Absolute Raid Bot: Setup and Admin Guide

Version 1.0.6 · for Absolute Raid Management 0.2.0 and Absolute Raid Companion · Absolute Gaming LLC

1. What the bot is

Absolute Raid Bot is your guild's **DKP book, run from Discord.** It is one of three pieces that work together:

Piece	Who uses it	What it does
Absolute Raid Management (addon, 0.2.0)	The raid leader in game	Runs the raid: sealed loot bidding, reserves, attendance, presence checks, boss and presence DKP.
Absolute Raid Companion (addon)	Every raider in game	Shows the raider's DKP live, lets them reserve loot and bid privately, and answers presence checks.
Absolute Raid Bot (this)	Officers and raiders in Discord	Keeps the guild's master DKP book between raids: registrations, standings, adjustments, the audit log, reports and backups.

The bot is the source of truth between raids. The addon is the source of truth during a raid. Before each raid, the bot hands the addon a **snapshot** of everyone's DKP. After the raid, the addon hands the bot an **export** of what changed. The bot adds those changes to its book.

What it can do

- **Raiders:** register their characters, then check their DKP, the standings, their history and loot.
- **Admins:**
  - approve registrations; add and remove users and characters
  - adjust DKP, with a required reason that is kept forever
  - set how much DKP a raid earns
  - make snapshots, import raids, make a navy and orange PDF report
  - back up and restore the book
- **Two-factor codes** on every admin command.
- **An audit log** that can't be changed or erased.
- **Wowhead look-ups** for items and boss loot, for WoW Forever and Retail.

2. What you need

- A **computer or server that stays on** while the bot runs. A home PC works when it's on; a small VPS or a Raspberry Pi works all the time.
- **Python 3.11 or newer** (python.org; on Windows, tick Add Python to PATH when installing).
- A Discord server where you have **Manage Server** permission.
- An **authenticator app** on your phone: Google Authenticator, Microsoft Authenticator, Authy, 1Password or Bitwarden.

3. Create the bot in Discord

1. Open **https://discord.com/developers/applications** and sign in.
2. Click **New Application**, name it **Absolute Raid Bot**, and accept the terms.
3. Optional: on **General Information**, upload the bot's picture (the dragon logo from the bot's `assets` folder).
4. Open the **Bot** page.
   - Click **Reset Token**, then **Copy**. That's the bot's password. **Never share it or post it.** If it leaks, come back and reset it.
   - The bot needs **no** privileged intents. Leave Presence, Server Members and Message Content off.

5. Open **OAuth2 -> URL Generator**.
   - **Scopes:** tick `bot` and `applications.commands`.
   - **Bot permissions:** tick **Send Messages**, **Embed Links** and **Attach Files**.
   - Copy the link at the bottom, open it in your browser, choose your server, and click **Authorize**.
6. Get your IDs. In Discord, open **User Settings -> Advanced** and turn on **Developer Mode**. Then:
   - right-click your server's icon and choose **Copy Server ID**. This is `guild_id`.
   - right-click your own name and choose **Copy User ID**. This is `owner_id` (optional).
   - Optional: right-click a staff-only channel and choose **Copy Channel ID**. This is `log_channel_id`, where the bot posts every DKP change.

4. Install and start the bot

> **Hosting in Microsoft Azure?** Follow **Absolute-Raid-Bot-Azure-Setup.pdf** instead of this section: it builds a small Linux VM with the bot installed as a service, the firewall locked down, and the pairing code ready. Then continue at §4.5 (pairing) and §5.

4.1 Unzip and install

Unzip **AbsoluteRaidBot-1.0.6.zip** where the bot will live (for example `C:\AbsoluteRaidBot`). Open a terminal in the `absolute-raid-bot` folder and run:

```
python -m pip install -r requirements.txt
```

That installs discord.py, reportlab (for the PDF report) and cryptography (for 2FA secrets and backups).

