



# Discord Matcha Restock Bot Setup Guide

## Prerequisites

- Node.js (v16 or higher)
- Discord Developer Account
- Basic server hosting (VPS, Heroku, Railway, etc.)

## Step 1: Discord Bot Setup

### Create Discord Application

1. Go to [Discord Developer Portal](#)
2. Click "New Application"
3. Name it "Matcha Bell" or similar
4. Go to "Bot" section
5. Click "Add Bot"
6. Copy the bot token (keep it secret!)

### Bot Permissions

Your bot needs these permissions:

- Send Messages
- Embed Links
- Add Reactions
- Manage Roles
- Read Message History
- Use Slash Commands

Permission Integer:

### Invite Bot to Server

Use this URL format (replace CLIENT\_ID with your bot's client ID):

```
https://discord.com/api/oauth2/authorize?client_id=YOUR_CLIENT_ID&permissions=268511296&scope=bot
```

## Step 2: Project Setup

### Install Dependencies

```
bash
```

```
npm init -y
```

```
npm install discord.js axios cheerio node-cron
```

## Project Structure

```
matcha-bot/
├── bot.js           # Main bot file (from artifact)
├── package.json
├── stock_data.json  # Auto-generated stock tracking
├── config.json      # Configuration file
├── vendor_configs/  # Vendor-specific scrapers
│   ├── marukyu.js
│   ├── ippodo.js
│   └── ...
```

## Environment Setup

Create a `.env` file:

```
DISCORD_TOKEN=your_bot_token_here
NODE_ENV=production
```

Or create `config.json`:

```
json
{
  "token": "your_bot_token_here",
  "checkInterval": 5,
  "timezone": "America/New_York"
}
```

## Step 3: Server Setup Commands

Once your bot is online, use these commands in your Discord server:

### Initial Setup

```
!matcha-setup
```

This creates the role selector with buttons for all vendors.

### Status Check

```
!matcha-status
```

Shows monitoring status and statistics.

## Step 4: Advanced Vendor Configuration

For better accuracy, you'll need to customize the stock checking for each vendor. Here's an example for Marukyu Koyamaen:

```
javascript

// vendor_configs/marukyu.js
module.exports = {
  name: 'Marukyu Koyamaen',
  baseUrl: 'https://www.marukyu-koyamaen.co.jp',
  checkStock: async (url) => {
    // Custom scraping logic for this vendor
    const response = await axios.get(url);
    const $ = cheerio.load(response.data);

    const inStock = !$('.out-of-stock').length;
    const price = $('.price').text().trim();

    return { inStock, price };
  },
  products: [
    {
      name: 'Wakatake',
      url: '/english/products/matcha/wakatake',
      image: 'https://example.com/wakatake.jpg'
    }
  ]
};
```

## Step 5: Hosting Options

### Option 1: Heroku (Free tier available)

1. Create Heroku account
2. Install Heroku CLI
3. Create `Procfile`:

```
worker: node bot.js
```

#### 4. Deploy:

```
bash
git init
heroku create your-matcha-bot
git add .
git commit -m "Initial commit"
git push heroku main
```

### Option 2: Railway (Recommended)

1. Connect GitHub repo to Railway
2. Add environment variables
3. Deploy automatically

### Option 3: VPS (DigitalOcean, Linode)

1. Set up Ubuntu server
2. Install Node.js and PM2
3. Clone repository
4. Run with PM2:

```
bash
pm2 start bot.js --name "matcha-bot"
pm2 startup
pm2 save
```

## Step 6: Channel Structure

Your server will automatically get these channels:

- `#marukyu-koyamaen` - Marukyu Koyamaen alerts
- `#ippodo-global` - Ippodo Global alerts
- `#ippodo-us` - Ippodo US alerts
- `#kanbayashi-shunsho` - Kanbayashi Shunsho alerts
- `#horii-shichimeien` - Horii Shichimeien alerts
- `#yamamasa-koyamaen` - Yamamasa Koyamaen alerts
- `#mizuba-tea` - Mizuba Tea Co alerts
- `#gion-tsujiri` - Gion Tsujiri alerts
- `#rocky-matcha` - Rocky's Matcha alerts

## Step 7: User Experience

### For Users:

1. Run `!matcha-setup` to see vendor buttons
2. Click vendor buttons to subscribe/unsubscribe
3. Get pinged in vendor channels when restocks happen
4. Click "View Product" link to purchase

### Alert Format:

 Wakatake is back in stock! @Marukyu Koyamaen Alerts

 Price: ¥1,720

 Stock:  Available

 Size: Check Website

 Type: Matcha

 Link: View Product

Matcha Bell | 2025-08-20 03:31 PM (EST)

## Step 8: Monitoring & Maintenance

### Log Monitoring

```
bash
```

```
# Check logs
```

```
pm2 logs matcha-bot
```

```
# Monitor process
```

```
pm2 monit
```

### Update Products

Edit the `vendors` object in `bot.js` to add new products or vendors.

### Database Backup

The bot saves stock data to `stock_data.json`. Back this up regularly.

## Troubleshooting

### Common Issues:

**Bot not responding:**

- Check token is correct
- Verify bot has proper permissions
- Check if bot is online in server member list

#### **No alerts sending:**

- Check if channels exist
- Verify roles are created
- Check console for scraping errors
- Some sites may block bots (need proxy/headers)

#### **Rate limiting:**

- Add delays between requests
- Use different User-Agent strings
- Consider using proxies for high-volume monitoring

#### **Support Channels:**

- `#bot-support` - For technical issues
- `#feature-requests` - For new vendor requests
- `#general-matcha` - General discussion


#### **Security Notes**

- Keep bot token secret
- Don't commit tokens to git
- Use environment variables in production
- Monitor for API rate limits
- Respect website terms of service

#### **Legal Considerations**

- Web scraping should respect robots.txt
- Don't overload vendor servers
- This is for personal/community use only
- No commercial reselling of alert data

---

 **Happy matcha hunting!** Your community will love getting instant notifications for their favorite ceremonial grade matcha restocks.