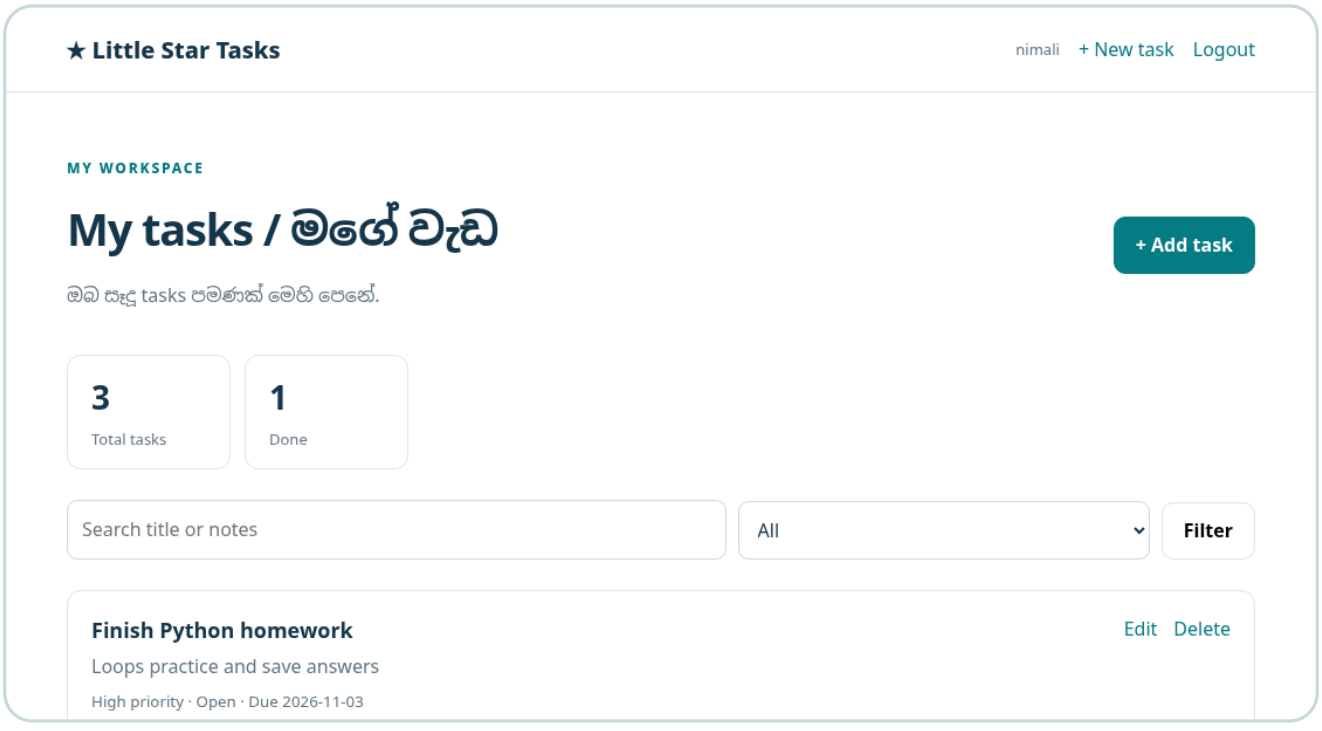
FROM ZERO TO YOUR FIRST DJANGO APP

Python Django Login + CRUD මුල සිටම ඉගෙනගනිමු

Beginner Project · Little Star Task Manager

Python basics → Django setup → Signup / Login → SQLite → Create, Read, Update, Delete → Security → Tests → Git. සැබෑවටම සෑදිය හැකි උදාහරණයක් සමඟ.



100% WHITE PAGES | REAL SOURCE CODE | VS CODE + CHROME

මේ පොත ඉගෙනගන්නේ කොහොමද?

මෙය Python ඉගෙනගන්න පටන්ගත් කෙනෙකුටත් පියවරෙන් පියවර අනුගමනය කළ හැකි Sinhala workbook එකක්. එක් පාඩමකට අරමුණක්, ක්‍රියා පියවර, code සහ අභ්‍යාසයක් අඩංගුයි.

01

පාඩම කියවන්න

02

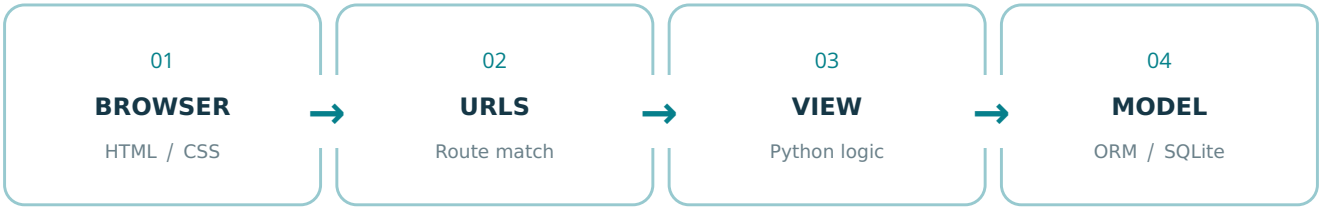
Code එක type කරන්න

03

Browser එකෙන් පරීක්ෂා කරන්න

04

Expected result එක ගළපන්න



ආරම්භ කරන්න පෙර

Computer එකක Windows, Python 3.12+ (හෝ supported newer version), VS Code සහ browser එකක් තියෙනවාද බලන්න. ZIP එකේ sample Django project එක ඇතුළත්.

