**Membuat Game Bot Auto Pilot untuk Rohan 2 & LordNine dengan Python 3.14**

Baik, saya akan membantu Anda membuat aplikasi game bot yang dapat melakukan auto pilot untuk game Rohan 2 dan LordNine. Ini adalah panduan lengkap step-by-step dengan bash code via PowerShell.

**Overview Project**

Game bot ini akan memiliki kemampuan:

* **Auto combat** - Otomatis menyerang monster
* **Auto healing** - Otomatis menggunakan potion
* **Auto looting** - Otomatis pickup item
* **Auto skill rotation** - Menggunakan skill secara otomatis
* **Screen detection** - Mendeteksi kondisi game melalui screen
* **Path finding** - Navigasi otomatis
* **GUI Control Panel** - Interface modern untuk mengontrol bot

**Step 1: Setup Project Environment**

# Buat folder project  
New-Item -Path "C:\Projects\GameBotAutoRohan" -ItemType Directory -Force  
cd C:\Projects\GameBotAutoRohan  
  
# Verify lokasi  
pwd  
  
# Create virtual environment  
python -m venv venv  
  
# Aktivasi virtual environment  
.\venv\Scripts\Activate.ps1  
  
# Jika error ExecutionPolicy:  
Set-ExecutionPolicy -ExecutionPolicy RemoteSigned -Scope CurrentUser  
.\venv\Scripts\Activate.ps1  
  
# Upgrade pip  
python -m pip install --upgrade pip

**Step 2: Install Dependencies untuk Game Bot**

# Core dependencies untuk bot automation  
pip install pyautogui # Untuk mouse & keyboard control  
pip install pillow # Untuk image processing  
pip install opencv-python # Untuk computer vision & screen detection  
pip install numpy # Untuk array operations  
pip install pytesseract # Untuk OCR (membaca text di screen)  
pip install keyboard # Untuk hotkey detection  
pip install mouse # Untuk mouse operations  
pip install mss # Untuk fast screenshot  
pip install pywin32 # Untuk Windows API access  
  
# GUI Framework - CustomTkinter (simple & modern)  
pip install customtkinter  
  
# Utilities  
pip install pyyaml # Untuk configuration files  
pip install loguru # Untuk logging  
pip install colorama # Untuk colored console output  
  
# Verify instalasi  
pip list

**Step 3: Install Tesseract OCR (Required)**

# Download Tesseract OCR installer  
# URL: https://github.com/UB-Mannheim/tesseract/wiki  
  
# Jalankan command ini untuk download (requires curl)  
curl -L -o tesseract-installer.exe https://digi.bib.uni-mannheim.de/tesseract/tesseract-ocr-w64-setup-5.3.3.20231005.exe  
  
# Atau download manual dari browser, kemudian install  
  
# Set Tesseract path (sesuaikan dengan lokasi instalasi)  
[Environment]::SetEnvironmentVariable("TESSERACT\_PATH", "C:\Program Files\Tesseract-OCR\tesseract.exe", "User")  
  
# Verify Tesseract  
tesseract --version

**Step 4: Create Project Structure**

# Buat struktur folder  
New-Item -Path "bot" -ItemType Directory  
New-Item -Path "bot\core" -ItemType Directory  
New-Item -Path "bot\detection" -ItemType Directory  
New-Item -Path "bot\actions" -ItemType Directory  
New-Item -Path "bot\gui" -ItemType Directory  
New-Item -Path "bot\utils" -ItemType Directory  
New-Item -Path "config" -ItemType Directory  
New-Item -Path "resources" -ItemType Directory  
New-Item -Path "resources\images" -ItemType Directory  
New-Item -Path "resources\templates" -ItemType Directory  
New-Item -Path "logs" -ItemType Directory  
New-Item -Path "screenshots" -ItemType Directory  
  
# Buat \_\_init\_\_.py files  
New-Item -Path "bot\\_\_init\_\_.py" -ItemType File  
New-Item -Path "bot\core\\_\_init\_\_.py" -ItemType File  
New-Item -Path "bot\detection\\_\_init\_\_.py" -ItemType File  
New-Item -Path "bot\actions\\_\_init\_\_.py" -ItemType File  
New-Item -Path "bot\gui\\_\_init\_\_.py" -ItemType File  
New-Item -Path "bot\utils\\_\_init\_\_.py" -ItemType File  
  
# Verify struktur  
tree /F

**Step 5: Create Configuration File**

# Buat config.yaml  
@"  
# Game Bot Configuration for Rohan 2 & LordNine  
  
bot\_settings:  
 game: "rohan2" # or "lordnine"  
 auto\_combat: true  
 auto\_heal: true  
 auto\_loot: true  
 auto\_skill: true  
   
hotkeys:  
 start\_bot: "F9"  
 stop\_bot: "F10"  
 emergency\_stop: "F12"  
   
combat:  
 attack\_key: "1"  
 skill\_1: "2"  
 skill\_2: "3"  
 skill\_3: "4"  
 skill\_4: "5"  
 skill\_rotation\_delay: 0.5  
   
healing:  
 hp\_potion\_key: "F1"  
 mp\_potion\_key: "F2"  
 hp\_threshold: 50 # Percentage  
 mp\_threshold: 30 # Percentage  
   
looting:  
 loot\_key: "z"  
 loot\_delay: 0.3  
 loot\_range: 100 # pixels  
   
detection:  
 monster\_color: [255, 0, 0] # RGB color for monster detection  
 hp\_bar\_position: [50, 50, 150, 70] # x1, y1, x2, y2  
 mp\_bar\_position: [50, 80, 150, 100]  
   
safety:  
 idle\_timeout: 300 # seconds  
 random\_movement: true  
 anti\_detection: true  
"@ | Out-File -FilePath "config\config.yaml" -Encoding utf8  
  
