

Cool Frog Tap 2 Earn Clicker Game

Telegram Mini App + API + Bot



Summary

What is CoolFrog Tap2Earn?	3
How to setup Cool Frog game frontend.....	4
How to setup Cool Frog API.....	5
How to setup Cool Frog Telegram bot.....	7
How to run the game locally	8
How to add/customize levels, missions and tasks.....	9

What is CoolFrog Tap2Earn?

CoolFrog Tap2Earn is an exciting and addictive Telegram mini app clicker game where tapping isn't just fun—it's rewarding! Step into the world of CoolFrog, where every tap brings you closer to earning coins, leveling up, and dominating the leaderboard.

Features include:

- **Tap to Earn Coins:** Tap your way to wealth by earning coins with every click. The faster you tap, the more you earn!
- **Buy Boosters:** Supercharge your tapping power with a variety of boosters. Maximize your earnings and dominate the competition.
- **Refer Friends:** Invite your friends to join the fun and get bonus coins for each referral. The more friends you refer, the richer you become!
- **Complete Tasks:** Take on daily tasks and challenges to earn extra coins. Each task completed brings you closer to your next big reward.
- **Upgrade Missions:** Invest in upgrades to earn coins passively, even when you're not tapping. Let your frog work for you around the clock!
- **Level Up:** Use your coins to level up and unlock new features, bonuses, and more powerful upgrades.
- **Leaderboard:** Compete against players worldwide to claim your spot at the top. Show everyone who the ultimate CoolFrog is!

Whether you're in it for fun, competition, or just to see your name on the leaderboard, CoolFrog Tap2Earn is the game for you!

How to setup Cool Frog game frontend

First let's setup the frontend of the game, this is the telegram mini app that the bot will open.

1. Open your project in VS Code or any other IDE
2. Navigate to the `coolfrog` folder
3. Open the `.env` file
4. Set your domains where you will host the api and frontend, and set your telegram bot's url (how to create a telegram bot: <https://core.telegram.org/bots/tutorial>)
5. Open the terminal while located in the `coolfrog` folder
6. Run `npm install`

```
$ npm install
npm WARN deprecated inflight@1.0.6: This module is not supported, and leaks memory. Do not use it. Check out lru-cache
more comprehensive and powerful.
npm WARN deprecated @humanwhocodes/config-array@0.11.14: Use @eslint/config-array instead
npm WARN deprecated rimraf@3.0.2: Rimraf versions prior to v4 are no longer supported
npm WARN deprecated glob@7.2.3: Glob versions prior to v9 are no longer supported
npm WARN deprecated @humanwhocodes/object-schema@2.0.3: Use @eslint/object-schema instead

added 372 packages, and audited 373 packages in 12s

78 packages are looking for funding
  run `npm fund` for details
```

7. After installation, run `npm run build`

```
$ npm run build

> clicker-game@0.0.0 build
> tsc && vite build

vite v5.3.1 building for production...
✓ 1776 modules transformed.
dist/index.html          1.46 kB | gzip: 0.74 kB
dist/assets/index-Dyl-bBIT.css  50.23 kB | gzip: 11.63 kB
dist/assets/index-Bdv6_R-W.js  850.18 kB | gzip: 270.70 kB

(!) Some chunks are larger than 500 kB after minification. Consider:
- Using dynamic import() to code-split the application
- Use build.rollupOptions.output.manualChunks to improve chunking: https://rollupjs.org/configuration-options/#output-manualchunks
- Adjust chunk size limit for this warning via build.chunkSizeWarningLimit.
✓ built in 6.65s
```

8. If you want to run the game locally, you need to run `npm run dev`, otherwise you need to put the contents of the generated `dist` folder on your hosting

How to setup Cool Frog API

Next let's setup the API of the game, this is where the frontend makes requests to get and save data of the game.