මෙහි Chrome-style pages කියන්නේ source HTML/CSS මත පදනම් වූ අධ්‍යාපනික previews. Django server එක ඇත්තටම run කරන විට ප්‍රතිඵල ඔබම පරීක්ෂා කරන්න.

අන්තර්ගතය

කුඩා පියවරෙන් පටන්ගෙන database එකට data save කරන සම්පූර්ණ web application එකක් දක්වා.

00 පළමු ගමන — 6 පාඩම්

01 Python Basics — 9 පාඩම්

02 Django Setup — 8 පාඩම්

03 Project Architecture — 8 පාඩම්

04 Database & Models — 7 පාඩම්

05 Registration & Login — 10 පාඩම්

06 CRUD Tasks — 9 පාඩම්

07 Search & UI — 6 පාඩම්

08 Security & Testing — 9 පාඩම්

09 Git & Deployment — 6 පාඩම්

10 Practice & Revision — 6 පාඩම්

අමතර සම්පත්

Source ZIP: Python files, Django templates, CSS, migrations, automated tests, SRS, QA checklist, Security/Deployment notes සහ Git instructions.

PROJECT MAP

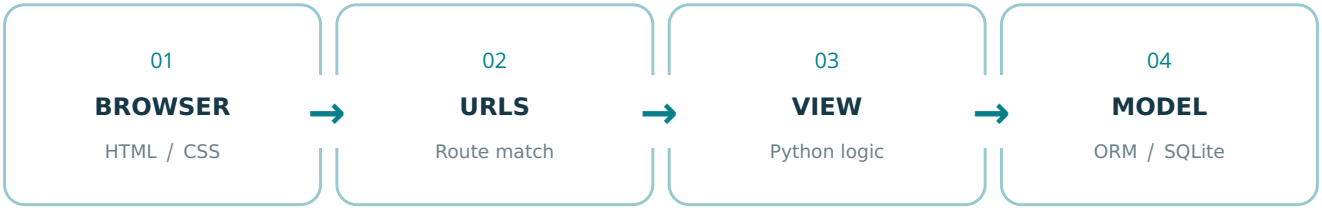
00

පළමු ගමන

මේ කොටස තුළ ඉතා සරල උදාහරණ, සත්‍ය source code සහ ඔබටම අත්හදා බැලිය හැකි අභ්‍යාස තිබෙනවා.

මේ කොටසේ පාඩම්

- ✓ අද අපි මොනවා හදනවාද?
- ✓ Django කියන්නේ මොකක්ද?
- ✓ CRUD යන අකුරු 4
- ✓ Browser එකෙන් Database එකට
- ✓ අපේ Project එකට තියෙන පිටු
- ✓ වෙබ් app එකේ නීති



00 • පළමු ගමන

LESSON 01 / 84

අද අපි මොනවා හදනවාද?

Little Star Tasks කියන්නේ ඔබේ දවසේ වැඩ ලියා තබාගන්න වෙබ් යෙදුමක්. එක් අයෙක් login වුණාම එයාගේ වැඩ පමණක් පෙන්වීම මේ project එකේ විශේෂම දෙයයි.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. මුලින් account එකක් හදන්න.
2. Login වී අලුත් task එකක් ඇතුළත් කරන්න.
3. Task එක බලන්න, වෙනස් කරන්න, මකා දමන්න.



පෙන්වන දර්ශනය අධ්‍යාපනික UI preview එකකි. Actual result එක local browser එකේ බලන්න.

පුංචි අභ්‍යාසය · TRY IT

ඔබේ දවසේ කළ යුතු වැඩ 3ක් කඩදාසියක ලියා තබන්න.

> **Expected result:** ඒ වැඩ 3ම පසුව Django app එකට ඇතුළත් කරනවා.

00 • පළමු ගමන

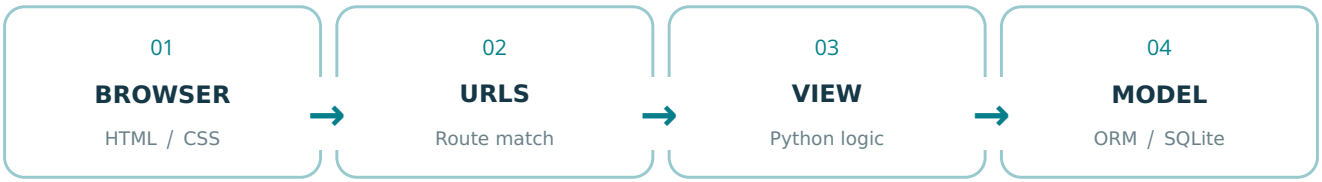
LESSON 02 / 84

Django කියන්නේ මොකක්ද?