Write-Host "Config file created!" -ForegroundColor Green

**Step 6: Create Core Bot Engine**

# Buat bot\core\engine.py  
@"  
import time  
import threading  
from loguru import logger  
import keyboard  
import pyautogui  
  
class BotEngine:  
 '''Core bot engine untuk game automation'''  
   
 def \_\_init\_\_(self, config):  
 self.config = config  
 self.running = False  
 self.paused = False  
 self.combat\_thread = None  
 self.heal\_thread = None  
   
 # Setup pyautogui  
 pyautogui.FAILSAFE = True  
 pyautogui.PAUSE = 0.1  
   
 logger.info('Bot Engine initialized')  
   
 def start(self):  
 '''Start bot'''  
 if self.running:  
 logger.warning('Bot already running')  
 return  
   
 self.running = True  
 logger.info('Bot started')  
   
 # Start threads  
 self.combat\_thread = threading.Thread(target=self.\_combat\_loop, daemon=True)  
 self.heal\_thread = threading.Thread(target=self.\_heal\_loop, daemon=True)  
   
 self.combat\_thread.start()  
 self.heal\_thread.start()  
   
 def stop(self):  
 '''Stop bot'''  
 self.running = False  
 logger.info('Bot stopped')  
   
 def pause(self):  
 '''Pause bot'''  
 self.paused = not self.paused  
 status = 'paused' if self.paused else 'resumed'  
 logger.info(f'Bot {status}')  
   
 def \_combat\_loop(self):  
 '''Main combat loop'''  
 while self.running:  
 if self.paused:  
 time.sleep(0.5)  
 continue  
   
 try:  
 # Auto combat logic  
 self.\_attack()  
 time.sleep(0.5)  
 except Exception as e:  
 logger.error(f'Combat error: {e}')  
   
 def \_heal\_loop(self):  
 '''Healing loop'''  
 while self.running:  
 if self.paused:  
 time.sleep(0.5)  
 continue  
   
 try:  
 # Auto heal logic  
 self.\_check\_health()  
 time.sleep(1)  
 except Exception as e:  
 logger.error(f'Heal error: {e}')  
   
 def \_attack(self):  
 '''Attack logic'''  
 attack\_key = self.config['combat']['attack\_key']  
 pyautogui.press(attack\_key)  
   
 def \_check\_health(self):  
 '''Check and heal if needed'''  
 # Placeholder - akan diimplementasi dengan screen detection  
 pass  
"@ | Out-File -FilePath "bot\core\engine.py" -Encoding utf8  
  
Write-Host "Bot engine created!" -ForegroundColor Green

**Step 7: Create Screen Detection Module**

# Buat bot\detection\screen.py  
@"  
import cv2  
import numpy as np  
import mss  
import pytesseract  
from PIL import Image  
from loguru import logger  
  
class ScreenDetector:  
 '''Screen detection untuk game analysis'''  
   
 def \_\_init\_\_(self, config):  
 self.config = config  
 self.sct = mss.mss()  
   
 # Set tesseract path  
 pytesseract.pytesseract.tesseract\_cmd = r'C:\Program Files\Tesseract-OCR\tesseract.exe'  
   
 logger.info('Screen Detector initialized')  
   
 def capture\_screen(self, region=None):  
 '''Capture screenshot'''  
 if region:  
 monitor = {  
 'top': region[1],  
 'left': region[0],  
 'width': region[2] - region[0],  
 'height': region[3] - region[1]  
 }  
 else:  
 monitor = self.sct.monitors[1]  
   
 screenshot = self.sct.grab(monitor)  
 img = np.array(screenshot)  
 return cv2.cvtColor(img, cv2.COLOR\_BGRA2BGR)  
   
 def detect\_hp\_percentage(self):  
 '''Detect HP bar percentage'''  
 hp\_region = self.config['detection']['hp\_bar\_position']  
 screenshot = self.capture\_screen(hp\_region)  
   
 # Convert to HSV for color detection  
 hsv = cv2.cvtColor(screenshot, cv2.COLOR\_BGR2HSV)  
   
 # Detect red color (HP bar)  
 lower\_red = np.array([0, 100, 100])  
 upper\_red = np.array([10, 255, 255])  
 mask = cv2.inRange(hsv, lower\_red, upper\_red)  
   
 # Calculate percentage  
 total\_pixels = mask.shape[0] \* mask.shape[1]  
 red\_pixels = np.sum(mask > 0)  
 percentage = (red\_pixels / total\_pixels) \* 100  
   
 return percentage  
   
 def detect\_mp\_percentage(self):  
 '''Detect MP bar percentage'''  
 mp\_region = self.config['detection']['mp\_bar\_position']  
 screenshot = self.capture\_screen(mp\_region)  
   
 # Convert to HSV  
 hsv = cv2.cvtColor(screenshot, cv2.COLOR\_BGR2HSV)  
   
 # Detect blue color (MP bar)  
 lower\_blue = np.array([100, 100, 100])  
 upper\_blue = np.array([130, 255, 255])  
 mask = cv2.inRange(hsv, lower\_blue, upper\_blue)  
   
 # Calculate percentage  
 total\_pixels = mask.shape[0] \* mask.shape[1]  
 blue\_pixels = np.sum(mask > 0)  
 percentage = (blue\_pixels / total\_pixels) \* 100  
   
 return percentage  
   
 def detect\_monster(self):  
 '''Detect monster on screen'''  
 screenshot = self.capture\_screen()  
   