1. Open your project in VS Code or any other IDE
2. Navigate to the `api` folder
3. Open the `.env` file
4. Set your database login data (username, password), APP_URL and APP_STORAGE_URL (should be the domain of your api)
5. Open the terminal while located in the `api` folder
6. Run `composer install` (you can download composer from here: <https://getcomposer.org/>)

```
- Installing theseer/tokenizer (1.2.3): Extracting archive
- Installing sebastian/lines-of-code (3.0.1): Extracting archive
- Installing sebastian/complexity (4.0.1): Extracting archive
- Installing sebastian/code-unit-reverse-lookup (4.0.1): Extracting archive
- Installing phpunit/php-code-coverage (11.0.5): Extracting archive
- Installing phar-io/version (3.2.1): Extracting archive
- Installing phar-io/manifest (2.0.4): Extracting archive
- Installing myclabs/deep-copy (1.12.0): Extracting archive
- Installing phpunit/phpunit (11.3.0): Extracting archive
Generating optimized autoload files
> Illuminate\Foundation\ComposerScripts::postAutoloadDump
> @php artisan package:discover --ansi

 INFO  Discovering packages.

laravel/breeze ..... DONE
laravel/sail ..... DONE
laravel/sanctum ..... DONE
laravel/tinker ..... DONE
nesbot/carbon ..... DONE
nunomaduro/collision ..... DONE
nunomaduro/termwind ..... DONE

79 packages you are using are looking for funding.
Use the `composer fund` command to find out more!
```

7. Make sure you have a mysql server running on your hosting or local environment
8. Run `php artisan migrate` (this will create the database)

```
$ php artisan migrate

 WARN  The database 'coolfrog' does not exist on the 'mysql' connection.

Would you like to create it? (yes/no) [yes]
> yes

 INFO  Preparing database.

Creating migration table ..... 15.28ms DONE

 INFO  Running migrations.

0001_01_01_000000_create_users_table ..... 65.34ms DONE
0001_01_01_000001_create_cache_table ..... 20.88ms DONE
0001_01_01_000002_create_jobs_table ..... 54.35ms DONE
2024_06_20_131332_create_personal_access_tokens_table ..... 35.26ms DONE
2024_06_21_071221_create_telegram_users_table ..... 23.27ms DONE
2024_06_21_071222_create_tasks_table ..... 112.62ms DONE
2024_06_21_072918_create_daily_tasks_table ..... 89.01ms DONE
2024_06_25_145341_create_levels_table ..... 6.81ms DONE
2024_07_03_131414_create_mission_types_table ..... 6.94ms DONE
2024_07_03_143422_create_missions_table ..... 45.73ms DONE
2024_07_03_152047_create_mission_levels_table ..... 42.66ms DONE
2024_07_03_152457_create_telegram_user_missions_table ..... 91.50ms DONE
2024_07_08_065814_add_image_to_tasks_table ..... 8.45ms DONE
2024_07_08_131311_add_requires_column_to_missions_table ..... 13.35ms DONE
2024_07_10_125312_change_from_to_balance_columns_type_in_levels_table ..... 46.53ms DONE
2024_07_15_075712_create_referral_tasks_table ..... 88.92ms DONE
2024_07_15_084737_create_popups_table ..... 43.97ms DONE
```

9. Run `php artisan db:seed` (this will seed the database with data – levels, missions etc.)

```
$ php artisan db:seed

INFO Seeding database.

Database\Seeders\LevelSeeder ..... RUNNING
Database\Seeders\LevelSeeder ..... 47 ms DONE

Database\Seeders\DailyTaskSeeder ..... RUNNING
Database\Seeders\DailyTaskSeeder ..... 37 ms DONE

Database\Seeders\TaskSeeder ..... RUNNING
Database\Seeders\TaskSeeder ..... 12 ms DONE

Database\Seeders\MissionTypeSeeder ..... RUNNING
Database\Seeders\MissionTypeSeeder ..... 4 ms DONE

Database\Seeders\MissionSeeder ..... RUNNING
Database\Seeders\MissionSeeder ..... 50 ms DONE
```

10. Run `php artisan key:generate`
11. Run `php artisan storage:link`
12. Now your API is ready to go, if you want to run it locally run `php artisan serve`, otherwise you need to put this code on your hosting

```
php artisan serve

INFO Server running on [http://127.0.0.1:8000].

Press Ctrl+C to stop the server
```

How to setup Cool Frog Telegram bot

Last but not least, let's setup the Telegram bot of the game so we can actually access it through Telegram.

