```
    💻
    cd ~/mr-moneybags
    cp .env.example .env   # if not present
    ```

    Edit `.env`:

    ```
    PORT=3000
    CORS_ORIGINS=http://localhost:3000
    ```

4. **Bootstrap database & sample data**

    ```
    💻 npm run bootstrap:mac
    ```

    • Installs PostgreSQL via Homebrew if missing
    • Creates database (schema version 2025-08-15-02)
    • Loads Principle Foundation dataset by default

5. **Start the server**

    ```
    💻 npm run start
    ```

   Browse `http://localhost:3000` and log in with the admin credentials set during bootstrap.

6. **Troubleshooting**
   - Port 3000 busy → change `PORT` in `.env` then restart.
   - CORS errors → ensure your origin is listed in `CORS_ORIGINS`.

---

3 Data Extraction Steps (≈ 2 hours)

Step	Action	AccuFund Screen	Output
3.1	□ Run **GL** → **Export** → **Chart of Accounts**	General Ledger	`gl_accounts.csv`
3.2	□ Run **GL** → **Export** → **Funds**	Funds	`funds.csv`
3.3	□ Export **Vendors** with bank info	AP → Vendors	`vendors.csv`
3.4	□ Export **Bank Accounts** list	Banking → Bank Accounts	`bank_accounts.csv`
3.5	□ Export **Journal Detail** per fiscal year	Reports → GL Detail	`je_YYYY.csv`
3.6	□ Export **Outstanding Checks**	AP → Checks	`open_checks.csv`
3.7	□ Export **Open Deposits**	Cash Receipt → Deposits	`open_deposits.csv`

Save all files in 📂 `/migration_exports/YYYY-MM-DD`.

4 CSV Import Steps (Postman) (≈ 1.5 – 3 hours)

Prerequisites

- Entities, Chart of Accounts, and Funds **must already exist** in Mr. MoneyBags.
    - Use `npm run bootstrap:mac` or the Admin UI to create any missing codes.
- Files must be comma-separated CSV with headers.
- Recommended date format: **YYYY-MM-DD**. Amount columns may include `$` or `,`; these are stripped automatically.

Authentication (required for Postman)

Before calling any `/api/import/*` endpoints you must authenticate so that Postman stores the session cookie:

	Setting	Value
Method	**POST**	
URL	`http://localhost:3000/api/auth/login`	
Headers	`Content-Type: application/json`	
Body (raw JSON)	`{ "username": "admin", "password": "yourPassword" }`	

If the response is *200 OK* with “Login successful” Postman will capture the `mmb.sid` cookie automatically. All subsequent requests in this collection will be sent with that cookie and be authenticated.

4.1 Analyze your CSV

	Setting	Value
Method	**POST**	
URL	`http://localhost:3000/api/import/analyze`	
Body	**form-data** → `file` *(type = File)* → choose your exported CSV	

Response contains `headers`, `sampleData`, and a `suggestedMapping` object such as:

```
{
  "transactionId": "TransactionID",
  "entryDate": "Date",
  "accountCode": "AccountCode",
  "fundCode": "FundCode",
  "debit": "Debit",
  "credit": "Credit",
  "description": "Description"
}
```

4.2 Validate the CSV (no JSON conversion needed)

	Setting	Value
Method	**POST**	
URL	`http://localhost:3000/api/import/validate-csv`	
Body	**form-data**	
	`file` → your CSV file `mapping` *(type = Text)* → paste JSON mapping from step 4.1	

Response:

```
{
  "isValid": true,
  "issues": [],
  "summary": {
    "totalRows": 1234,
    "uniqueTransactions": 456,
    "unbalancedTransactions": 0
  }
}
```

4.3 Process (import) the CSV

Same body as **Validate**, but send to:
`POST http://localhost:3000/api/import/process-csv`

Returns **202 Accepted** with `{ "importId": "uuid-here" }`.

4.4 Monitor progress

`GET http://localhost:3000/api/import/status/:importId`
Shows `status`, `progress`, counters, and any errors.

4.5 Rollback if needed

`POST http://localhost:3000/api/import/rollback/:importId`
Deletes all journal entries created by that import.