 # Convert to HSV  
 hsv = cv2.cvtColor(screenshot, cv2.COLOR\_BGR2HSV)  
   
 # Detect monster color  
 monster\_color = self.config['detection']['monster\_color']  
 lower = np.array([max(0, monster\_color[0]-10), 100, 100])  
 upper = np.array([min(255, monster\_color[0]+10), 255, 255])  
   
 mask = cv2.inRange(hsv, lower, upper)  
   
 # Find contours  
 contours, \_ = cv2.findContours(mask, cv2.RETR\_EXTERNAL, cv2.CHAIN\_APPROX\_SIMPLE)  
   
 if contours:  
 # Get largest contour  
 largest = max(contours, key=cv2.contourArea)  
 x, y, w, h = cv2.boundingRect(largest)  
 return True, (x + w//2, y + h//2)  
   
 return False, None  
   
 def find\_loot(self):  
 '''Find loot items on screen'''  
 screenshot = self.capture\_screen()  
   
 # Detect bright/gold colored items  
 hsv = cv2.cvtColor(screenshot, cv2.COLOR\_BGR2HSV)  
   
 lower\_gold = np.array([20, 100, 100])  
 upper\_gold = np.array([30, 255, 255])  
 mask = cv2.inRange(hsv, lower\_gold, upper\_gold)  
   
 contours, \_ = cv2.findContours(mask, cv2.RETR\_EXTERNAL, cv2.CHAIN\_APPROX\_SIMPLE)  
   
 loot\_positions = []  
 for contour in contours:  
 if cv2.contourArea(contour) > 100: # Filter small noise  
 x, y, w, h = cv2.boundingRect(contour)  
 loot\_positions.append((x + w//2, y + h//2))  
   
 return loot\_positions  
"@ | Out-File -FilePath "bot\detection\screen.py" -Encoding utf8  
  
Write-Host "Screen detector created!" -ForegroundColor Green

**Step 8: Create Action Module**

# Buat bot\actions\player.py  
@"  
import pyautogui  
import time  
import random  
from loguru import logger  
  
class PlayerActions:  
 '''Player actions untuk game control'''  
   
 def \_\_init\_\_(self, config):  
 self.config = config  
 self.last\_skill\_time = {}  
   
 def attack(self):  
 '''Perform basic attack'''  
 attack\_key = self.config['combat']['attack\_key']  
 pyautogui.press(attack\_key)  
 logger.debug('Attack executed')  
   
 def use\_skill(self, skill\_number):  
 '''Use skill'''  
 skill\_key = self.config['combat'].get(f'skill\_{skill\_number}')  
 if skill\_key:  
 pyautogui.press(skill\_key)  
 self.last\_skill\_time[skill\_number] = time.time()  
 logger.debug(f'Skill {skill\_number} used')  
   
 def skill\_rotation(self):  
 '''Execute skill rotation'''  
 delay = self.config['combat']['skill\_rotation\_delay']  
   
 # Check cooldowns and use skills  
 for i in range(1, 5):  
 last\_use = self.last\_skill\_time.get(i, 0)  
 if time.time() - last\_use > 5: # 5 second cooldown  
 self.use\_skill(i)  
 time.sleep(delay)  
   
 def heal\_hp(self):  
 '''Use HP potion'''  
 hp\_key = self.config['healing']['hp\_potion\_key']  
 pyautogui.press(hp\_key)  
 logger.info('HP potion used')  
   
 def heal\_mp(self):  
 '''Use MP potion'''  
 mp\_key = self.config['healing']['mp\_potion\_key']  
 pyautogui.press(mp\_key)  
 logger.info('MP potion used')  
   
 def loot\_item(self):  
 '''Pickup loot'''  
 loot\_key = self.config['looting']['loot\_key']  
 pyautogui.press(loot\_key)  
 time.sleep(self.config['looting']['loot\_delay'])  
 logger.debug('Loot picked up')  
   
 def move\_to(self, x, y):  
 '''Move character to position'''  
 pyautogui.click(x, y)  
 logger.debug(f'Moving to ({x}, {y})')  
   
 def random\_movement(self):  
 '''Random movement untuk anti-detection'''  
 screen\_width, screen\_height = pyautogui.size()  
   
 # Random position near center  
 center\_x, center\_y = screen\_width // 2, screen\_height // 2  
 offset = 100  
   
 rand\_x = center\_x + random.randint(-offset, offset)  
 rand\_y = center\_y + random.randint(-offset, offset)  
   
 self.move\_to(rand\_x, rand\_y)  
 logger.debug('Random movement executed')  
"@ | Out-File -FilePath "bot\actions\player.py" -Encoding utf8  
  
Write-Host "Player actions created!" -ForegroundColor Green

**Step 9: Create Complete Bot Logic**

# Buat bot\core\bot.py  
@"  
import time  
import yaml  
import keyboard  
from loguru import logger  
from bot.detection.screen import ScreenDetector  
from bot.actions.player import PlayerActions  
  
class GameBot:  
 '''Main game bot controller'''  
   
 def \_\_init\_\_(self, config\_path='config/config.yaml'):  
 # Load config  
 with open(config\_path, 'r') as f:  
 self.config = yaml.safe\_load(f)  
   
 self.detector = ScreenDetector(self.config)  
 self.player = PlayerActions(self.config)  
   
 self.running = False  
 self.paused = False  
   
 # Setup hotkeys  
 self.\_setup\_hotkeys()  
   
 logger.info('Game Bot initialized')  
   
 def \_setup\_hotkeys(self):  
 '''Setup keyboard hotkeys'''  
 hotkeys = self.config['hotkeys']  
   
 keyboard.add\_hotkey(hotkeys['start\_bot'], self.start)  
 keyboard.add\_hotkey(hotkeys['stop\_bot'], self.stop)  
 keyboard.add\_hotkey(hotkeys['emergency\_stop'], self.emergency\_stop)  
   
 logger.info('Hotkeys registered')  
   
 def start(self):  
 '''Start bot'''  
 if self.running:  
 logger.warning('Bot already running')  
 return  
   
 self.running = True  
 self.paused = False  
 logger.info('🚀 Bot started!')  
   
