



POWER BANK INVESTMENT

Hostinger New Domain Installation & Operations Guide

Installation • Database • Permissions • Cron • Admin Settings • User Guide • Investment Flow

IMPORTANT NAMING NOTE

This documentation/product title is “Power Bank Investment”. The supplied application itself remains “LG Investment” in its code, database content and user/admin interface. Do not rename internal script branding as part of installation.

Prepared for Hostinger deployment on a fresh domain or subdomain

Edition: September 2026

Installation Roadmap

01	Before you upload	Server requirements, backup, domain and database preparation
02	Hostinger deployment	Create domain/subdomain, upload ZIP, extract to public_html
03	Database & configuration	Import SQL and edit both database configuration locations
04	Permissions & cache	0775 writable directories, storage/cache checks and cache clear
05	Cron installation	Use the application /cron endpoint every 5 minutes
06	Admin setup	General settings, plans, payment methods, users, deposits and withdrawals
07	User operations	Registration/login, deposit, investment, history and withdrawal
08	Maintenance	Backup, update workflow, troubleshooting and security checklist

Recommended installation method

For this final build, use manual Hostinger installation with the supplied project ZIP + SQL dump. The project already contains the Composer vendor directory, so a normal upload does not require running composer install.

1. Server Requirements

The bundled installer checks for PHP 8.1 or newer and the following PHP capabilities. Hostinger should be configured before the project is opened in the browser.

PHP	8.1 or newer (PHP 8.2 is a good target for this Laravel 9 build)
Database	MySQL/MariaDB with utf8mb4 support
Extensions	BCMath, CType, cURL, DOM, Fileinfo, GD, JSON, Mbstring, OpenSSL, PCRE, PDO, pdo_mysql, Tokenizer, XML
Web server	Apache/LiteSpeed with .htaccess + mod_rewrite-compatible rewrite support
Writable paths	core/bootstrap/cache, core/storage and its app/framework/logs directories
SSL	HTTPS certificate enabled for the new domain
Cron	Ability to call one HTTP URL every 5 minutes

Do not install on PHP 7.x

Although the SQL export header may show an older phpMyAdmin PHP version, the application installer explicitly requires PHP 8.1+. Use Hostinger hPanel → Advanced → PHP Configuration and select PHP 8.1+.

Pre-upload checklist

- Keep one untouched backup of the source ZIP and SQL file.
- Decide the final new domain/subdomain, for example <https://investment.example.com>.
- Create a new empty database and database user. Do not reuse a production database.
- Enable SSL before final testing so APP_URL and asset URLs can use https:// from the beginning.
- Never publish live database credentials or an APP_KEY in marketplace screenshots/documentation.

2. Create a New Domain / Subdomain in Hostinger

1. Open Hostinger hPanel and choose the hosting plan where the script will run.
2. Open Websites → Add website / Create website, or create a subdomain under Domains if you prefer a subdomain installation.
3. Point the domain DNS to the Hostinger hosting account. Wait until the domain resolves to the server.
4. Open Files → File Manager and confirm the document root. For a normal website this is commonly `public_html`; for an addon/subdomain Hostinger may create a domain-specific `public_html` folder.
5. Open Security → SSL and activate the free SSL certificate. Confirm `https://YOURDOMAIN` opens without a certificate warning.

Document root rule

The supplied ZIP is already structured to run from the web root. Files such as `index.php`, `.htaccess`, `assets/`, `main/`, `pages/`, `config/` and `core/` must sit directly inside the domain document root - not inside an extra nested `public_html` folder.

Correct structure after extraction

```
public_html/  
├─ index.php  
├─ .htaccess  
├─ assets/  
├─ config/  
├─ core/  
├─ main/  
├─ pages/  
├─ task/  
├─ payout/  
├─ install/  
└─ ...
```

3. Create Database and Import SQL

6. In hPanel open Databases → MySQL Databases.
7. Create a new database, create a database user and assign the user to that database with all privileges.
8. Save these four values: DB host, DB name, DB username and DB password. On Hostinger the host is usually localhost for a database on the same account.
9. Open phpMyAdmin for the new database.
10. Choose Import → select the supplied SQL file → keep UTF-8/utf8mb4 defaults → Import.
11. After import, verify that tables such as admins, users, deposits, invests, plans, cron_jobs and cron_schedules exist.

