Auto Panel API

Create users, deposit, withdraw, update credit and change passwords on your panel sites through one HTTPS endpoint.

Version 2.2 · September 2026

Contents

- Endpoint and key
- Sites
- Request and reply
- createUser
- deposit
- withdraw
- updateCredit
- changePassword
- getTask, cancelTask, listTasks, verifyKey
- Field rules
- Errors
- Code samples

---

1. Endpoint and key

```
POST https://forms.apkfactoryproject.com/api
Content-Type: application/json
x-access-key: YOUR_ACCESS_KEY
```

- Every action uses this one URL.
- Keep your access key on your server only.
- The access key is for the API only. To sign in to the website (forms.apkfactoryproject.com), use the username and password the admin gave you — not the key.
- Wrong key → **401**. Disabled key → **403**. Too many wrong keys → **429** (wait 15 minutes).

2. Sites

Site	Send as `site`	Status
Allpanel	`allpanelexch9.game` (or `allpanel9.global`)	Live
Allpanel.buzz	`allpanel.buzz` (or `allpanel.buzz/admin/`)	Live
Silverexch1	`silverexch1.com` (or `silverexch1.com/admin/`)	Live
D247	`d247.com`	Live
Allpanel777	`777allid.com` (or `world777.now`)	Live
Diam247x	`diam247x.com`	Live
Kingexch365	`kingexch365.game`	Live
4wicket	`ag.4wicket.com` (or `4wicket.com`)	Live (credit reference: whole numbers)
Skyx4	`ag.skyx4.com` (or `skyx4.com`)	Live (credit reference: whole numbers)
Kingexch9	`ag.kingexch9.club` (or `kingexch9.club`)	Live (credit reference: whole numbers)
Lords	`lordsadmin.external247services.com` (or `lordsx.com`)	Live
Lotusbookx247	`admin.lotusbookx247.com` (or `lotusbookx247.com`, `lotus7book.com`)	Live
Betbhaiexch	`ag.betbhaiexch.com`	Live
Betcric	`admin.betcric.com` (or `betcric.com`)	Live (deposit / withdraw amounts are whole numbers)
Radheexch247	`admin.radheexch247.com` (or `radheexch247.com`)	Live (client names: letters and digits only)
Tenexch	`tenexch.com`	Live
Greyexch99	`greyexch99.com`	Live
Tigerexch	`tigerexch.pro`	Live (no `updateCredit`: the site has no credit reference; `changePassword` resets to the site's 123456, returned in the reply)
Tiger365	`tiger365.pro`	Live (no `updateCredit`: the site has no credit reference)
Betbhai9 (also Deal999, Lcplay, Powergreen)	`betbhai9.tax`, `deal999.com`, `lcplay.online` or `powergreen.club`	Live
Mytiger247 (also My99exch, Sunexch, Mylaser247)	`mytiger247.win`, `my99exch.info`, `sunexch.pro` or `mylaser247.pro`	Live

Send `site` on every request. The request runs on your own account on that site.

3. Request and reply

Request body

Field		Meaning
`site`	required	See Sites
`action`	required	`createUser`, `deposit`, `withdraw`, `updateCredit`, `changePassword`, `getTask`, `cancelTask`, `listTasks`, `verifyKey`
`data`	required	The action's fields (below)
`requestId`	recommended	Your own unique id (e.g. order id). Sending the same `requestId` again returns the same result and never repeats the job.
`wait`	optional	Default `true`: wait for the result. `false`: get a `taskId` at once and check it with `getTask`.

Reply

Every reply has `success` (true / false) and `message`, plus the action's fields. **Decide by `success`.**

HTTP	Meaning
200	Finished. Check `success`.
202	Accepted, not finished yet. Reply has `taskId`; check it with `getTask`.
400	Bad request; `message` says why. Nothing was done.
401 / 403 / 429	Key problem (see Endpoint and key)
409	This `requestId` was already used for a different action or site

Set your HTTP timeout to at least **110 seconds**.

`createUser` Create a user

Field		Rules
`clientName`	required	Letters, numbers and `_`, up to 30 characters
`password`	optional	Leave out and we create one. See Field rules.

```
{
  "site": "allpanelexch9.game",
  "action": "createUser",
  "data": { "clientName": "Rahul" },
  "requestId": "signup-5521"
}

→ { "success": true, "message": "Account created", "clientName": "Rahul4", "password": "Sophia@3857",
"passwordGenerated": true }
```

Use `clientName` and `password` from the reply: the created name can differ from the one you sent.