4.2 config.json

Copy `config.example.json` to `config.json` and edit it:

Setting	Put in
`owner_id`	Optional. Your Discord user ID: you become the bot's **owner** and first admin. Leave it empty and whoever pairs the bot (§4.5) becomes the owner.
`guild_id`	Your server's ID. Commands appear there straight away (instead of taking up to an hour).
`log_channel_id`	Optional: the staff channel's ID for DKP changes, approvals and imports.
`require_2fa`	Leave it `true`.
`keep_backups`	Daily backups to keep (default 14).
`snapshots_private`	`true`: only the admin who asks sees the snapshot file.
`wowhead_paths`	Leave it as it is unless Wowhead moves its WoW Forever pages.

4.3 The token

Put the token in an environment variable rather than in a file.

- **Windows PowerShell:** `$env:DISCORD_TOKEN = "paste-the-token"`
- **Windows, permanently:** `setx DISCORD_TOKEN "paste-the-token"`, then open a new terminal.
- **Linux/macOS:** `export DISCORD_TOKEN="paste-the-token"`

(You can put it in `config.json` as `"token"`, but then never share that file.)

4.4 Start it

```
python run.py
```

The terminal should show **"Absolute Raid Bot 1.0.6 ready"**. In Discord, type `/dkp` and the commands appear.

4.5 Pair the bot with your server

The first time the bot starts it makes a one-time **pairing code** like `K7QM-2XVD-9FHT`. It prints it in the terminal (or the service log) and saves it in `data/pairing-code.txt`. Until the code is entered, the bot answers nothing but the pairing command.

In your Discord server, a member with **Manage Server** runs:

```
/dkpadmin pair code:K7QM-2XVD-9FHT
```

- After pairing the bot works only in this server. If it's invited anywhere else it leaves.
- The code works once and is deleted. Five wrong codes and a new one is made.
- If `owner_id` was empty, whoever paired becomes the owner.
- To move the bot to a different server: stop it, run `python run.py --new-pairing-code` in the bot folder, start it, and pair again. It's written to the audit log.

4.6 Keep it running

- **Windows:** in Task Scheduler, create a task that runs `python run.py` with Start in set to the bot folder, triggered at log-on (or at start-up). Under Settings, turn on If the task fails, restart every 1 minute.
- **Linux:** run it as a systemd service with `Restart=always`.
- **Docker:** the bot includes a Dockerfile:

`docker build -t absolute-raid-bot .` then

```
docker run -d --restart unless-stopped -e DISCORD_TOKEN=... -v /path/data:/app/data -v /path/config.json:/app/config.json absolute-raid-bot
```

4.7 The data folder

Everything the bot keeps is in `data`:

File	What it is
`arb.db`	The DKP book: users, characters, transactions, raids, loot, settings.
`audit.db`	The audit log (kept separately so a restore can never touch it).
`secret.key`	Encrypts 2FA secrets and backups. **Copy it somewhere safe today** (a password manager or a USB stick). Without it, backups can't be opened.
`backups/`	The daily automatic backups, plus a safety copy made before any restore.
`pairing-code.txt`	The pairing code, only until the bot is paired.

5. First-time setup in Discord

Do these once, in order.

0. **Pair the bot** if you haven't yet (§4.5): `/dkpadmin pair code:<the code>`.
1. **Turn on your two-factor** (see §7): `/dkpadmin 2fa-setup`, then `/dkpadmin 2fa-confirm code:123456`.
2. **Name the guild:** `/dkpadmin setting key: Guild name value: Absolute Gaming`.
3. **Set DKP per raid:** `/dkpadmin set-dkp per_raid: 10 per_boss: 0`. Every raider present at a raid night earns `per_raid`. Raise `per_boss` to pay for each boss kill as well.
4. **Check the other rules:** `/dkpadmin settings` (each one is explained in §10).
5. **Register your officers:** `/dkpadmin admin-add member: @Officer code: 123456`. Owner only, with a fresh code. Each officer gets their two-factor key by DM and finishes with `/dkpadmin 2fa-confirm` within 24 hours (§7).
6. **Tell raiders to register:** `/dkp register` (see §6).