Python භාෂාවෙන් වෙබ් යෙදුම් ඉක්මනින් සෑදීමට Django framework එක උදව් කරනවා. Login, database, forms සහ admin වගේ වැදගත් දේවල් සඳහා සූදානම් කොටස් එහි තිබෙනවා.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. Python = programming language.
2. Django = Python භාවිතා කරන web framework.
3. අපේ app = Django tools එකට එකතු කළ project එක.



පුංචි අභ්‍යාසය · TRY IT

Python සහ Django යන වචන දෙකට ඔබේම වාක්‍ය දෙකක් ලියන්න.

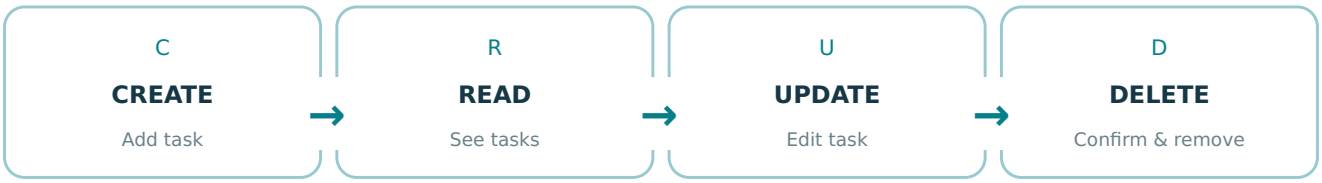
> **Expected result:** Programming language එකක් සහ framework එකක් අතර වෙනස පැහැදිලි වීම.

CRUD යන අකුරු 4

CRUD කියන්නේ database එකක තොරතුරු කළමනාකරණය කරන ප්‍රධාන ක්‍රියා හතරයි. Task Manager project එකේ සෑම ක්‍රියාවකටම button හෝ page එකක් තිබෙනවා.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. C = Create → අලුත් task එකක්.
2. R = Read → tasks ලැයිස්තුව බලන්න.
3. U = Update → task එක වෙනස් කරන්න.
4. D = Delete → අවශ්‍ය නැති task එක මකා දමන්න.



පුංචි අභ්‍යාසය · TRY IT

කඩදාසියේ C/R/U/D වෙන වෙනම ලියා ඒකට ගැළපෙන ක්‍රියා සඳහන් කරන්න.

> **Expected result:** Create, Read, Update, Delete පිළිවෙළ මතක තබාගැනීම.

00 • පළමු ගමන

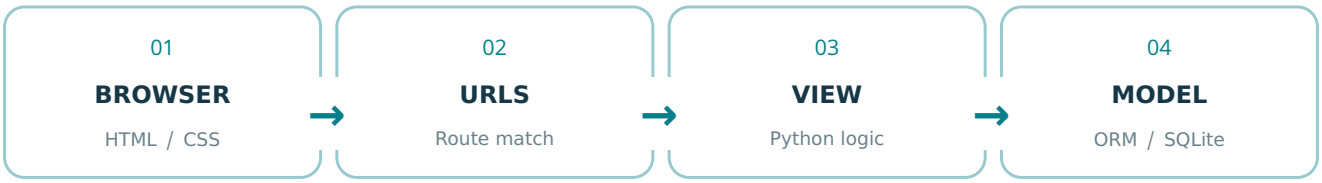
LESSON 04 / 84

Browser එකෙන් Database එකට

Browser එකේ අපි button එකක් click කරනවා. ඒ request එක Django වෙත යනවා. Django අවශ්‍ය data database එකෙන් අරගෙන HTML පිටුවක් browser එකට යවනවා.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. Browser → URL එකක් ඉල්ලයි.
2. Django view → ඉල්ලීම හසුරවයි.
3. Model → SQLite database එක සමඟ වැඩ කරයි.
4. Template → පෙන්විය යුතු HTML හදයි.



පුංචි අභ්‍යාසය · TRY IT

ඊතල භාවිතයෙන් Browser → View → Model → Database සිතියම අඳින්න.

> **Expected result:** Frontend, backend, database කාර්යයන් වෙන් කිරීම.