SQL included with this release

The supplied database dump already contains the application schema, cron job definitions and current application data. If you intend to sell/distribute a clean copy, sanitize real users, transactions and payment details before delivery. For your own deployment, import the intended final database backup.

Database verification

Tables to check	admins, users, deposits, invests, plans, transactions, cron_jobs, cron_schedules
Cron schedule	The database contains a “5 Minutes” schedule with interval 300 seconds.
Admin account	An admin record exists in the dump. Use the supplied admin credentials or reset the admin password after installation.

4. Upload and Extract the Script

12. Open File Manager → the new domain document root.
13. Remove only default placeholder files created for the empty website, such as a default index.html, if present.
Do not delete Hostinger system folders outside the web root.
14. Upload the final project ZIP.
15. Right-click → Extract. If the ZIP creates one extra top-level folder, move that folder's contents directly into the document root.
16. Confirm that root index.php can see core/vendor/autoload.php. The final project already bundles vendor/autoload.php, so normal deployment does not need Composer.
17. Keep the .htaccess file. It routes non-file requests into the Laravel front controller and blocks direct .env access.

Large ZIP upload tip

If File Manager upload times out, upload through Hostinger FTP/SFTP and then extract from File Manager. Do not upload the application file-by-file unless necessary; missing hidden files such as .htaccess can break routes.

5. Configure the New Domain and Database

This build uses two configuration locations because it combines the Laravel core with legacy PHP modules. Update both locations with the same database credentials and the new HTTPS domain.

A. Root configuration: config/config.php

```
'site' => [
    'name' => 'LG Investment',
    'url' => 'https://YOURDOMAIN.com',
    'timezone' => 'Asia/Kolkata',
],
'db' => [
    'host' => 'localhost',
    'port' => '3306',
    'name' => 'YOUR_DB_NAME',
    'username' => 'YOUR_DB_USER',
    'password' => 'YOUR_DB_PASSWORD',
    'charset' => 'utf8mb4',
],
```

B. Laravel configuration: core/.env

```
APP_NAME="LG Investment"
APP_ENV=production
APP_DEBUG=false
APP_URL=https://YOURDOMAIN.com
ASSET_URL=https://YOURDOMAIN.com

DB_CONNECTION=mysql
DB_HOST=localhost
DB_PORT=3306
DB_DATABASE=YOUR_DB_NAME
DB_USERNAME=YOUR_DB_USER
DB_PASSWORD="YOUR_DB_PASSWORD"
```

Keep LG Investment inside the script

For this product package, “Power Bank Investment” is the documentation/product name only. Keep APP_NAME and the internal site name as LG Investment unless the end customer intentionally performs a separate rebrand.

APP_KEY

The supplied build already contains an APP_KEY. For a private clone/new independent installation, generating a fresh Laravel APP_KEY is preferable if SSH/Terminal is available. Do not change APP_KEY on an existing live database if encrypted values depend on it.

```
cd ~/domains/YOURDOMAIN.com/public_html/core
php artisan key:generate --force
```

6. File Permissions and Writable Directories

The project's installer checks the following paths for 0775-style write permission. On Hostinger, directories normally work with 755, but this build explicitly checks for 0775; use 775 on these writable application directories if required.

core/bootstrap/cache	775
core/storage	775
core/storage/app	775
core/storage/framework	775
core/storage/logs	775

Using File Manager

18. Right-click each path → Permissions.
19. Set owner/group writable as supported by the Hostinger account. Use 775 for the listed directories; files can normally remain 644.
20. Do not set the whole public_html tree to 777.

Using SSH (optional)

```
cd ~/domains/YOURDOMAIN.com/public_html
chmod -R 775 core/storage core/bootstrap/cache
find core/storage -type f -exec chmod 664 {} \;
```

If you get “Permission denied” or 500 errors

Check core/storage/logs, core/storage/framework/views, sessions, cache and core/bootstrap/cache first. These are the most common writable-path failures after moving a Laravel project.

7. First Boot, Cache Clear and Health Check

21. Open <https://YOURDOMAIN.com>. Guests should be redirected to the user login route.
22. Open <https://YOURDOMAIN.com/user/login> and confirm CSS/images load normally.
23. Open <https://YOURDOMAIN.com/admin> and confirm the admin login page appears.
24. If you changed .env or the domain and see old settings, clear Laravel caches.