 self.main\_loop()  
   
 def stop(self):  
 '''Stop bot'''  
 self.running = False  
 logger.info('⏹️ Bot stopped')  
   
 def emergency\_stop(self):  
 '''Emergency stop'''  
 self.running = False  
 logger.critical('🛑 EMERGENCY STOP!')  
   
 def main\_loop(self):  
 '''Main bot loop'''  
 idle\_counter = 0  
 max\_idle = self.config['safety']['idle\_timeout']  
   
 while self.running:  
 try:  
 # Check HP  
 if self.config['bot\_settings']['auto\_heal']:  
 hp = self.detector.detect\_hp\_percentage()  
 hp\_threshold = self.config['healing']['hp\_threshold']  
   
 if hp < hp\_threshold:  
 self.player.heal\_hp()  
 time.sleep(1)  
   
 # Check MP  
 if self.config['bot\_settings']['auto\_heal']:  
 mp = self.detector.detect\_mp\_percentage()  
 mp\_threshold = self.config['healing']['mp\_threshold']  
   
 if mp < mp\_threshold:  
 self.player.heal\_mp()  
 time.sleep(1)  
   
 # Detect and attack monster  
 if self.config['bot\_settings']['auto\_combat']:  
 found\_monster, monster\_pos = self.detector.detect\_monster()  
   
 if found\_monster:  
 # Move to monster  
 self.player.move\_to(\*monster\_pos)  
 time.sleep(0.5)  
   
 # Attack  
 self.player.attack()  
   
 # Use skills  
 if self.config['bot\_settings']['auto\_skill']:  
 self.player.skill\_rotation()  
   
 idle\_counter = 0  
 else:  
 idle\_counter += 1  
   
 # Auto loot  
 if self.config['bot\_settings']['auto\_loot']:  
 loot\_positions = self.detector.find\_loot()  
   
 if loot\_positions:  
 for loot\_pos in loot\_positions[:3]: # Max 3 items  
 self.player.move\_to(\*loot\_pos)  
 time.sleep(0.3)  
 self.player.loot\_item()  
   
 # Anti-detection: random movement  
 if self.config['safety']['random\_movement']:  
 if idle\_counter > 10:  
 self.player.random\_movement()  
 idle\_counter = 0  
   
 # Safety: stop if idle too long  
 if idle\_counter > max\_idle:  
 logger.warning('Idle timeout reached, stopping bot')  
 self.stop()  
   
 time.sleep(0.5)  
   
 except Exception as e:  
 logger.error(f'Error in main loop: {e}')  
 time.sleep(1)  
   
 logger.info('Bot loop ended')  
"@ | Out-File -FilePath "bot\core\bot.py" -Encoding utf8  
  
Write-Host "Complete bot logic created!" -ForegroundColor Green

**Step 10: Create GUI Control Panel**

# Buat bot\gui\control\_panel.py  
@"  
import customtkinter as ctk  
import threading  
from loguru import logger  
from bot.core.bot import GameBot  
  
ctk.set\_appearance\_mode('dark')  
ctk.set\_default\_color\_theme('blue')  
  
class BotControlPanel(ctk.CTk):  
 '''GUI Control Panel untuk Game Bot'''  
   
 def \_\_init\_\_(self):  
 super().\_\_init\_\_()  
   
 self.title('🎮 Rohan 2 & LordNine Game Bot')  
 self.geometry('800x600')  
   
 self.bot = None  
 self.bot\_thread = None  
   
 self.setup\_ui()  
   
 def setup\_ui(self):  
 # Header  
 header = ctk.CTkLabel(  
 self,  
 text='🎮 Game Bot Control Panel',  
 font=ctk.CTkFont(size=28, weight='bold')  
 )  
 header.pack(pady=20)  
   
 # Game selection  
 game\_frame = ctk.CTkFrame(self)  
 game\_frame.pack(fill='x', padx=20, pady=10)  
   
 ctk.CTkLabel(  
 game\_frame,  
 text='Select Game:',  
 font=ctk.CTkFont(size=14)  
 ).pack(side='left', padx=10)  
   
 self.game\_selector = ctk.CTkSegmentedButton(  
 game\_frame,  
 values=['Rohan 2', 'LordNine']  
 )  
 self.game\_selector.pack(side='left', padx=10)  
 self.game\_selector.set('Rohan 2')  
   
 # Features frame  
 features\_frame = ctk.CTkFrame(self)  
 features\_frame.pack(fill='both', expand=True, padx=20, pady=10)  
   
 ctk.CTkLabel(  
 features\_frame,  
 text='Bot Features',  
 font=ctk.CTkFont(size=18, weight='bold')  
 ).pack(pady=10)  
   
 # Feature toggles  
 self.auto\_combat = ctk.CTkSwitch(  
 features\_frame,  
 text='🗡️ Auto Combat',  
 font=ctk.CTkFont(size=14)  
 )  
 self.auto\_combat.pack(pady=5)  
 self.auto\_combat.select()  
   
 self.auto\_heal = ctk.CTkSwitch(  
 features\_frame,  
 text='💊 Auto Healing',  
 font=ctk.CTkFont(size=14)  
 )  
 self.auto\_heal.pack(pady=5)  
 self.auto\_heal.select()  
   
 self.auto\_loot = ctk.CTkSwitch(  
 features\_frame,  
 text='💰 Auto Looting',  
 font=ctk.CTkFont(size=14)  
 )  
 self.auto\_loot.pack(pady=5)  
 self.auto\_loot.select()  
   
 self.auto\_skill = ctk.CTkSwitch(  
 features\_frame,  
 text='⚡ Auto Skill Rotation',  
 font=ctk.CTkFont(size=14)  
 )  
 self.auto\_skill.pack(pady=5)  
 self.auto\_skill.select()  
   