1. Open your project in VS Code or any other IDE
2. Navigate to the `bot` folder
3. Open the `.env` file
4. Setup your APP_URL (this is the domain of the frontend) and Telegram bot token, which you get from Bot Father on Telegram after creating it.
5. Open the terminal while located in the `bot` folder
6. Run `npm install`

```
$ npm install
npm WARN deprecated inflight@1.0.6: This module is not supported, and leaks memory. Do not use it. Check out lru-cache
more comprehensive and powerful.
npm WARN deprecated rimraf@3.0.2: Rimraf versions prior to v4 are no longer supported
npm WARN deprecated glob@7.2.3: Glob versions prior to v9 are no longer supported
npm WARN deprecated core-js@2.6.12: core-js@<3.23.3 is no longer maintained and not recommended for usage due to the nu
could cause a slowdown up to 100x even if nothing is polyfilled. Some versions have web compatibility issues. Please, u

added 228 packages, and audited 229 packages in 6s

29 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
```

7. After installation, run `npm run build`

```
$ npm run build
> build
> rimraf ./dist && tsc
```

8. If you want to run the bot locally, you need to run `npm run dev`, otherwise you need to run the index.js file located in the generated `dist` folder on your hosting server (this can be done with pm2: <https://pm2.keymetrics.io/>)

That is all. Now you should be able to access your bot by sending the `/start` command to your bot and pressing the Play Game button!

How to run the game locally

Note that you can run the game locally (without bot, just game and api), but if you want it to be launchable through telegram, the game needs to be hosted on domain with SSL (https).

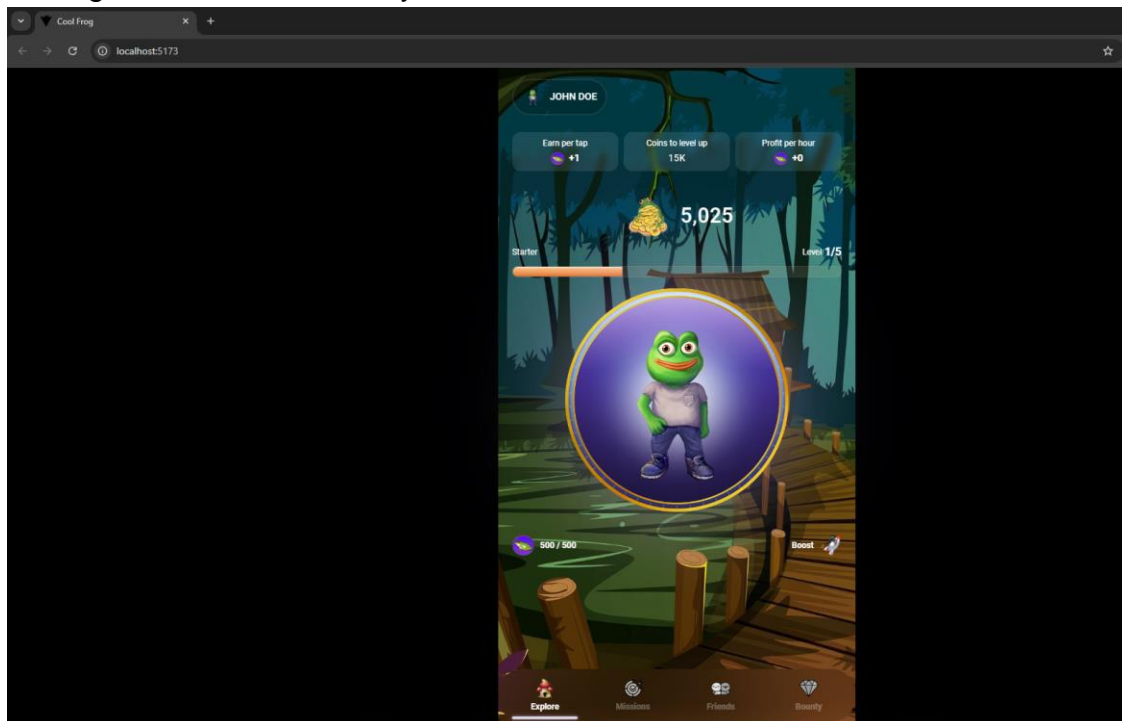
1. Setup your API as shown [here](#)
2. Setup your frontend as shown [here](#)
3. Next thing is to set the environment variables (in .env file) in your frontend to your local URLs as shown below:

```
.env X
coolfrog > .env
1 VITE_APP_URL="http://localhost:5173"
2 VITE_API_URL="http://127.0.0.1:8000"
3 VITE_BOT_URL="#"
```

You can leave the VITE_BOT_URL empty, since we won't be using the bot locally.