00 • පළමු ගමන

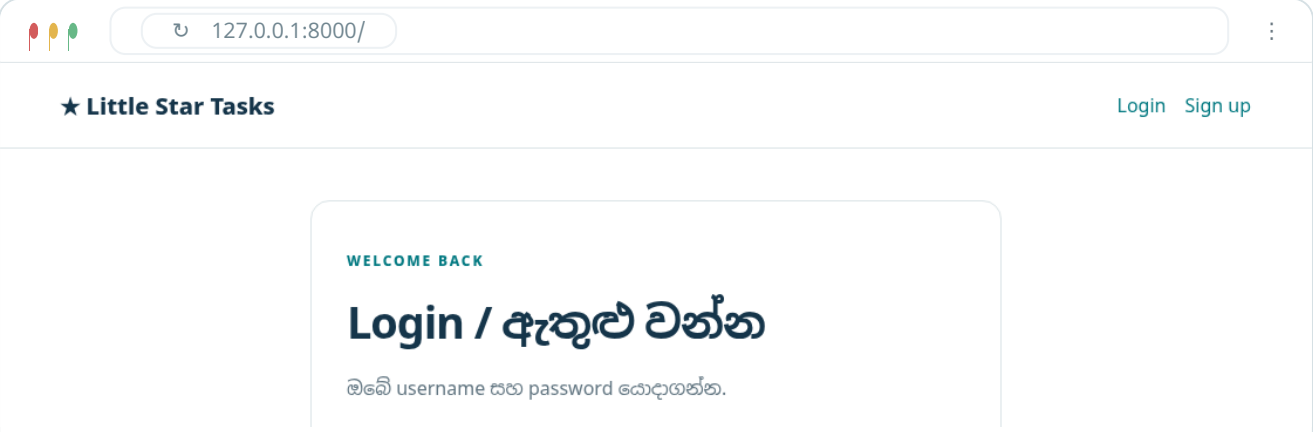
LESSON 05 / 84

අපේ Project එකට තියෙන පිටු

මෙහි Signup page, Login page, My Tasks dashboard, New/Edit form සහ Delete confirmation page ඇතුළත් වෙනවා. සෑම page එකකටම වෙනම URL එකක් තිබෙනවා.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. Signup: /accounts/signup/
2. Login: /accounts/login/
3. Dashboard: /
4. New task: /tasks/new/



පෙන්වන දර්ශනය අධ්‍යාපනික UI preview එකකි. Actual result එක local browser එකේ බලන්න.

පුංචි අභ්‍යාසය · TRY IT

Login button එක සහ Add Task button එක ඔබේම UI sketch එකක ඇඳ බලන්න.

> **Expected result:** Page flow එක ඔබේ හිතේ පැහැදිලි වීම.

00 • පළමු ගමන

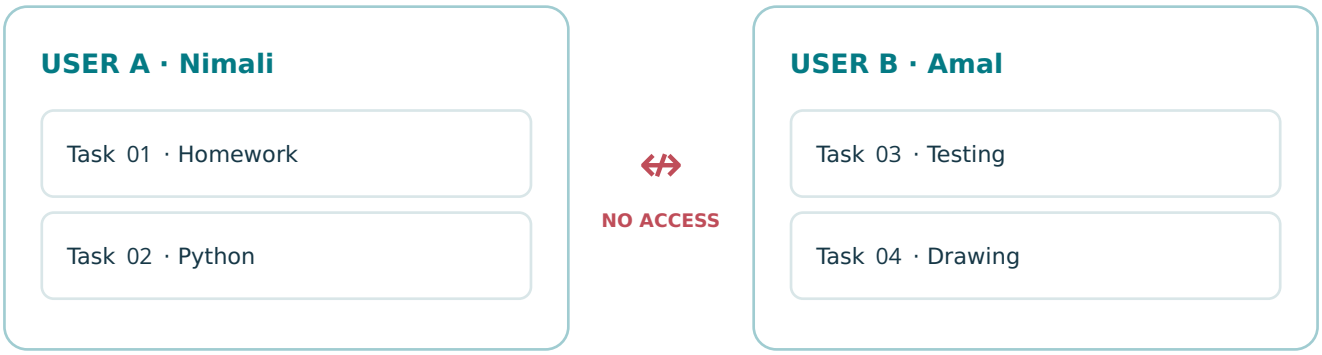
LESSON 06 / 84

වෙබ් app එකේ නීති

User කෙනෙකුගේ task තවත් කෙනෙකුට පෙන්වීම හෝ වෙනස් කිරීමට ඉඩදීම හොඳ නැහැ. ඒ නිසා login කිරීම පමණක් නොව database query එකේ owner පරීක්ෂාවත් අවශ්‍යයි.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. Login නැති අය → login page වෙත.
2. Login ඇති අය → තමන්ගේ dashboard එකට.
3. වෙනත් userගේ task ID දුන්නොත් → 404.



පුංචි අභ්‍යාසය · TRY IT

Nimali සහ Amal යන user දෙදෙනාගේ data වෙනම තීරු දෙකකින් අඳින්න.

> **Expected result:** “Authentication” සහ “Authorization” වෙනස හඳුනාගැනීම.

PYTHON FIRST

01

Python Basics

මේ කොටස තුළ ඉතා සරල උදාහරණ, සත්‍ය source code සහ ඔබටම අත්හදා බැලිය හැකි අභ්‍යාස තිබෙනවා.

මේ කොටසේ පාඩම්

- ✓ Python ආරම්භය
- ✓ Variable එකක් කියන්නේ?
- ✓ String, Integer, Boolean
- ✓ if / else තීරණ
- ✓ List සහ for loop
- ✓ Dictionary = නාම සහ අගයන්
- ✓ Function කියන්නේ නැවත යොදන වැඩක්
- ✓ Class සහ Object අදහස
- ✓ Import: වෙනත් code එක භාවිතා කිරීම

Command Prompt · Administrator privileges not required

```
C:\learn\django-little-star> python manage.py runserver
Watching for file changes...
Starting development server at
http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```

Python ආරම්භය

Python කියන්නේ මිනිසුන්ට කියවීමට පහසු programming language එකක්. Django එකත් Python භාවිතා කරන නිසා අපි මුලින්ම print, variables සහ functions හඳුනාගනිමු.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. VS Code විවෘත කරන්න.
2. hello.py ගොනුවක් හදන්න.
3. Terminal එකේ py hello.py දාන්න.

VS CODE / hello.py

```
print("Hello, Django!")
print("I can build a website")
```

Command Prompt · Administrator privileges not required

```
C:\learn\django-little-star> python manage.py runserver
Watching for file changes...
Starting development server at
http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```

පුංචි අභ්‍යාසය · TRY IT

Hello යන වචනය වෙනුවට ඔබේ නම යොදන්න.