6. Users and registration

6.1 How a raider registers

The raider types:

`/dkp register character: Brannoc-Stormrage class: Warrior`

- Use the character's name, or **Name-Realm** for a character on another realm. The class is optional.
- With approval on (the default), the registration **waits for an admin**. The log channel shows New registration.
- If the character is already in the book (because it was seen in a raid import), the raider is simply **claiming** it, and it keeps all its DKP.
- A raider can register several characters (alts). `/dkp me` shows each one.

6.2 Approving

Command	What it does
`/dkpmanage pending`	Everything waiting, with who asked.
`/dkpmanage approve character: Brannoc-Stormrage`	The character is linked to that member.
`/dkpmanage deny character: Brannoc-Stormrage reason: Not in the guild`	Turned down; the reason is logged.

To skip approval: `/dkpadmin setting key: Registrations need approval value: 0`.

6.3 Adding, removing and linking (admins)

Command	What it does
`/dkpmanage add-character character: Seraphine class: Priest member: @Sera`	Adds a character, linked to a member if you name one. Adding a removed character back restores it **with its history and DKP.**
`/dkpmanage delete-character character: Seraphine reason: Left the guild`	Hides the character. Its history and the audit log are kept. A reason is required.
`/dkpmanage delete-user member: @Sera reason: Left the guild`	Removes the member's registration. Their characters stay in the book, unlinked. A reason is required.

Characters that show up in a raid import but aren't registered are **created automatically**, so their DKP builds up. When the player later registers that name, they claim it.

7. Two-factor (2FA) for admins

Two-factor is built into admin registration and can't be turned off. Step-by-step instructions are in **Admin Registration and Account Recovery**.

The admin commands come in two groups, because Discord allows at most 25 commands in one group: `/dkpadmin` (setup, security, raids, data) and `/dkpmanage` (registrations, characters and DKP changes). One unlock covers both.

7.1 Registering

- **The owner** registers when pairing the bot: the pairing reply holds their two-factor key. (An owner set in config runs `/dkpadmin 2fa-setup`.)

- **Officers** are registered by the owner with `/dkpadmin admin-add member: @Officer code: 123456`. They start as **pending** and get their key by DM (or with `/dkpadmin 2fa-setup`).
- Scan the QR code in the bot's reply with an authenticator app (no camera: **Enter a setup key** and type the key under it), then `/dkpadmin 2fa-confirm code: 123456`. Only then is the person an admin. An officer who doesn't finish within **24 hours** has to be registered again.
- Finishing gives **10 one-time recovery codes**, shown once. Store them outside Discord.
- `/dkpadmin admins` lists every admin's status and recovery codes left.

7.2 Day to day

- **Unlock:** `/dkpadmin unlock code: 123456` opens admin commands for **15 minutes**. **Lock early:** `/dkpadmin lock`.
- **Step-up codes:** `admin-add`, `admin-remove`, `restore`, applying `import-db` and `recovery-codes` need a fresh code typed into the command, even when unlocked.
- Wrong codes are written to the audit log. If codes keep failing, set your phone's clock to automatic.

7.3 A lost phone

- **Any admin with a recovery code:** `/dkpadmin recover code: K7QM-2XVD`. The old authenticator stops working, a new key is shown, and `2fa-confirm` gives a fresh set of codes.
- **An officer with no codes left:** the owner runs `admin-remove` then `admin-add`.
- **The owner with no codes left:** on the bot's machine, `python run.py --reset-2fa YOUR_DISCORD_USER_ID` (Azure: `sudo arb-reset-2fa <id>`), then `/dkpadmin 2fa-setup` and `2fa-confirm`. It's written to the audit log.
- New codes any time: `/dkpadmin recovery-codes code: 123456`.

8. Running a raid with Absolute Raid Management

This is the loop for every raid night.