SSH commands

```
cd ~/domains/YOURDOMAIN.com/public_html/core  
php artisan optimize:clear  
php artisan config:clear  
php artisan route:clear  
php artisan view:clear
```

Browser cache-clear route included in this build

```
https://YOURDOMAIN.com/clear
```

Use /clear only during setup

The route calls Laravel optimize:clear. After deployment, use it only when you intentionally need to clear caches; do not repeatedly expose or share it as a normal user feature.

8. Install the Required Cron Job

This application has its own web cron controller. The database's default job schedule uses 300 seconds (5 minutes), and the /cron endpoint executes due jobs such as investment interest and scheduled investments, records cron logs, and updates each job's next_run time.

Primary cron URL

<https://YOURDOMAIN.com/cron>

Hostinger hPanel method

25. Open hPanel → Advanced → Cron Jobs.
26. Create a new custom cron job.
27. Schedule it every 5 minutes: Minute */5, Hour *, Day *, Month *, Weekday *.
28. Use an HTTP command that calls the project cron endpoint.

```
curl -fsS https://YOURDOMAIN.com/cron >/dev/null 2>&1
```

If curl is unavailable in the cron environment

```
wget -q -O /dev/null https://YOURDOMAIN.com/cron
```

Test before saving

Open the cron URL once in a browser. A healthy response is JSON containing fields similar to ok, site, time, due_jobs, executed, errors and next_check_seconds. When there are no errors, HTTP status is 200.

Admin cron monitoring

Cron manager	https://YOURDOMAIN.com/admin/cron/index
Main endpoint	https://YOURDOMAIN.com/cron
Recommended external frequency	Every 5 minutes
Expected internal schedule	300 seconds for the default 5 Minutes schedule

Do not create one Hostinger cron per internal job

The /cron controller already checks the cron_jobs table and executes every due active job. One Hostinger cron calling /cron every 5 minutes is the normal setup.

Legacy PHP cron files

The source also contains legacy utility files such as econ.php and main/monthlyplancron.php. They are not referenced by the main Laravel cron scheduler. Do not schedule them by default unless you specifically need that legacy custom behavior and have tested it separately.

9. Admin First Login and Essential Settings

Admin URL: <https://YOURDOMAIN.com/admin>

After the first login, change the admin password and review the settings below before allowing users to transact.

Area	Path	What to verify
Dashboard	/admin/dashboard	Review users, deposits, withdrawals, investment activity and notifications.
Admin Profile	/admin/profile	Update admin identity/profile image as needed.
Admin Password	/admin/password	Change the default/transferred admin password immediately.
General Settings	/admin/general-setting	Check site-wide investment, registration and system behavior toggles.
System Configuration	/admin/setting/system-configuration	Review platform/system configuration.
Logo & Icon	/admin/setting/logo-icon	Confirm LG Investment logo/favicon assets.
Payment Methods	/admin/ui-content/payment-methods	Create/edit UPI, Bank and USDT TRC20 destinations; show/hide active methods.
Cron Jobs	/admin/cron/index	Confirm Interest Cron and Schedule Invest are active and last run updates.
Notifications	/admin/notifications	Review animated unread alerts and action links.

10. Configure Plans, Time, Referral and Ranking

Before taking real deposits, configure the investment rules from the admin panel. The exact earning percentages, product amounts and durations are business settings - verify them against the intended model before launch.

Plans	/admin/plan - create/update plans and enable/disable status.
Time Settings	/admin/time - configure available time/duration settings used by the plan system.
Referral	/admin/referrals - configure referral levels/rewards and enable/disable levels.
User Ranking	/admin/user/ranking/list - manage ranking records/status.
Promotional Tool	/admin/promotional-tool - manage promotional assets shown to users.
Investment reports	/admin/invest/report/dashboard and report menus - review investment activity before/after cron runs.

Launch rule

Create at least one active plan/time combination, verify its return logic using a test user, then confirm the cron updates earnings as expected before opening registration to real users.

Recommended test

29. Create or use one test user.
30. Credit only a small test balance or use a controlled test deposit.
31. Invest in one plan.
32. Open /user/invest/log and confirm the investment record.
33. Run /cron manually once if the job is due.
34. Check cron logs and the user investment/balance records.
35. Only after the test succeeds, enable production payment details.