 # Status  
 self.status\_label = ctk.CTkLabel(  
 features\_frame,  
 text='Status: Stopped',  
 font=ctk.CTkFont(size=16),  
 text\_color='red'  
 )  
 self.status\_label.pack(pady=20)  
   
 # Control buttons  
 button\_frame = ctk.CTkFrame(self)  
 button\_frame.pack(fill='x', padx=20, pady=20)  
   
 self.start\_btn = ctk.CTkButton(  
 button\_frame,  
 text='▶️ Start Bot',  
 command=self.start\_bot,  
 font=ctk.CTkFont(size=16, weight='bold'),  
 height=50,  
 fg\_color='green',  
 hover\_color='darkgreen'  
 )  
 self.start\_btn.pack(side='left', expand=True, padx=5)  
   
 self.stop\_btn = ctk.CTkButton(  
 button\_frame,  
 text='⏹️ Stop Bot',  
 command=self.stop\_bot,  
 font=ctk.CTkFont(size=16, weight='bold'),  
 height=50,  
 fg\_color='red',  
 hover\_color='darkred',  
 state='disabled'  
 )  
 self.stop\_btn.pack(side='left', expand=True, padx=5)  
   
 # Hotkey info  
 info\_frame = ctk.CTkFrame(self)  
 info\_frame.pack(fill='x', padx=20, pady=10)  
   
 info\_text = '''  
 ⌨️ Hotkeys:  
 F9 - Start Bot | F10 - Stop Bot | F12 - Emergency Stop  
 '''  
   
 ctk.CTkLabel(  
 info\_frame,  
 text=info\_text,  
 font=ctk.CTkFont(size=12)  
 ).pack(pady=10)  
   
 def start\_bot(self):  
 '''Start the bot'''  
 try:  
 # Initialize bot  
 self.bot = GameBot()  
   
 # Update config from GUI  
 self.bot.config['bot\_settings']['auto\_combat'] = self.auto\_combat.get()  
 self.bot.config['bot\_settings']['auto\_heal'] = self.auto\_heal.get()  
 self.bot.config['bot\_settings']['auto\_loot'] = self.auto\_loot.get()  
 self.bot.config['bot\_settings']['auto\_skill'] = self.auto\_skill.get()  
   
 # Start bot in thread  
 self.bot\_thread = threading.Thread(target=self.bot.start, daemon=True)  
 self.bot\_thread.start()  
   
 # Update UI  
 self.status\_label.configure(text='Status: Running', text\_color='green')  
 self.start\_btn.configure(state='disabled')  
 self.stop\_btn.configure(state='normal')  
   
 logger.info('Bot started from GUI')  
   
 except Exception as e:  
 logger.error(f'Failed to start bot: {e}')  
 self.status\_label.configure(text=f'Error: {e}', text\_color='red')  
   
 def stop\_bot(self):  
 '''Stop the bot'''  
 if self.bot:  
 self.bot.stop()  
   
 # Update UI  
 self.status\_label.configure(text='Status: Stopped', text\_color='red')  
 self.start\_btn.configure(state='normal')  
 self.stop\_btn.configure(state='disabled')  
   
 logger.info('Bot stopped from GUI')  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 app = BotControlPanel()  
 app.mainloop()  
"@ | Out-File -FilePath "bot\gui\control\_panel.py" -Encoding utf8  
  
Write-Host "GUI Control Panel created!" -ForegroundColor Green

**Step 11: Create Main Entry Point**

# Buat main.py  
@"  
#!/usr/bin/env python3  
'''  
Game Bot Auto Pilot for Rohan 2 & LordNine  
Python 3.14 | Simple & Detailed  
'''  
  
import sys  
from loguru import logger  
from bot.gui.control\_panel import BotControlPanel  
  
# Setup logging  
logger.remove()  
logger.add(  
 'logs/bot\_{time}.log',  
 rotation='1 day',  
 retention='7 days',  
 level='DEBUG',  
 format='{time:YYYY-MM-DD HH:mm:ss} | {level} | {message}'  
)  
logger.add(sys.stdout, level='INFO')  
  
def main():  
 '''Main entry point'''  
 logger.info('=' \* 60)  
 logger.info('Game Bot Auto Pilot Started')  
 logger.info('Python 3.14 | Rohan 2 & LordNine')  
 logger.info('=' \* 60)  
   
 try:  
 # Launch GUI  
 app = BotControlPanel()  
 app.mainloop()  
   
 except KeyboardInterrupt:  
 logger.info('Bot terminated by user')  
 except Exception as e:  
 logger.critical(f'Fatal error: {e}')  
 finally:  
 logger.info('Bot shutdown complete')  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 main()  
"@ | Out-File -FilePath "main.py" -Encoding utf8  
  
Write-Host "Main entry point created!" -ForegroundColor Green

**Step 12: Create Utility Scripts**

# Buat bot\utils\calibration.py untuk calibrate screen regions  
@"  
import cv2  
import mss  
import numpy as np  
from PIL import Image  
  
def calibrate\_hp\_bar():  
 '''Tool untuk calibrate HP bar position'''  
 print('Click and drag to select HP bar region')  
 print('Press ENTER when done, ESC to cancel')  
   
 with mss.mss() as sct:  
 monitor = sct.monitors[1]  
 screenshot = sct.grab(monitor)  
 img = np.array(screenshot)  
 img = cv2.cvtColor(img, cv2.COLOR\_BGRA2BGR)  
   