> **Expected result:** Terminal එකේ වාක්‍ය දෙකක් පෙනිය යුතුයි.

Variable එකක් කියන්නේ?

Variable එකක් කියන්නේ අගයක් තබාගන්න නමක් ඇති පෙට්ටියක් වගේ. Username, task title, count වගේ values ඊට තබන්න පුළුවන්.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. username නමැති variable එක හදන්න.
2. task_count ලෙස අංකයක් තබන්න.
3. print() එකෙන් ඒවා පෙන්වන්න.

VS CODE / python_basics.py

```
username = "Nimali"
task_count = 3
print(username)
print(task_count)
```

පුංචි අභ්‍යාසය · TRY IT

task_count අගය 5 කර නැවත run කරන්න.

> **Expected result:** Nimali සහ 5 කියලා පෙන්නනවා.

String, Integer, Boolean

Task title එක text එකක්; task ID එක number එකක්; completed status එක True/False value එකක්. මේවාට data types කියනවා.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. str = text.
2. int = පූර්ණ අංක.
3. bool = True හෝ False.

VS CODE / python_basics.py

```
title = "Read a book"
task_id = 7
is_done = False
print(type(title))
print(type(task_id))
```

පුංචි අභ්‍යාසය · TRY IT

is_done = True කළ විට type එක වෙනස් වෙනවාද?

> **Expected result:** එය තවමත් bool type එකක්.

if / else තීරණ

Task එකක් ඉවරද නැද්ද කියලා පෙන්වීමට අපි condition භාවිතා කරනවා. Python indentation ඉතා වැදගත්.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. is_done boolean එක සකස් කරන්න.
2. if condition එක ලියන්න.
3. else කොටසට වෙනත් message එකක් දෙන්න.

VS CODE / python_basics.py

```
is_done = False
if is_done:
    print("Completed!")
else:
    print("Keep going!")
```

පුංචි අභ්‍යාසය · TRY IT

False → True කර output වෙනස් වන ආකාරය බලන්න.

> **Expected result:** Completed! ලෙස print වෙනවා.

List සහ for loop

Tasks 3ක් වෙන වෙනම variables 3ක තබනවාට වඩා list එකක් ගන්න පුළුවන්. for loop එකෙන් list එකේ සෑම අංගයක්ම පරීක්ෂා කළ හැකියි.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. tasks නමැති list එක හදන්න.
2. for loop එකෙන් title එකක් බැගින් print කරන්න.
3. ඔබේ task එකක් තවත් එකතු කරන්න.

VS CODE / python_basics.py

```
tasks = ["Study", "Draw", "Play"]
for task in tasks:
    print("-", task)
```

පුංචි අභ්‍යාසය · TRY IT

list එකට “Practice Django” යන්න එකතු කරන්න.

> **Expected result:** Output ලැයිස්තුවේ සිව්වැනි item එකත් පෙන්වයි.

Dictionary = නාම සහ අගයන්

Task එකක් title සහ status වැනි තොරතුරු සමඟ තබාගන්න dictionary එක උපකාරී වෙනවා. Database model එකට කලින් මේ අදහස තේරුම් ගන්න.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. title සහ done යන keys දෙක දාන්න.
2. task["title"] කියලා value එක ගන්න.
3. status වෙනස් කර බලන්න.

VS CODE / python_basics.py

```
task = {"title": "Study", "done": False}
print(task["title"])
task["done"] = True
print(task)
```

පුංචි අභ්‍යාසය · TRY IT

priority: high key එකක් එකතු කරන්න.

> **Expected result:** Dictionary එකේ values තුනක් පෙන්වයි.

Function කියන්නේ නැවත යොදන වැඩක්

නැවත නැවත එකම logic ලිවීම වෙනුවට function එකක් සෑදිය හැකියි. Django views ද Python functions ලෙස ලියන්න පුළුවන්.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. def යොදා function එකක් හදන්න.
2. parameter එකක් භාවිතා කරන්න.
3. return මඟින් value එක ලබා දෙන්න.

VS CODE / python_basics.py

```
def greeting(name):
    return "Hello " + name

print(greeting("Nimali"))
```

පුංචි අභ්‍යාසය · TRY IT

greeting("Amal") කියලා run කරන්න.

> **Expected result:** Hello Amal ලැබෙනවා.

Class සහ Object අදහස

Django models වලින් database records නියෝජනය කිරීමට classes භාවිතා වෙනවා. Class එක සැලැස්මක්, object එක ඒ සැලැස්මෙන් හදන සැබෑ උදාහරණයක්.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. class TaskDemo ලියන්න.
2. __init__ තුළ title තබන්න.
3. object එකක් සාදා .title කියවන්න.

VS CODE / python_basics.py

```
class TaskDemo:
    def __init__(self, title):
        self.title = title

first = TaskDemo("Homework")
print(first.title)
```

පුංචි අභ්‍යාසය · TRY IT

තවත් object එකකට “Test Django” title එකක් යොදන්න.