11. Configure Deposit Payment Methods

Admin path: <https://YOURDOMAIN.com/admin/ui-content/payment-methods>

The current payment-method manager supports multiple destinations and card-wise controls. Configure only methods that belong to the new installation.

UPI	Set title, UPI label/ID, instructions, QR image, sort order and Visible status.
Bank	Set bank name, account holder, account number, IFSC, extra details, QR if used and visibility.
USDT TRC20	Set TRC20 network, wallet address, QR image, instructions, order and visibility.
Hide/Show	Hidden methods should not be offered to users. Verify from a user deposit page after every change.
QR files	Use clear square QR images and test scanning from another phone before launch.

Deposit approval flow

36. User opens /user/deposit and selects an enabled payment method.
37. User enters amount and follows the shown payment instructions.
38. For manual payment, the user submits required transaction/proof information.
39. Admin reviews /admin/deposit and opens the deposit details.
40. Admin verifies the proof/UTR and approves or rejects according to the actual payment received.
41. After approval, verify the user wallet/balance and transaction history.

UTR search utility

This build also contains /main/searchbyupi.php for UTR-based lookup. Treat it as an admin utility and restrict access appropriately in production.

12. User Management and Withdrawal Administration

User manager: <https://YOURDOMAIN.com/admin/users>

All/Active/Banned Users	Use the Manage Users filters to find accounts by status.
User Detail	Open <code>/admin/users/detail/{id}</code> to inspect account information.
Direct Login	Admin can use the user login action for support/testing; use carefully.
Balance tools	Admin has add/subtract balance controls. Record a reason and avoid manual changes without an audit trail.
KYC	Review pending KYC if the KYC feature is enabled.
Notifications	Send user notifications where needed.

Withdrawal flow

42. User saves/uses payout account information and submits a withdrawal request.
43. Admin reviews Link Card / payout account verification under the Withdrawals menu.
44. Approve the user's Bank/UPI/USDT payout details only after verifying the information.
45. Review Pending Withdrawals; verify wallet balance, requested amount and payout account.
46. Pay externally according to the chosen method, then approve the withdrawal in admin. Reject with a clear reason if validation fails.
47. Verify the record appears in Approved or Rejected Withdrawals and user history.

Operational control

Do not approve a withdrawal only because the user submitted it. Verify the linked payout account, balance, transaction history and actual payout status first.

13. User Guide - Registration, Login and Account

Main domain	https://YOURDOMAIN.com - guests are redirected to Sign In.
User Login	https://YOURDOMAIN.com/user/login
Registration	https://YOURDOMAIN.com/user/register
Dashboard	https://YOURDOMAIN.com/user/dashboard
Profile	https://YOURDOMAIN.com/user/profile-setting
Change Password	https://YOURDOMAIN.com/user/change-password
Transactions	https://YOURDOMAIN.com/user/transactions
Deposit History	https://YOURDOMAIN.com/user/deposit/history
Referrals	https://YOURDOMAIN.com/user/referrals

New user flow

48. Open the registration page and create the user account with valid information.
49. Complete email/mobile/other verification only if those verification switches are enabled by admin.
50. Log in and open the dashboard.
51. Complete profile/KYC requirements if the installation requires them.
52. Deposit funds using an enabled payment method and wait for approval when using manual payment.
53. After wallet credit, proceed to investment.

14. User Investment Guide

The project exposes its investment actions through the user investment routes and displays investment history at `/user/invest/log`.

54. Login to the user account and confirm the available wallet balance is enough for the selected product/plan.
55. Open the investment/product area from the dashboard and choose the desired active investment option.
56. Review the displayed amount, duration/time, return conditions and any capital/interest handling option before confirmation.
57. Submit the investment. The backend creates the investment record and related transaction/notification data according to the configured plan.
58. Open Investment Log to confirm the new record: <https://YOURDOMAIN.com/user/invest/log>
59. Use Investment Details where available to review the individual investment.
60. Cron processing handles due interest/scheduled investment operations. Keep the 5-minute cron active continuously.
61. Review earnings/balance and transaction history. Never rely only on a UI animation; confirm the stored transaction/investment record.