 # Select ROI  
 roi = cv2.selectROI('Select HP Bar', img, False)  
 cv2.destroyAllWindows()  
   
 if roi[2] > 0 and roi[3] > 0:  
 x1, y1, w, h = roi  
 x2, y2 = x1 + w, y1 + h  
 print(f'HP Bar Position: [{x1}, {y1}, {x2}, {y2}]')  
 print('Copy this to config.yaml under detection.hp\_bar\_position')  
 return [x1, y1, x2, y2]  
   
 return None  
  
def calibrate\_mp\_bar():  
 '''Tool untuk calibrate MP bar position'''  
 print('Click and drag to select MP bar region')  
 print('Press ENTER when done, ESC to cancel')  
   
 with mss.mss() as sct:  
 monitor = sct.monitors[1]  
 screenshot = sct.grab(monitor)  
 img = np.array(screenshot)  
 img = cv2.cvtColor(img, cv2.COLOR\_BGRA2BGR)  
   
 roi = cv2.selectROI('Select MP Bar', img, False)  
 cv2.destroyAllWindows()  
   
 if roi[2] > 0 and roi[3] > 0:  
 x1, y1, w, h = roi  
 x2, y2 = x1 + w, y1 + h  
 print(f'MP Bar Position: [{x1}, {y1}, {x2}, {y2}]')  
 print('Copy this to config.yaml under detection.mp\_bar\_position')  
 return [x1, y1, x2, y2]  
   
 return None  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 print('=== Screen Calibration Tool ===')  
 print('1. Calibrate HP Bar')  
 print('2. Calibrate MP Bar')  
 choice = input('Select option: ')  
   
 if choice == '1':  
 calibrate\_hp\_bar()  
 elif choice == '2':  
 calibrate\_mp\_bar()  
"@ | Out-File -FilePath "bot\utils\calibration.py" -Encoding utf8  
  
Write-Host "Calibration tool created!" -ForegroundColor Green

**Step 13: Create Requirements File**

# Generate requirements.txt  
pip freeze > requirements.txt  
  
# Display requirements  
Write-Host "`n=== Installed Packages ===" -ForegroundColor Cyan  
cat requirements.txt

**Step 14: Create Run Scripts**

# Buat run.ps1  
@"  
# Game Bot Launcher Script  
  
Write-Host "🎮 Starting Game Bot..." -ForegroundColor Green  
  
# Activate virtual environment  
.\venv\Scripts\Activate.ps1  
  
# Run bot  
python main.py  
"@ | Out-File -FilePath "run.ps1" -Encoding utf8  
  
# Buat calibrate.ps1  
@"  
# Screen Calibration Tool  
  
Write-Host "🎯 Starting Calibration Tool..." -ForegroundColor Yellow  
  
# Activate virtual environment  
.\venv\Scripts\Activate.ps1  
  
# Run calibration  
python bot\utils\calibration.py  
"@ | Out-File -FilePath "calibrate.ps1" -Encoding utf8  
  
Write-Host "Run scripts created!" -ForegroundColor Green

**Step 15: Test Run Application**

# Test import semua modules  
python -c "from bot.core.bot import GameBot; from bot.detection.screen import ScreenDetector; from bot.actions.player import PlayerActions; print('✅ All modules imported successfully')"  
  
# Run GUI  
python main.py

**Step 16: Calibration Setup (PENTING!)**

# Jalankan game terlebih dahulu (Rohan 2 atau LordNine)  
# Kemudian run calibration tool  
  
.\calibrate.ps1  
  
# Ikuti instruksi:  
# 1. Select HP bar region dengan mouse  
# 2. Press ENTER  
# 3. Copy coordinates ke config.yaml  
# 4. Repeat untuk MP bar

**Step 17: Build Executable**

# Install PyInstaller jika belum  
pip install pyinstaller  
  
# Build executable  
pyinstaller --name="GameBotRohan" `  
 --windowed `  
 --onefile `  
 --add-data "config;config" `  
 --add-data "resources;resources" `  
 --hidden-import="PIL.\_tkinter\_finder" `  
 main.py  
  
# Executable akan ada di: dist\GameBotRohan.exe  
  
Write-Host "✅ Executable created: dist\GameBotRohan.exe" -ForegroundColor Green

**Step 18: Create Complete Setup Script**

# Buat setup\_gamebot.ps1 - All-in-one installer  
@"  
# ============================================  
# Game Bot Auto Pilot - Complete Setup  
# For Rohan 2 & LordNine  
# Python 3.14  
# ============================================  
  
Write-Host "🎮 Game Bot Setup Starting..." -ForegroundColor Green  
Write-Host ""  
  
# Check Python  
Write-Host "Checking Python installation..." -ForegroundColor Yellow  
`$pythonVersion = python --version 2>&1  
if (`$LASTEXITCODE -ne 0) {  
 Write-Host "❌ Python not found! Please install Python 3.14" -ForegroundColor Red  
 exit 1  
}  
Write-Host "✅ `$pythonVersion" -ForegroundColor Green  
  
# Create project  
`$projectPath = "C:\Projects\GameBotAutoRohan"  
Write-Host "Creating project at `$projectPath..." -ForegroundColor Yellow  
New-Item -Path `$projectPath -ItemType Directory -Force | Out-Null  
Set-Location `$projectPath  
  
# Create venv  
Write-Host "Creating virtual environment..." -ForegroundColor Yellow  
python -m venv venv  
.\venv\Scripts\Activate.ps1  
  
