Fiche Projet : TermuxToolkit

Ce document résume les composants techniques et fonctionnels du projet TermuxToolkit développé par Zoubirou Mohammed Ilyes.

Ce projet permet une gestion intelligente des commandes sous Termux avec interface CLI et Web.

[main.py]

```
from app import app

if __name__ == "__main__":
    app.run(host="0.0.0.0", port=5000, debug=True)
```

[app.py]

```
import os
import logging
from datetime import datetime
from flask import Flask, render_template, request, jsonify, session
from utils import detect_commands, execute_command, get_detected_commands, get_custom_commands

# Configure logging
logging.basicConfig(level=logging.DEBUG)

# Initialize Flask app
app = Flask(__name__)
app.secret_key = os.environ.get("SESSION_SECRET", "dev_secret_key")

# File paths for storing data
OPTIONS_FILE = "termux_custom_options.txt"
DETECTED_COMMANDS = "termux_detected_commands.txt"
LOG_FILE = "termux_log.txt"

# Initialize files if they don't exist
for file_path in [OPTIONS_FILE, DETECTED_COMMANDS, LOG_FILE]:
    if not os.path.exists(file_path):
        with open(file_path, 'w') as f:
            pass

@app.route('/')
def index():
    """Render the main application page."""
    return render_template('index.html')

@app.route('/detect_commands', methods=['POST'])
def detect_ubuntu_commands():
    """Detect Ubuntu commands and save them to file."""
    try:
        detect_commands(DETECTED_COMMANDS)
        return jsonify({
            'status': 'success',
            'message': ' Commandes dtectes et enregistres.'
```

```
        })
    except Exception as e:
        logging.error(f"Error detecting commands: {str(e)}")
        return jsonify({
            'status': 'error',
            'message': f' Erreur lors de la dtection des commandes: {str(e)}'
        })

@app.route('/get_commands', methods=['GET'])
def get_commands():
    """Get detected
...
```

[utils.py]

```
import os
import subprocess
import logging
from typing import List, Tuple

def detect_commands(output_file: str) -> None:
    """
    Detect Ubuntu commands from /usr/bin and save to file.

    Args:
        output_file: Path to file where commands will be saved
    """
    try:
        # Get all files in /usr/bin
        result = subprocess.run(
            ["ls", "/usr/bin"],
            capture_output=True,
            text=True,
            check=True
        )

        # Write the output to file
        with open(output_file, 'w') as f:
            f.write(result.stdout)

    except subprocess.CalledProcessError as e:
        logging.error(f"Command detection failed: {e}")
        raise Exception(f"chec de la dtection des commandes: {e}")
    except Exception as e:
        logging.error(f"Error in detect_commands: {e}")
        raise

def execute_command(command: str) -> str:
    """
    Execute a shell command safely and return its output.
```

```
    Args:
        command: The command to execute

    Returns:
        The command output (stdout and stderr)
    """
    try:
        # Execute the command and capture output
        process = subprocess.run(
            command,
            shell=True,  # Use shell for command parsing
            capture_output=True,
            text=True,
            timeout=60  # Timeout after 60 seconds
        )

        # Combine stdout and stderr
        output = process.stdout
        if proc
...
```

[index.html]

```
{% extends "layout.html" %}

{% block content %}
<div class="row">
    <!-- Menu Panel -->
    <div class="col-lg-3 mb-4">
        <div class="card">
            <div class="card-header bg-primary text-white">
                <h2 class="h5 mb-0"> Menu Principal</h2>
            </div>
            <div class="card-body">
                <div class="list-group">
                    <button id="detect-commands" class="list-group-item list-group-item-action">
                        1 Dtecter les commandes Ubuntu
                    </button>
                    <button id="show-commands" class="list-group-item list-group-item-action">
                        2 Afficher toutes les options disponibles
                    </button>
                    <button id="select-execute" class="list-group-item list-group-item-action">
                        3 Slectionner et excuter une commande
                    </button>
                    <button id="manual-execute" class="list-group-item list-group-item-action">
                        4 Saisir manuellement une commande
                    </button>
                    <button id="add-command" class="list-group-item list-group-item-action">
                        5 Ajouter une commande personnalise
```

```
                    </button>
                    <button id="delete-command" class="list-group-item list-group-item-action">
                        6 Supprimer une commande personnalise
                    </button>

...
```

[termux_manager.sh]

```
#!/bin/bash

OPTIONS_FILE="termux_custom_options.txt"
DETECTED_COMMANDS="termux_detected_commands.txt"
LOG_FILE="termux_log.txt"

[ ! -f "$OPTIONS_FILE" ] && touch "$OPTIONS_FILE"
[ ! -f "$DETECTED_COMMANDS" ] && touch "$DETECTED_COMMANDS"
[ ! -f "$LOG_FILE" ] && touch "$LOG_FILE"

LAST_RESULT=""

afficher_cadre() {
    echo -e "\n"
    echo -e " $1"
    echo -e ""
}

detecter_commandes() {
    afficher_cadre " Dtection des commandes Ubuntu"
    ls /usr/bin > "$DETECTED_COMMANDS"
    LAST_RESULT=" Commandes dtectes et enregistres."
}

afficher_options() {
    afficher_cadre " Liste des commandes disponibles"
    LAST_RESULT=" Commandes dtectes :\n$(cat -n "$DETECTED_COMMANDS")\n"
    if [ -s "$OPTIONS_FILE" ]; then
        LAST_RESULT+=" Commandes personnalises :\n$(cat -n "$OPTIONS_FILE")\n"
    else
        LAST_RESULT+=" Aucune option personnalise.\n"
    fi
}

executer_par_selection() {
    afficher_options
    read -p "Entrez le numro de la commande  excuter : " num_option
    option=$(sed -n "${num_option}p" "$OPTIONS_FILE")
    if [ -z "$option" ]; then
        option=$(sed -n "${num_option}p" "$DETECTED_COMMANDS")
```

Fiche Projet : TermuxToolkit

```
    fi
    if [ -n "$option" ]; then
        read -p " Confirmer l'excution de '$option' ? (o/n) : " confirmation
        if [[ "$confirmation" == "o" || "$confirmation" == "O" ]]; then
            LAST_RESULT=" Excution de : $option\n$(eval "$option" 2>&1)"
            echo -
...
```