> **Expected result:** Objects දෙකට වෙන වෙනම titles තියෙනවා.

Import: වෙනත් code එක භාවිතා කිරීම

Python එකේ module එකක තියෙන function එකක් වෙනත් file එකකින් import කළ හැකියි. Django එකේ from django... import ... යන පේළි ගොඩක් ඔබට පෙනේවි.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. datetime module එක හඳුනාගන්න.
2. date.today() කියලා අද දිනය ගන්න.
3. Django imports කියවන්න සූදානම් වන්න.

VS CODE / python_basics.py

```
from datetime import date
print(date.today())

from pathlib import Path
print(Path.cwd())
```

පුංචි අභ්‍යාසය · TRY IT

ඔබේ project folder path එක output එකෙන් හඳුනාගන්න.

> **Expected result:** අද දිනය සහ folder path එක පෙන්වයි.

SETUP YOUR COMPUTER

02

Django Setup

මේ කොටස තුළ ඉතා සරල උදාහරණ, සත්‍ය source code සහ ඔබටම අත්හදා බැලිය හැකි අභ්‍යාස තිබෙනවා.

මේ කොටසේ පාඩම්

- ✓ 2026 සඳහා Python + Django version
- ✓ VS Code තුළ Project Folder
- ✓ Virtual Environment යනු කුමක්ද?
- ✓ Django install කිරීම
- ✓ Django Project ආරම්භ කරන්න
- ✓ Django App එකක් හදමු
- ✓ ඉක්මන් මාර්ගය: Source ZIP එක
- ✓ manage.py වැදගත් ඇයි?

EXPLORER · LITTLE-STAR-TASKS

```
▾ little_star_tasks/
      ├── manage.py
      ├── little_star_tasks/
      ├── tasks/
      ├── templates/
      ├── static/css/
      └── requirements.txt
```

2026 සඳහා Python + Django version

මෙම පොතේ උදාහරණ Django 6.1.2 සහ Python 3.12 හෝ ඊට අලුත් සහය ලැබෙන version සඳහා සකස් කළා. Django වෙබ් framework එකක් වුණත් ආරම්භයේ SQLite භාවිතයට වෙනම database server එකක් අවශ්‍ය නැහැ.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. Python 3.12+ ස්ථාපනය කරන්න.
2. CMD එකේ py --version පරීක්ෂා කරන්න.
3. පසුව Django version එකත් පරීක්ෂා කරන්න.

VS CODE / Windows CMD

```
py --version
py -m pip --version
```

Command Prompt · Administrator privileges not required

```
C:\learn\django-little-star> python manage.py runserver
Watching for file changes...
Starting development server at
http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```

පුංචි අභ්‍යාසය · TRY IT

ඔබේ Python version එක ලියන්න.

> **Expected result:** Python 3.12+ version number එක පෙනේවි.

VS Code තුළ Project Folder

අපි සියලුම Python files එකම project folder එකක් තුළ තබනවා. ඒ folder එක open කරලා Terminal → New Terminal යන්න.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. New folder: django-little-star.
2. VS Code > File > Open Folder.
3. Terminal > New Terminal.

VS CODE / Windows CMD

```
mkdir django-little-star
cd django-little-star
code .
```

EXPLORER · LITTLE-STAR-TASKS

```
▾ little_star_tasks/
      ├── manage.py
      ├── little_star_tasks/
      ├── tasks/
      ├── templates/
      ├── static/css/
      └── requirements.txt
```

පුංචි අභ්‍යාසය · TRY IT

VS Code Explorer එකේ හිස් folder එක පෙනේද බලන්න.

> **Expected result:** එකම folder එක project root එක ලෙස භාවිතා කරනවා.

Virtual Environment යනු කුමක්ද?

එක් project එකේ Python packages තවත් project එකක් සමඟ ගැටෙන්නේ නැතිව තබන්න .venv පරිසරය උපකාරී වෙනවා.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. CMD එකෙන් py -m venv .venv දෙන්න.
2. Windows හි activate script එක run කරන්න.
3. Prompt එකේ (.venv) පෙනේද බලන්න.

VS CODE / Windows CMD

```
py -m venv .venv
.venv\Scripts\activate
python --version
```

Command Prompt · Administrator privileges not required

```
C:\learn\django-little-star> python manage.py runserver
Watching for file changes...
Starting development server at
http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```

පුංචි අභ්‍යාසය · TRY IT

Prompt එකේ (.venv) ලකුණ තිබෙනවාද?

> **Expected result:** Virtual environment active.

Django install කිරීම

Django pip මඟින් ස්ථාපනය කරන්න. මෙම tutorial එක source code එකට ගැළපීමට version එක pin කර තිබෙනවා.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. Virtual environment active කරගන්න.
2. පහත install command එක run කරන්න.
3. Version එකත් check කරන්න.