Common pitfalls

- **Account code not found** → create the code in Settings → Chart of Accounts or fix the CSV.
- **Unbalanced transaction** → lines sharing a Transaction ID must have equal debits & credits.
- **Date parse errors** → use YYYY-MM-DD.
- **Large files (100 k+ lines)** → split by fiscal year.

---

5 Banking Setup Steps (≈ 1 hour)

macOS local install *– you can run* `npm run bootstrap:mac` *to set up PostgreSQL, load the sample dataset, and create* `.env` *automatically.*

#	Action	UI Path	Done
5.1	□ Import bank accounts (`bank_accounts.csv`)	Settings → Bank Accounts → Import	
5.2	□ Connect live feeds / upload first statements	Bank Reconciliation → Bank Statements	
5.3	□ Configure default check format (11-inch voucher)	Check Printing → Check Formats	
5.4	□ Enter opening cleared balances	Bank Reconciliation → New Reconciliation	

---

6 User Authentication Setup (≈ 30 min)

1. **Create admin**

   💻

   ```
   npm run create-admin -- \
     --user admin --email admin@example.org --password
   'TempP@ss123!'
   ```

2. **Bulk-import users**
   - Save file as 📂 `users_import.csv` (`full_name,email,role`)
   - UI → Settings → Users → Import CSV.

3. **Role audit**
   - ☐ Confirm Settings tab only visible to `admin` role.
   - ☐ Finance role can post JE but cannot manage users.

Pitfall: Copy-pasting passwords with trailing space ➜ login failure.

---

7 Testing & Validation Steps (≈ 4 hours)

Task	Tool / Location	Expected	Done
□ Compare record counts (staging vs prod)	💻 `sp_compare_counts()`	0 variance	
□ Trial balance per entity	Reports → Trial Balance	Matches AccuFund	
□ Bank rec ending balance	Bank Reconciliation report	Equals statement	
□ Random JE drill-down (5/mo)	JE screen	Debit = Credit	
□ User login & permissions	Login as each role	Proper access	
□ Print sample check to blank paper	Check Printing	Fields align	

8 Go-Live Steps (≈ 2 hours)

1. □ **Freeze** AccuFund data entry (set to read-only).
2. □ Export delta transactions (last 48 h) and import via CSV using the Postman flow.
3. □ Verify critical user logins and permissions on this Mac instance.
4. □ Send “System Live” email with local access instructions.
5. □ Monitor server output in Terminal (`npm run start`) for 2 hours.
6. □ Archive / retire AccuFund (snapshot + power off).

9 Post-Migration Steps (≈ 1 day)

#	Action	Owner	Done
9.1	□ Schedule nightly `pg_dump` cron job	IT	
9.2	□ Conduct user training webinar (slides in `📂 training/`)	Trainer	
9.3	□ Review first week reconciliations	Finance	
9.4	□ Document any custom report gaps	Finance	
9.5	□ Retire AccuFund license / contract	Procurement	

A Required Tools

Tool	Version	Purpose
PostgreSQL client (`psql`)	16.x	DB inspection / maintenance (optional)
AccuFund 9.x	Latest	Data export
Spreadsheet software	n/a	Mapping sheets
bcrypt-cli	2.x	Password hashing
Postman	≥ 10.0	API testing / file uploads
Homebrew	latest	Package manager (PostgreSQL install)
Node.js LTS	18.x or 20.x	Runtime for server & scripts

B Time & Effort Summary

Phase	**Est. Hours**
Preparation & Backup	4
Environment Setup	1
Extraction & Import	5
Banking & Auth Config	1.5
Testing & Validation	4
Go-Live	2
Post-Migration	8
Total	**25.5 hrs**

C Appendix - Useful Commands

Action	**Command**
Dump prod DB	`pg_dump -U postgres -Fc mrmoneybags > db_$(date +%F).dump`
Restore dump	`pg_restore -U postgres -d mrmoneybags db.dump`
Reset admin PW	`npm run reset-password -- --user admin --password 'NewP@ss!'`

Prepared August 2025 – San Francisco AI Factory