`deposit` Deposit

Field		Rules
`clientName`	required	The client's user name
`amount`	required	Greater than 0, up to 2 decimals
`remark`	optional	Up to 100 characters

```
{
  "site": "allpanelexch9.game",
  "action": "deposit",
  "data": { "clientName": "Rahul4", "amount": "500" },
  "requestId": "order-88213"
}

→ { "success": true, "message": "Balance deposited successfully", "clientName": "Rahul4", "amount": "500"
}
```

`withdraw` Withdraw

Field		Rules
`clientName`	required	The client's user name
`amount`	required	Greater than 0, up to 2 decimals
`remark`	optional	Up to 100 characters

```
{
  "site": "d247.com",
  "action": "withdraw",
  "data": { "clientName": "Rahul4", "amount": "300" },
  "requestId": "payout-4410"
}

→ { "success": true, "message": "Withdrawn 300", "clientName": "Rahul4", "amount": "300",
"availableBefore": 600, "availableAfter": 300 }
```

`availableBefore` / `availableAfter` are included on D247, Allpanel777, Tenexch, Greyexch99, Lords, Diam247x, Mytiger247, Betbhai9, Kingexch365, 4wicket, Allpanel.buzz, Betbhaiexch, Betcric, Kingexch9, Radheexch247, Silverexch1, Skyx4, Tigerexch, Tiger365 and Lotusbookx247 (deposit and withdraw). On Lotusbookx247 the site makes the client passwords itself (createUser, changePassword): the `password` in the reply is the one to use; a password you send is not used there.

`updateCredit` Update credit reference

Field		Rules
`clientName`	required	The client's user name
`credit`	required	The new credit reference: 0 or more, up to 2 decimals

```
{
  "site": "allpanelexch9.game",
  "action": "updateCredit",
  "data": { "clientName": "Rahul4", "credit": "5000" }
}

→ { "success": true, "message": "Credit updated successfully", "clientName": "Rahul4", "oldCredit": "0",
"newCredit": "5000" }
```

`changePassword` Change password

Field		Rules
`clientName`	required	The client's user name
`newPassword`	optional	Leave out and we create one. See Field rules.

```
{
  "site": "allpanelexch9.game",
  "action": "changePassword",
  "data": { "clientName": "Rahul4" }
}

→ { "success": true, "message": "Password changed successfully", "clientName": "Rahul4", "password":
"Martin@7427", "passwordGenerated": true }
```

On **Diam247x**, **Mytiger247** and **Betbhai9** the password is always reset to `123456`, and the reply gives `"password": "123456"`.

On **Tiger365** the password is always reset to `123456` (as on Diam247x), and `updateCredit` is refused.

On **Kingexch365** a password must be 8–20 characters with a letter and a digit (letters, digits and `!@#$%^&`). Leave it out and one is made for you.

`getTask` Check a request

```
{ "action": "getTask", "data": { "taskId": "muj2k8pq-3f9a1c2e" } }

→ { "success": true, "task": { "id": "muj2k8pq-3f9a1c2e", "status": "done",
      "result": { "success": true, "message": "Balance deposited successfully", "clientName": "Rahul4",
"amount": "500" }, … } }
```

`task.status`	Meaning
`queued`, `running`, `awaiting_approval`	Not finished yet: check again in a few seconds
`done`	Finished successfully; the reply is in `task.result`
`failed`	Not done; `task.result.message` says why
`cancelled`, `rejected`	Not done; nothing changed
`unknown`	Not confirmed. Check the client before sending it again.

`cancelTask` Cancel a request that has not started

```
{ "action": "cancelTask", "data": { "taskId": "muj2k8pq-3f9a1c2e" } }

→ { "success": true, "message": "Cancelled", "task": { …, "status": "cancelled" } }
```

`listTasks` List your requests

```
{ "action": "listTasks" }

→ { "success": true, "tasks": [ { "id": "…", "site": "allpanel", "action": "createUser", "status":
"done", … }, … ] }
```

`verifyKey` Check your key

```
{ "action": "verifyKey" }

→ { "success": true, "merchant": { "name": "YourCompany", "sites": ["allpanel", "d247"] }, "sites": [ … ]
}
```