VS CODE / Windows CMD

```
python -m pip install Django==6.1.2
python -m django --version
```

Command Prompt · Administrator privileges not required

```
C:\learn\django-little-star> python manage.py runserver
Watching for file changes...
Starting development server at
http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```

පුංචි අභ්‍යාසය · TRY IT

pip command error නම් python -m pip භාවිතා කරන්න.

> **Expected result:** Version 6.1.2 ලැබිය යුතුයි.

Django Project ආරම්භ කරන්න

Project යනු settings සහ ප්‍රධාන URL map රඳවාගන්නා කොටසයි. dot (.) එකෙන් files දැනටමත් විවෘත folder එකේම සෑදේ.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. අලුත් හිස් folder එකක් භාවිතා කරන්න.
2. django-admin startproject අණ දාන්න.
3. manage.py file එක ලැබුණාද බලන්න.

VS CODE / Windows CMD

```
django-admin startproject little_star_tasks .
python manage.py runserver
```

Command Prompt · Administrator privileges not required

```
C:\learn\django-little-star> python manage.py runserver
Watching for file changes...
Starting development server at
http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```

පුංචි අභ්‍යාසය · TRY IT

http://127.0.0.1:8000/ browser එකෙන් බලන්න.

> **Expected result:** Django default setup page එක පෙනෙයි.

Django App එකක් හදමු

Project එක තුළ task feature එක වෙනම “app” ලෙස වෙන් කරන්න. සාමාන්‍ය project එකකට apps ගණනාවක් තිබිය හැකියි.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. Server එක නවත්වන්න (Ctrl + C).
2. startapp tasks run කරන්න.
3. tasks folder එකේ files හඳුනාගන්න.

VS CODE / Windows CMD

```
python manage.py startapp tasks
```

EXPLORER · LITTLE-STAR-TASKS

```
▾ little_star_tasks/
      ├── manage.py
      ├── little_star_tasks/
      ├── tasks/
      ├── templates/
      ├── static/css/
      └── requirements.txt
```

පුංචි අභ්‍යාසය · TRY IT

models.py, views.py, apps.py files ගණන් කරන්න.

> **Expected result:** tasks folder එක project එක තුළ පෙනෙයි.

ඉක්මන් මාර්ගය: Source ZIP එක

මෙම PDF එකට දෙන source ZIP එකේ project files දැනටමත් සකස් කරලා තිබෙනවා. ZIP භාවිතා කරන්නේ නම් startproject/startapp නැවත කරන්න එපා.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. ZIP එක extract කරන්න.
2. manage.py ඇති folder එක CMD එකෙන් open කරන්න.
3. .venv activate කර requirements install කරන්න.

VS CODE / Windows CMD

```
py -m venv .venv
.venv\Scripts\activate
python -m pip install -r requirements.txt
```

EXPLORER · LITTLE-STAR-TASKS

```
▾ little_star_tasks/
      ├── manage.py
      ├── little_star_tasks/
      ├── tasks/
      ├── templates/
      ├── static/css/
      └── requirements.txt
```

පුංචි අභ්‍යාසය · TRY IT

ZIP මාර්ගය සහ from-scratch මාර්ගය එකට මිශ්‍ර නොකරන්න.

> **Expected result:** Folder එකේ manage.py සහ tasks/ තිබේ.

manage.py වැදගත් ඇයි?

manage.py හරහා Django project එකට අදාළ commands ක්‍රියාත්මක කරනවා. Server run, tests, migrations, admin user creation සියල්ලම මෙහිදී හැසිරවිය හැකියි.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. runserver = local server.
2. migrate = database tables.
3. test = automated tests.

VS CODE / Windows CMD

```
python manage.py runserver
python manage.py migrate
python manage.py test
```

පුංචි අභ්‍යාසය · TRY IT

ඒ commands තුනේ අර්ථය ඔබේම වචනවලින් ලියන්න.

> **Expected result:** සෑම command එකක්ම වෙනස් වැඩක් කරන බව තේරේ.

FILES THAT WORK TOGETHER

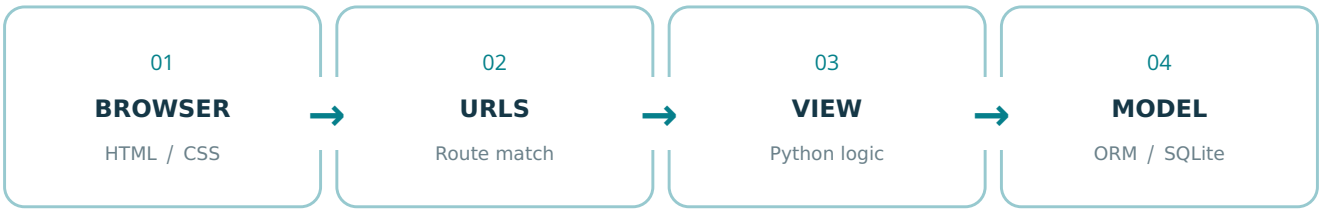
03

Project Architecture

මේ කොටස තුළ ඉතා සරල උදාහරණ, සත්‍ය source code සහ ඔබටම අත්හදා බැලිය හැකි අභ්‍යාස තිබෙනවා.