8.1 Before the raid: the snapshot

1. An admin runs `/dkpadmin snapshot pool: Main`.
   - `pool` is the name the points pool gets in the addon (keep using the same one).
   - The bot replies with a file called **dkp-snapshot-xxxxxxxx.txt**, holding every active character's DKP and a check number.
2. The **raid leader** downloads that file and opens it in Notepad. Press **Ctrl+A**, then **Ctrl+C**.
3. In game, the raid leader types `/arm import`. (The addon also asks for the snapshot by itself when you form a raid without a fresh one.) Click the box, press **Ctrl+V**, then click **Load snapshot**.
4. The addon checks the paste. A paste cut short, or a line changed while copying, is refused with a message saying so. If that happens, copy the whole file again.
5. You'll see "Snapshot xxxxxxxx loaded: N raiders". That pool is now tonight's DKP. Raid Companion raiders see their snapshot balance straight away.

> Loading a snapshot **replaces** that pool's balances and history in the addon. That's intended: the bot is the master book. It doesn't touch any other pool.

8.2 During the raid

Everything happens in the addon.

- **Loot:** drops appear on the LOOT page. Right-click a drop for **Start bidding**, **Start bidding, reservers only**, a roll, or **Award to...**.
  - Bidding is **sealed**: raiders bid from Raid Companion, the addon's pop-up, or a whisper, and only you see the amounts.
  - The winner pays their bid and only the winner hears the price.
- **DKP for being there:** boss kills (if Give boss kill DKP automatically is ticked), the **On-time bonus**, **Give to raid**, and **presence ticks** every few minutes (TOOLS -> Presence tick every).

- **Presence checks:** each of those asks every Raid Companion raider "are you here?". The ATTENDANCE page's **Companion** column shows who answered. With Presence DKP needs Raid Companion ticked, only raiders who answered (and aren't AFK) are paid.
- **In-game corrections** (for example a late penalty) go through the POINTS page as usual. They are sent to the bot with the export.

8.3 After the raid: the export

1. When you leave the raid, the addon reminds you. Type `/arm export bot` (or ATTENDANCE -> Export... -> For Absolute Raid Bot).
2. A window opens with the export already selected. Press **Ctrl+C**.
3. Open Notepad, press **Ctrl+V**, and **save it as** `raid-YYYY-MM-DD.json`. In Save as type, choose All files so Notepad doesn't add `.txt`; the bot accepts `.txt` too.
4. In Discord, an admin runs `/dkpadmin import` and attaches that file.

The bot answers with a summary:

Line	Meaning
Raid nights added	New nights, with who was there and the bosses killed. Nights it already had are skipped.
DKP earned	`dkp_per_raid` for each raider present, plus `dkp_per_boss` × their boss kills (and the addon's own awards if `earn_source` is `addon` or `both` and a snapshot was loaded).
DKP spent	Every item won, at its winning bid, plus in-game penalties.
New characters	Raiders the bot hadn't seen; they were added automatically.
No raid DKP (Raid Companion didn't confirm them)	Shown when `require_companion` is on.
Warnings	For example: no snapshot was loaded for this raid, an older snapshot was used, or someone is below zero.

8.4 Why it can't double-count

- Importing the **same file twice** is refused.
- Every night, boss payment, item and in-game change has its own key. An overlapping export (say, a whole week) only adds what is new.
- The export carries **tonight's changes**, never whole balances. An adjustment an admin made in Discord during the raid is kept.

9. Managing DKP

Command	What it does
`/dkpmanage adjust character: Tarnok amount: 25 reason: Bench pay`	Adds DKP. Use a negative amount to take it (for example `-10`). **A reason is required.**
`/dkpmanage set-balance character: Tarnok balance: 150 reason: Correcting the import`	Sets an exact balance. It is recorded as an adjustment of the difference, with the reason.
`/dkpadmin set-dkp per_raid: 10 per_boss: 2`	What a raid night (and each boss kill) earns.
`/dkpadmin audit character: Tarnok`	Every change to that character: who made it, when, and why.

Nothing is ever overwritten. A balance is the sum of a character's transactions, and transactions can only be added. A correction is a new line, so the history always shows what happened.

Negative DKP: an admin adjustment can't take a balance below zero unless `allow_negative` is `1`. Loot won in game is always charged; the import warns you if someone goes below zero.

10. Settings

Change any of these with `/dkpadmin setting`. Each change is logged.