# Upgrade pip  
Write-Host "Upgrading pip..." -ForegroundColor Yellow  
python -m pip install --upgrade pip | Out-Null  
  
# Install packages  
Write-Host "Installing bot dependencies..." -ForegroundColor Yellow  
pip install pyautogui pillow opencv-python numpy pytesseract keyboard mouse mss pywin32 customtkinter pyyaml loguru colorama pyinstaller | Out-Null  
  
# Create structure  
Write-Host "Creating project structure..." -ForegroundColor Yellow  
New-Item -Path "bot\core", "bot\detection", "bot\actions", "bot\gui", "bot\utils", "config", "resources\images", "resources\templates", "logs", "screenshots" -ItemType Directory -Force | Out-Null  
New-Item -Path "bot\\_\_init\_\_.py", "bot\core\\_\_init\_\_.py", "bot\detection\\_\_init\_\_.py", "bot\actions\\_\_init\_\_.py", "bot\gui\\_\_init\_\_.py", "bot\utils\\_\_init\_\_.py" -ItemType File -Force | Out-Null  
  
Write-Host ""  
Write-Host "✅ Setup Complete!" -ForegroundColor Green  
Write-Host ""  
Write-Host "Next steps:" -ForegroundColor Yellow  
Write-Host "1. Copy all Python files to their respective folders" -ForegroundColor White  
Write-Host "2. Run calibration: .\calibrate.ps1" -ForegroundColor White  
Write-Host "3. Start bot: .\run.ps1" -ForegroundColor White  
Write-Host ""  
Write-Host "Project location: `$projectPath" -ForegroundColor Cyan  
"@ | Out-File -FilePath "setup\_gamebot.ps1" -Encoding utf8  
  
Write-Host "Setup script created!" -ForegroundColor Green

**Quick Start Guide**

# 1. Run complete setup  
.\setup\_gamebot.ps1  
  
# 2. Ensure all code files are in place  
  
# 3. Open game (Rohan 2 or LordNine)  
  
# 4. Run calibration tool  
.\calibrate.ps1  
# - Select HP bar region  
# - Select MP bar region  
# - Update config.yaml with coordinates  
  
# 5. Start bot  
.\run.ps1  
  
# 6. In GUI:  
# - Select game (Rohan 2 or LordNine)  
# - Toggle desired features  
# - Click "Start Bot"

**Hotkeys Reference**

F9 - Start Bot  
F10 - Stop Bot  
F12 - Emergency Stop (instant)

**Configuration Tips**

Edit config\config.yaml untuk customize:

combat:  
 attack\_key: "1" # Change ke key attack Anda  
 skill\_1: "2" # Skill hotkey  
 skill\_2: "3"  
 skill\_3: "4"  
 skill\_4: "5"  
  
healing:  
 hp\_potion\_key: "F1" # HP potion hotkey  
 mp\_potion\_key: "F2" # MP potion hotkey  
 hp\_threshold: 50 # Heal saat HP < 50%  
 mp\_threshold: 30 # Heal saat MP < 30%  
  
looting:  
 loot\_key: "z" # Loot/pickup key

**Troubleshooting**

**Issue 1: Tesseract not found**

# Download dan install Tesseract OCR  
# https://github.com/UB-Mannheim/tesseract/wiki  
  
# Set environment variable  
[Environment]::SetEnvironmentVariable("TESSERACT\_PATH", "C:\Program Files\Tesseract-OCR\tesseract.exe", "User")

**Issue 2: Bot tidak detect HP/MP bar**

# Run calibration tool lagi  
.\calibrate.ps1  
  
# Pastikan coordinates di config.yaml benar

**Issue 3: Game tidak respond ke bot**

# Check game running as Administrator  
# Run bot juga as Administrator

**Advanced Features (Optional)**

**Add Telegram Notifications**

pip install python-telegram-bot  
  
# Add to config.yaml:  
# telegram:  
# bot\_token: "YOUR\_TOKEN"  
# chat\_id: "YOUR\_CHAT\_ID"

**Add Web Dashboard**

pip install flask flask-socketio  
  
# Access bot status via browser

**Project Summary**

Struktur akhir project:

GameBotAutoRohan/  
├── bot/  
│ ├── core/  
│ │ ├── bot.py # Main bot logic  
│ │ └── engine.py # Bot engine  
│ ├── detection/  
│ │ └── screen.py # Screen detection  
│ ├── actions/  
│ │ └── player.py # Player actions  
│ ├── gui/  
│ │ └── control\_panel.py # GUI interface  
│ └── utils/  
│ └── calibration.py # Calibration tool  
├── config/  
│ └── config.yaml # Configuration  
├── resources/  
├── logs/  
├── main.py # Entry point  
├── run.ps1 # Run script  
├── calibrate.ps1 # Calibration script  
└── requirements.txt

**Total Features:**

* ✅ Auto Combat dengan skill rotation
* ✅ Auto Healing (HP & MP)
* ✅ Auto Looting
* ✅ Screen Detection dengan OpenCV
* ✅ Modern GUI Control Panel
* ✅ Hotkey Support
* ✅ Anti-Detection (random movement)
* ✅ Safety Features (idle timeout)
* ✅ Logging System
* ✅ Calibration Tool
* ✅ Multi-game Support (Rohan 2 & LordNine)

Bot siap digunakan! 🎮🚀

1. <https://www.blog.pythonlibrary.org/2025/10/09/an-intro-to-python-3-14s-new-features/>