4. Field rules

Field	Rule	Example
Client name	Letters, numbers, `_`; 1–30 characters	`Rahul_22`
Password	Optional. If sent: an uppercase letter, a lowercase letter and `@`; only letters, numbers and `@`; up to 40 characters	`Oliver@4729`
Amount	Greater than 0, up to 2 decimals	`500`, `250.50`
Credit	0 or more, up to 2 decimals	`5000`
Remark	Optional, up to 100 characters	`Top-up 26 Sep`
requestId	Letters, numbers and `_ . : -`; up to 80 characters	`order-88213`

5. Errors

When `success` is `false`, `message` says why. Common ones:

message	What to do
`Required: …` / `… can only contain …`	Fix the field (HTTP 400)
`Unknown site …` / `Unknown action …`	Check the spelling (HTTP 400)
`Your account has no … login yet`	Ask us to set up that site for you (HTTP 400)
`Client "X" not found …`	Check the client name
`Not enough balance: …`	Lower the amount
`… Check the client before retrying`	**Check the client's balance before sending again**

For deposit and withdraw, **only send the same request again with the same** `requestId`. A new `requestId` is a new deposit or withdrawal.

---

6. Code samples

curl

```
curl -X POST https://forms.apkfactoryproject.com/api \
  -H "Content-Type: application/json" \
  -H "x-access-key: YOUR_ACCESS_KEY" \
  --max-time 110 \
  -d '{"site":"allpanelexch9.game","action":"deposit","data":
{"clientName":"Rahul4","amount":"500"},"requestId":"order-88213"}'
```

JavaScript (Node 18+)

```
const API = 'https://forms.apkfactoryproject.com/api';
const KEY = process.env.PANEL_API_KEY;
const FINAL = ['done', 'failed', 'cancelled', 'rejected', 'unknown'];

async function call(body) {
  const r = await fetch(API, {
    method: 'POST',
    headers: { 'Content-Type': 'application/json', 'x-access-key': KEY },
    body: JSON.stringify(body),
    signal: AbortSignal.timeout(110_000),
  });
  return { status: r.status, body: await r.json() };
}

// Send, then check with getTask until finished.
async function run(site, action, data, requestId) {
  const q = await call({ site, action, data, requestId, wait: false });
  if (q.status !== 202) return q.body;
  for (;;) {
    await new Promise((r) => setTimeout(r, 3000));
    const { body } = await call({ action: 'getTask', data: { taskId: q.body.taskId } });
    if (FINAL.includes(body.task.status)) return body.task.result;
  }
}

const res = await run('allpanelexch9.game', 'createUser', { clientName: 'Rahul' }, 'signup-5521');
console.log(res.success, res.clientName, res.password);
```

Python (requests)

```
import os, time, requests

API = "https://forms.apkfactoryproject.com/api"
HEADERS = {"x-access-key": os.environ["PANEL_API_KEY"]}
FINAL = ("done", "failed", "cancelled", "rejected", "unknown")

def call(body):
    r = requests.post(API, json=body, headers=HEADERS, timeout=110)
    return r.status_code, r.json()

def run(site, action, data, request_id):
    status, body = call({"site": site, "action": action, "data": data, "requestId": request_id, "wait":
False})
    if status != 202:
        return body
    while True:
        time.sleep(3)
        _, t = call({"action": "getTask", "data": {"taskId": body["taskId"]}})
        if t["task"]["status"] in FINAL:
            return t["task"]["result"]

print(run("d247.com", "deposit", {"clientName": "Rahul4", "amount": "500"}, "order-88213"))
```

PHP

```php
<?php
function panel_call(array $body) {
    $ch = curl_init('https://forms.apkfactoryproject.com/api');
    curl_setopt_array($ch, [
        CURLOPT_POST => true,
        CURLOPT_RETURNTRANSFER => true,
        CURLOPT_TIMEOUT => 110,
        CURLOPT_HTTPHEADER => ['Content-Type: application/json', 'x-access-key: ' .
getenv('PANEL_API_KEY')],
        CURLOPT_POSTFIELDS => json_encode($body),
    ]);
    $res = json_decode(curl_exec($ch), true);
    $status = curl_getinfo($ch, CURLINFO_HTTP_CODE);
    curl_close($ch);
    return [$status, $res];
}

[$status, $res] = panel_call([
    'site' => 'allpanelexch9.game', 'action' => 'withdraw',
    'data' => ['clientName' => 'Rahul4', 'amount' => '200'], 'requestId' => 'payout-4410',
]);
if ($status === 202) { /* not finished yet: check getTask with $res['taskId'] */ }
echo $res['success'] ? 'OK: ' . $res['message'] : 'Not done: ' . $res['message'];
```