Setting	Default	Meaning
DKP per raid night (`dkp_per_raid`)	10	Earned by every raider present at a raid night.
DKP per boss kill (`dkp_per_boss`)	0	Earned per boss kill the raider was there for.
Earn source (`earn_source`)	bot	`bot`: the bot pays the two amounts above. `addon`: use what the addon awarded in game (boss kills, on-time, ticks; needs that night's snapshot loaded in game). `both`: both.
Start balance (`start_balance`)	0	A new character starts with this.
Allow negative DKP (`allow_negative`)	0	`1` lets admin adjustments go below zero.
Registrations need approval (`require_approval`)	1	`0` makes `/dkp register` link at once.
Raid DKP needs Raid Companion presence (`require_companion`)	0	`1`: raid DKP only for raiders whose Raid Companion answered at least one presence check that night.
Guild name, Report title		Shown on the standings and the PDF.
Default game for loot (`game`)	forever	`forever` or `retail`, for `/dkp loot`.

11. Backups and restores

11.1 Automatic

Every day at **09:00 UTC** (2:00 AM Arizona time), the bot writes an encrypted backup to `data/backups` and keeps the newest 14 (see `keep_backups`).

11.2 On demand

`/dkpadmin backup note: before season reset` makes a backup and **DMs it to you** as `absolute-raid-bot-YYYYMMDD-HHMMSS.arbbak`. If your DMs are closed, it shows the file only to you instead.

- A backup holds the DKP book and a copy of the audit log, with SHA-256 checksums.
- It is encrypted with `data/secret.key`: **only this bot (or a bot with a copy of that key) can open it.** Keep the key safe, and keep backups off the bot's machine too.

11.3 Restoring

`/dkpadmin restore file: (attach the .arbbak) code: 123456 confirm: RESTORE`

1. The bot checks the file: the right key, matching checksums, and a database integrity check.

2. It saves a **safety backup** of the book as it is now in `data/backups`, so a restore can be undone by restoring that.
3. It puts the backup's DKP book in place: users, characters, balances, raids and settings.
4. **The audit log is never restored or rolled back.** It keeps everything, including the restore itself, so a restore can't hide a change.

11.4 The DKP database in and out (no server access needed)

- `/dkpadmin export-db format: CSV / JSON / Both` gives you the database: a CSV with one row per character (character, class, discord_id, balance, earned, spent, raids, attendance_pct) or the whole book as JSON. It isn't encrypted.
- `/dkpadmin import-db file:` loads balances from a CSV (columns `character` and `balance`, `class` optional) or a JSON export. It previews first; run it again with `confirm: IMPORT` and your 2FA `code` to apply. Each character ends up with exactly the file's balance, recorded as a transaction and in the audit log; the same file twice changes nothing.
- Use them to start from another DKP system, move to a new bot without touching its server, or fix many balances at once. Full details: Importing and Exporting Data.

11.5 Moving the bot to a new machine

Stop the bot, copy the whole `data` folder and `config.json` to the new machine, install (§4.1), and start it. Or make a backup, set up the new bot, copy `secret.key` into its `data` folder, and restore the backup there.

12. The audit log

- **What's recorded:**
  - registrations, approvals and denials
  - additions and deletions of characters and users
  - every DKP adjustment, with the reason and the balance before and after
  - setting changes, imports (with the file's fingerprint), snapshots, reports, backups and restores
  - admin changes, 2FA set-ups and resets, and failed 2FA codes
- **It can't be changed.** It lives in its own database (`audit.db`), which refuses to edit or delete rows.
- **Tampering shows.** Every row carries a SHA-256 hash of the row before it. If someone edits the file outside the bot, the chain breaks. `/dkpadmin audit-verify` checks the whole chain and names the first entry that doesn't match.
- **Read it** with `/dkpadmin audit` (the latest 20), `/dkpadmin audit character: Name`, and `count:` up to 50.
- With `log_channel_id` set, changes are also posted live to your staff channel.