1. <https://docs.python.org/3/whatsnew/3.14.html>

1. <https://kinsta.com/blog/python-frameworks/>

1. <https://blog.protradingart.com/2024/07/pyside6-vs-pyqt-key-differences.html>

1. <https://www.bairesdev.com/blog/best-python-gui-libraries/>

1. <https://www.pythonguis.com/faq/pyqt6-vs-pyside6/>

1. <https://realpython.com/python-pyqt-layout/>

1. <https://www.pythonguis.com/pyqt6-tutorial/>

1. <https://customtkinter.tomschimansky.com>

1. <https://github.com/TomSchimansky/CustomTkinter>

1. <https://python-hub.com/intro-customtkinter-modern-looking-guis-in-python/>

1. <https://www.pythonguis.com/faq/which-python-gui-library/>

1. <https://flet.dev>

1. <https://dev.to/imrrobot/flet-build-real-time-web-desktop-and-mobile-apps-in-python-2nlk>

1. <https://www.geeksforgeeks.org/python/building-flutter-apps-in-python/>

1. <https://flet.dev/docs/cookbook/packaging-desktop-app/>

1. <https://flet.dev/docs/publish/>

1. <https://www.geeksforgeeks.org/android/difference-between-mvc-mvp-and-mvvm-architecture-pattern-in-android/>

1. <https://stackoverflow.com/questions/14214664/mvc-pattern-in-desktop-gui-with-python>

1. <https://www.appventurez.com/blog/difference-between-mvc-mvp-and-mvvm-architecture>

1. <https://www.technource.com/blog/best-python-gui-frameworks-to-build-ai-applications/>

1. <https://fullscale.io/blog/python-gui-frameworks/>

1. <https://www.pythonguis.com/tutorials/packaging-tkinter-applications-windows-pyinstaller/>

1. <https://www.pythonguis.com/tutorials/packaging-pyqt6-applications-windows-pyinstaller/>

1. <https://www.pyinstaller.org>

1. <https://stackoverflow.com/questions/73886669/how-to-distribute-a-python-application-professionaly>

1. <https://shakuro.com/blog/mvc-vs-mvvm>

1. <https://www.geeksforgeeks.org/python/building-desktop-applications-in-python/>

1. <https://www.pythonguis.com/pyqt6/>

1. <https://www.pythonguis.com/pyside6-tutorial/>

1. <https://blog.martinfitzpatrick.com/pyqt6-pyside6-books-updated-2025/>

1. <https://realpython.com/python-pyqt-gui-calculator/>

1. <https://github.com/saikotek/python-design-patterns>

1. <https://refactoring.guru/design-patterns/python>

1. <https://www.youtube.com/watch?v=SPQxHt_ovu0>

1. <https://www.geeksforgeeks.org/best-python-gui-frameworks-for-developers/>

1. <https://realpython.com/python314-new-features/>

1. <https://stepmediasoftware.com/blog/best-gui-library-for-python/>

1. <https://www.linkedin.com/posts/realpython-com_which-python-gui-library-should-you-use-in-activity-7309657710931644416-VujJ>

1. <https://www.wedowebapps.com/python-gui-libraries/>

1. <https://www.reddit.com/r/learnpython/comments/1b3can3/pyside6_or_pyqt6/>

1. <https://www.reddit.com/r/Python/comments/oauw9w/which_python_framework_is_used_by_professional_to/>

1. <https://www.youtube.com/watch?v=JK-B-CT34EU>

1. <https://kivy.org>

1. <https://python.plainenglish.io/i-ditched-tkinter-these-5-python-gui-libraries-made-coding-uis-fun-again-052d74ba3c67>

1. <https://towardsdatascience.com/modern-gui-applications-for-computer-vision-in-python/>

1. <https://stackoverflow.com/questions/44455257/how-to-create-beautiful-uis-with-python>

1. <https://www.youtube.com/watch?v=1GTnhhywXz8>

1. <https://dev.to/yash_makan/4-ways-to-create-modern-gui-in-python-in-easiest-way-possible-5e0e>

1. <https://python.plainenglish.io/no-more-clunky-interfaces-8-clean-python-gui-options-for-2025-f9342aa8062e>

1. <https://pub.dev/packages/flet>

1. <https://www.youtube.com/watch?v=eHhXoCNCI1c>

1. <https://www.linkedin.com/posts/sina-riyahi_mvc-mvp-mvi-mvvm-and-viper-patterns-activity-7376914732613730304-WOQr>

1. <https://mvysny.github.io/mvc-mvp-mvvm-no-thanks/>

1. <https://www.geeksforgeeks.org/blogs/best-python-gui-frameworks-for-developers/>

1. <https://jurnal.stkippgritulungagung.ac.id/index.php/jipi/article/download/3293/1469>

1. <https://dev.to/fadygrab/build-a-gui-and-package-your-killer-python-scripts-with-tkinter-and-pyinstaller-4afl>

1. <https://www.youtube.com/watch?v=nJQh1L9Y70U>

1. <https://doc.qt.io/qtforpython-6/deployment/deployment-pyinstaller.html>

1. <https://serverascode.com/2024/12/21/python-installation-methods.html>

1. <https://stackoverflow.com/questions/5888870/how-do-i-compile-a-pyqt-script-py-to-a-single-standalone-executable-file-for>

1. <https://github.com/topics/pyinstaller-gui>

1. <https://discuss.python.org/t/python-desktop-application-distribution/26314>

1. <https://www.youtube.com/watch?v=U4zlCx-emzc>

1. <https://stackoverflow.com/questions/79678141/build-python-project-with-custom-packages-using-pyinstaller>

1. <https://wiki.python.org/moin/PyQt/Deploying_PyQt_Applications>

1. <https://realpython.com/pyinstaller-python/>

1. <https://code-b.dev/blog/building-desktop-applications-using-python>

1. <https://doc.qt.io/qtforpython-6/deployment/index.html>

1. <https://python.plainenglish.io/building-desktop-applications-with-python-and-tkinter-0d709ee4700a>