Also please note, that you may have different ports, change them accordingly.

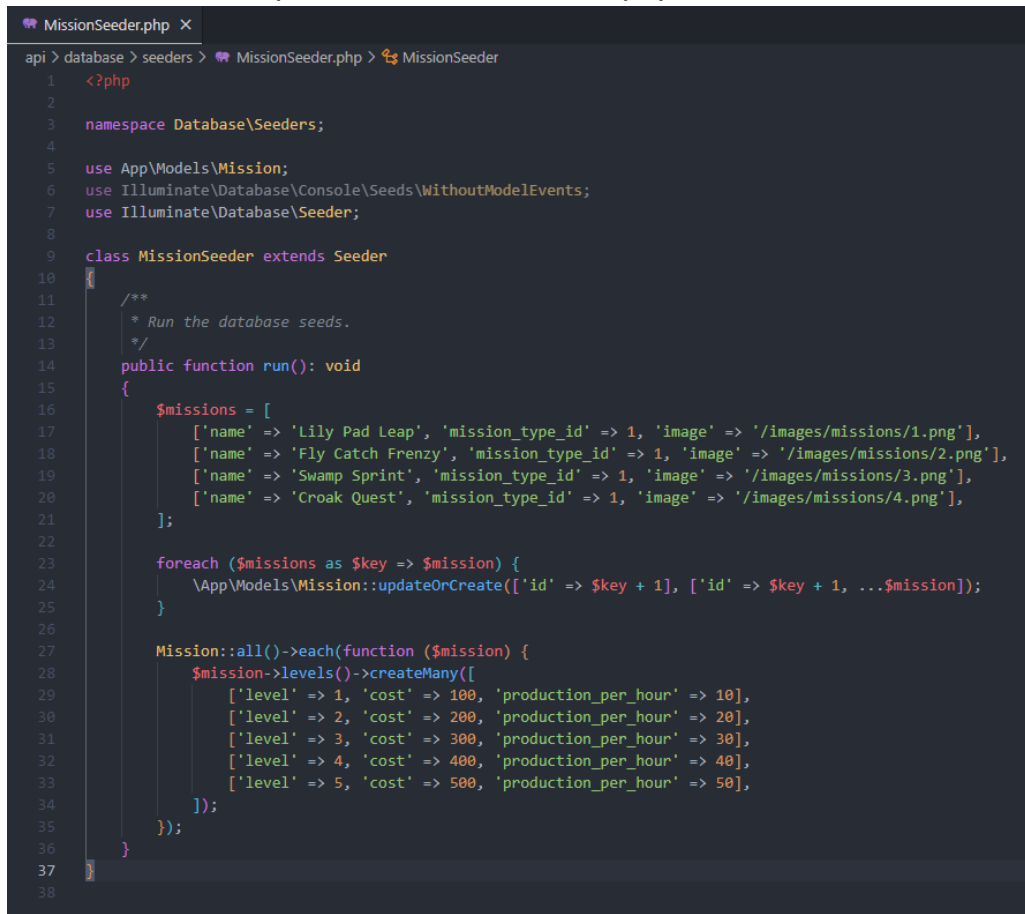
4. After you have set up both frontend and API, you should be able to access it by visiting the frontend URL in your browser.



How to add/customize levels, missions and tasks

The customization/adding of levels, missions and tasks is done in the API, because they are saved in the database. So, if you want to change them, please follow these steps:

1. Open your project in VS Code or any other IDE
2. Navigate to the `api` folder
3. Open the `database/seeder` folder, there you will have all the database seeder files
4. To add missions, open the `MissionSeeder.php` file, it should look like this:



```
1  <?php
2
3  namespace Database\Seeders;
4
5  use App\Models\Mission;
6  use Illuminate\Database\Console\Seeds\WithoutModelEvents;
7  use Illuminate\Database\Seeder;
8
9  class MissionSeeder extends Seeder
10 {
11     /**
12      * Run the database seeds.
13      */
14     public function run(): void
15     {
16         $missions = [
17             ['name' => 'Lily Pad Leap', 'mission_type_id' => 1, 'image' => '/images/missions/1.png'],
18             ['name' => 'Fly Catch Frenzy', 'mission_type_id' => 1, 'image' => '/images/missions/2.png'],
19             ['name' => 'Swamp Sprint', 'mission_type_id' => 1, 'image' => '/images/missions/3.png'],
20             ['name' => 'Croak Quest', 'mission_type_id' => 1, 'image' => '/images/missions/4.png'],
21         ];
22
23         foreach ($missions as $key => $mission) {
24             \App\Models\Mission::updateOrCreate(['id' => $key + 1], ['id' => $key + 1, ...$mission]);
25         }
26
27         Mission::all()->each(function ($mission) {
28             $mission->levels()->createMany([
29                 ['level' => 1, 'cost' => 100, 'production_per_hour' => 10],
30                 ['level' => 2, 'cost' => 200, 'production_per_hour' => 20],
31                 ['level' => 3, 'cost' => 300, 'production_per_hour' => 30],
32                 ['level' => 4, 'cost' => 400, 'production_per_hour' => 40],
33                 ['level' => 5, 'cost' => 500, 'production_per_hour' => 50],
34             ]);
35         });
36     }
37 }
38
```

5. Following the provided example, you can add your own missions and customize their level cost and production per hour.

6. If you want to add levels, open the `LevelSeeder.php` file, it should look like this:

```
LevelSeeder.php X
api > database > seeders > LevelSeeder.php > ...
1  <?php
2
3  namespace Database\Seeders;
4
5  use App\Models\Level;
6  use Illuminate\Database\Seeder;
7
8  class LevelSeeder extends Seeder
9  {
10     /**
11      * Run the database seeds.
12      */
13     public function run(): void
14     {
15         $levels = [
16             ['level' => 1, 'name' => 'Starter', 'from_balance' => 5_000, 'to_balance' => 15_000],
17             ['level' => 2, 'name' => 'Adventurer', 'from_balance' => 15_000, 'to_balance' => 50_000],
18             ['level' => 3, 'name' => 'Explorer', 'from_balance' => 50_000, 'to_balance' => 150_000],
19             ['level' => 4, 'name' => 'Traveler', 'from_balance' => 150_000, 'to_balance' => 500_000],
20             ['level' => 5, 'name' => 'Boss', 'from_balance' => 500_000, 'to_balance' => 1_000_000],
21         ];
22
23         foreach ($levels as $level) {
24             Level::updateOrCreate(['level' => $level['level']], $level);
25         }
26     }
27 }
28
```

7. Following the provided example, you can add your own levels and customize their required balance.

8. If you want to add tasks, open the `TaskSeeder.php` file, it should look like this:

```
TaskSeeder.php X
api > database > seeders > TaskSeeder.php > ...
1  <?php
2
3  namespace Database\Seeders;
4
5  use Illuminate\Database\Seeder;
6  use App\Models\Task;
7
8  class TaskSeeder extends Seeder
9  {
10     public function run()
11     {
12         $tasks = [
13             [
14                 'name' => 'Watch Tutorial Video',
15                 'description' => 'Watch our game tutorial video on YouTube.',
16                 'reward_coins' => 100,
17                 'link' => 'https://youtube.com',
18                 'type' => 'video',
19                 'action_name' => 'Watch Video',
20             ],
21             [
22                 'name' => 'Join Our Discord',
23                 'description' => 'Join our official Discord server and say hello in the #welcome channel.',
24                 'reward_coins' => 100,
25                 'link' => 'https://discord.gg/yourgame',
26                 'type' => 'other',
27                 'action_name' => 'Join'
28             ],
29             [
30                 'name' => 'Follow on Twitter',
31                 'description' => 'Follow our official Twitter account and retweet our pinned tweet.',
32                 'reward_coins' => 150,
33                 'link' => 'https://twitter.com/youngame',
34                 'type' => 'other',
35                 'action_name' => 'Follow'
36             ],
37         ],
38     }
39 }
```

9. Following the provided example, you can add your tasks and customize their rewarded coin amount.
10. The same goes for referral tasks (ReferralTaskSeeder.php), daily tasks (DailyTaskSeeder.php), popups (PopupSeeder.php) and mission types (MissionTypeSeeder.php).