13. Reports and loot

- `/dkpadmin report` makes the PDF: navy and orange, with the logo, totals, full standings (class-colored, with earned, spent, raids and attendance), recent raid nights, recent loot, and the guild's rules.
  - Add `public: True` to post it in the channel.
  - It is also safe to share with the guild.
- `/dkp standings top: 25` is a quick table in the channel.
- `/dkp loot item: 17068` looks the item up on Wowhead (WoW Forever by default; add `game:` for Retail) and shows who in your guild has won it.
- `/dkpadmin loot-fetch npc_id: 10184 name: Onyxia game: WoW Forever` reads a boss's drop list from its Wowhead page, so raiders can see it with `/dkp raidloot source: Onyxia`.
  - The NPC ID is the number in the boss's Wowhead address.
  - Wowhead has no official API. The bot fetches each item once, keeps it, and spaces requests out.
  - Items won in your raids are learned from the imports even without Wowhead.

14. Commands at a glance

Everyone (`/dkp`):

Command	
`register character class`	Claim your character.
`me`	Your DKP and recent history (private).
`check character`	Anyone's DKP.
`standings [top]`	The standings.
`history character [count]`	A character's history (private).
`loot item [game]`	Item look-up, and who won it here.
`raidloot source [game]`	A boss's loot list.
`help`	How the guild's DKP works.

Admins (`/dkpadmin`, unlocked with 2FA):

Command	
`pair code`	Pair the bot with this server and register the owner (Manage Server, before anything else).
`2fa-setup`, `2fa-confirm code`, `unlock code`, `lock`	Two-factor registration and unlocking.
`recover code`, `recovery-codes code`, `admins`	Recovery codes, and who's registered.
`set-dkp per_raid [per_boss]`, `setting key value`, `settings`	Rules.
`snapshot [pool]`, `import file`	The raid round trip.
`export-db [format]`, `import-db file [confirm] [code]`	The DKP database in and out (§11.4).
`report [public]`	The PDF.
`backup [note]`, `restore file code confirm`	Backups.
`audit [character] [count]`, `audit-verify`	The audit log.
`admin-add member code`, `admin-remove member code`	Owner only, with a fresh code.
`loot-fetch npc_id name [game]`	Wowhead boss loot.

Admins (`/dkpmanage`: raiders and their DKP, same 2FA unlock):

Command	
`pending`, `approve character`, `deny character reason`	Registrations.

Command	
`add-character character [class] [member]`	Add or restore a character.
`delete-character character reason`, `delete-user member reason`	Remove (history kept).
`adjust character amount reason`, `set-balance character balance reason`	Change DKP.
`link-character character member reason`	Move a character to a member (a raider's new Discord account).

In game (Absolute Raid Management 0.2.0): `/arm import` (load a snapshot) · `/arm export bot` (export tonight) · `/arm companion` (who runs Raid Companion).

15. Troubleshooting

Problem	Fix
The log says the bot isn't in your server, or "403 Missing Access"	Invite the bot to the server in `guild_id` (the log prints the link), and check `guild_id` is that server's ID.
The `/dkp` commands don't appear	Check `guild_id` in config.json and restart the bot. Without it, Discord can take up to an hour.
Every command says the bot isn't paired	Enter the pairing code: `/dkpadmin pair code:...` (§4.5). Lost it? It's in `data/pairing-code.txt`; on Azure, `sudo arb-status`.
The bot left a server by itself	It's paired with another server and only works there.
"Admins only"	Your Discord ID isn't an admin. The owner adds you with `/dkpadmin admin-add`.
"Finish your admin registration first", "Admin registration expired" or "Locked"	See §7.
The snapshot paste is refused in game	Copy all of the file (Ctrl+A first) and paste it again. The first line must be `ARB-SNAPSHOT 1` and the last `END`.
"This exact export was already imported"	Nothing to do: it's in.
"That file isn't an Absolute Raid Management export"	Make sure you saved the text from `/arm export bot`, not another export.
A backup "can't be opened with this bot's secret.key"	It was made by a bot with a different `secret.key`. Copy the right key into `data`.
`/dkp loot` can't reach Wowhead	Try again later. Items already known still work. If Wowhead moved its WoW Forever pages, update `wowhead_paths` in config.json.

Questions or bugs: our Discord, https://discord.gg/a8nM6yvMQM · https://absolutegaming.store