Useful user investment paths

Investment Log	<code>/user/invest/log</code>
Investment Statistics	<code>/user/invest/statistics</code>
Investment Details	<code>/user/invest/details/{id}</code>
Investment Ranking	<code>/user/invest/ranking</code>
Scheduled Investments	<code>/user/invest/schedule</code>

Financial configuration warning

The software executes the plan and return settings configured by the site administrator. Documentation does not define or guarantee profit. Test every plan calculation and legal/compliance requirement before production use.

15. User Withdrawal Guide

Withdraw Request	https://YOURDOMAIN.com/user/withdraw
Withdraw History	https://YOURDOMAIN.com/user/withdraw/history

62. User logs in and completes required KYC if KYC is enabled.
63. User ensures payout/Link Card details are correct and approved when the installation requires verification.
64. Open Withdraw, choose the available method and enter a valid amount within configured limits.
65. Review the preview page and submit the request.
66. The request remains pending until admin processing.
67. After admin approval/rejection, check Withdraw History for the final status.

User safety

Users should never send passwords, OTPs or private wallet keys to an administrator. Payout verification should use account/payment details and transaction evidence only.

16. Backup and Safe Update Procedure

Before changing code, payment settings or database structure, take both a file backup and database backup.

68. Hostinger → Files → Backups: create/download a backup of the domain files, or compress public_html into a ZIP.
69. phpMyAdmin → Export → Quick/Custom SQL export of the current database.
70. Record the current domain, PHP version, cron command and any custom payment settings.
71. Apply updates to a staging copy first when possible.
72. After code replacement, preserve production config/config.php and core/.env credentials unless the update explicitly requires changes.
73. Run cache clear and test user login, admin login, deposit, invest log, cron and withdrawal pages.
74. If anything fails, restore both files and database from the same backup point.

Do not mix database versions casually

A code rollback with a newer incompatible database (or vice versa) can create subtle errors. Treat file backup + SQL backup as one deployment snapshot.

17. Troubleshooting

Problem	Check / Fix
500 Internal Server Error	Check core/storage/logs, PHP 8.1+, required extensions, .env DB values and writable storage/bootstrap/cache permissions.
Database connection error	Recheck DB_DATABASE, DB_USERNAME, DB_PASSWORD in core/.env and the same values in config/config.php. Confirm the DB user has privileges.
CSS/images missing	Confirm APP_URL and ASSET_URL use the correct https domain; clear caches; verify assets/ sits in the web root.
Routes return 404	Confirm root .htaccess exists and the project is not one directory too deep.
Login loops / session fails	Check HTTPS URL consistency and core/storage/framework/sessions write permission.
Cron output not updating	Open /cron manually; inspect JSON errors; check /admin/cron/index and cron logs; verify Hostinger schedule is every 5 minutes.
Investment return not processing	Confirm relevant cron job is active, next_run is due, general investment switches are enabled and plan/time settings are valid.
QR/payment method not visible	Check method visibility/status in /admin/ui-content/payment-methods and test as a normal user, not only in admin.
Old Blade layout after update	Run /clear or php artisan optimize:clear and refresh browser cache.
Upload too large	Use SFTP/FTP or adjust Hostinger PHP/upload limits; project vendor directory makes the archive large.

18. Final Go-Live Checklist

- ☐ New domain resolves to Hostinger and SSL is active.
- ☐ Project files are directly in the correct document root.
- ☐ PHP 8.1+ and required extensions are enabled.
- ☐ New database created; SQL imported successfully.
- ☐ config/config.php updated with new domain/database values.
- ☐ core/.env updated with the same DB values and correct APP_URL/ASSET_URL.
- ☐ core/storage and core/bootstrap/cache are writable.
- ☐ User login and admin login both open without 500/404 errors.
- ☐ Admin password changed after installation.
- ☐ Plans/time/referral/ranking settings reviewed.
- ☐ UPI/Bank/USDT payment destinations checked and only intended methods are visible.
- ☐ One manual deposit tested end-to-end.
- ☐ One test investment created and visible in /user/invest/log.
- ☐ Hostinger cron calls https://YOURDOMAIN.com/cron every 5 minutes.
- ☐ Cron manager/logs confirm successful executions.
- ☐ Withdrawal/Link Card verification and approval flow tested.
- ☐ Notification alerts tested for unread/read state.
- ☐ Final file + database backup created after successful setup.

FINAL RESULT

When every item above passes, Power Bank Investment is installed on the new Hostinger domain. The operating application continues to display and run as LG Investment, exactly as supplied.

End of documentation