මේ කොටසේ පාඩම්

- ✓ Project Folder Structure
- ✓ Settings හි INSTALLED_APPS
- ✓ URL Routing කියන්නේ මොකක්ද?
- ✓ Project URLs සහ App URLs
- ✓ View Function එකක වැඩ
- ✓ Template එකක HTML
- ✓ Static CSS එක සම්බන්ධ කිරීම
- ✓ HTTP GET සහ POST



Project Folder Structure

Project root එකේ manage.py, settings module, tasks app, templates සහ static CSS තිබෙනවා. files වලට හොඳ නම් දීමෙන් errors සොයාගැනීම පහසු වෙනවා.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. manage.py = commands.
2. little_star_tasks/ = settings + root URLs.
3. tasks/ = application logic.
4. templates/ + static/ = UI.

VS CODE / VS Code / Explorer

```
little_star_tasks/
--- manage.py
--- little_star_tasks/
--- tasks/
--- templates/
--- static/css/
```

EXPLORER · LITTLE-STAR-TASKS

```
▾ little_star_tasks/
      ├── manage.py
      ├── little_star_tasks/
      ├── tasks/
      ├── templates/
      ├── static/css/
      └── requirements.txt
```

පුංචි අභ්‍යාසය · TRY IT

ZIP extract කළ folder එකේ මේ paths 5 සොයාගන්න.

> **Expected result:** Source files එකම tree එකකින් පෙනෙනවා.

Settings හි INSTALLED_APPS

Django ට tasks නමැති app එක project එකේ කොටසක් බව කියන්න INSTALLED_APPS ලැයිස්තුවට එය එකතු කරනවා.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. settings.py විවෘත කරන්න.
2. INSTALLED_APPS හොයාගන්න.
3. tasks.apps.TasksConfig පේළිය ඇතිද බලන්න.

VS CODE / little_star_tasks/settings.py

```
INSTALLED_APPS = [
    "django.contrib.admin",
    "django.contrib.auth",
    "django.contrib.contenttypes",
    "django.contrib.sessions",
    "django.contrib.messages",
    "django.contrib.staticfiles",
    "tasks.apps.TasksConfig",
]

MIDDLEWARE = [
    "django.middleware.security.SecurityMiddleware",
    "django.contrib.sessions.middleware.SessionMiddleware",
    "django.middleware.common.CommonMiddleware",
    "django.middleware.csrf.CsrfViewMiddleware",
    "django.contrib.auth.middleware.AuthenticationMiddleware",
    "django.contrib.messages.middleware.MessageMiddleware",
    "django.middleware.clickjacking.XFrameOptionsMiddleware",
]

ROOT_URLCONF = "little_star_tasks.urls"
TEMPLATES = [
    {
        "BACKEND": "django.template.backends.django.DjangoTemplates",
        "DIRS": [BASE_DIR / "templates"],
        "APP_DIRS": True,
        "OPTIONS": {
            "context_processors": [
                "django.template.context_processors.debug",
                "django.template.context_processors.request",
```

පුංචි අභ්‍යාසය · TRY IT

tasks entry එක ඉවත් කළොත් migrations ක්‍රියාත්මක නොවන්නේ ඇයි කියලා සිතන්න.

> **Expected result:** Django tasks app එක හඳුනාගන්නවා.

03 • Project Architecture

LESSON 26 / 84

URL Routing කියන්නේ මොකක්ද?

Browser එක /tasks/new/ ලෙස ඉල්ලූ විට යා යුතු view function එක Django URL map මඟින් තීරණය කරනවා.

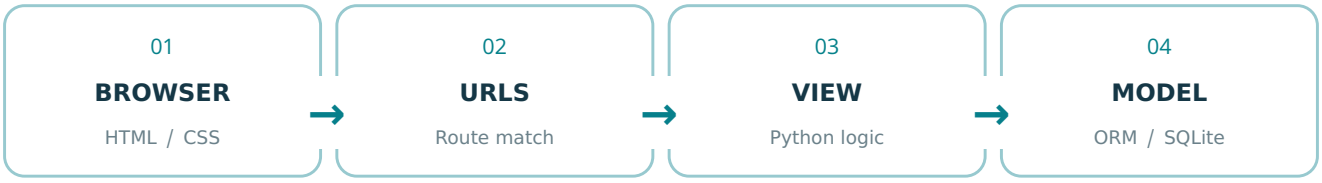
පියවරෙන් පියවර · FOLLOW THESE STEPS

1. URL path එක request එකට ගළපන්න.
2. එයට අදාළ view එක run වෙයි.
3. view එක HTML response එකක් ලබා දෙයි.

VS CODE / tasks/urls.py

```
"""URL names keep templates independent of hard-coded paths."""
from django.urls import path
from . import views

urlpatterns = [
    path("", views.task_list, name="task_list"),
    path("tasks/new/", views.task_create, name="task_create"),
    path("tasks/<int:pk>/edit/", views.task_update, name="task_update"),
    path("tasks/<int:pk>/delete/", views.task_delete, name="task_delete"),
]
```



පුංචි අභ්‍යාසය · TRY IT

task_create සඳහා URL එක හොයාගන්න.

> **Expected result:** /tasks/new/ → task_create.

Project URLs සහ App URLs

Root urls.py ගොනුවේ admin, login, signup, logout සහ tasks app link එක ඇත. tasks/urls.py තුළ CRUD URLs තබා තිබෙනවා.

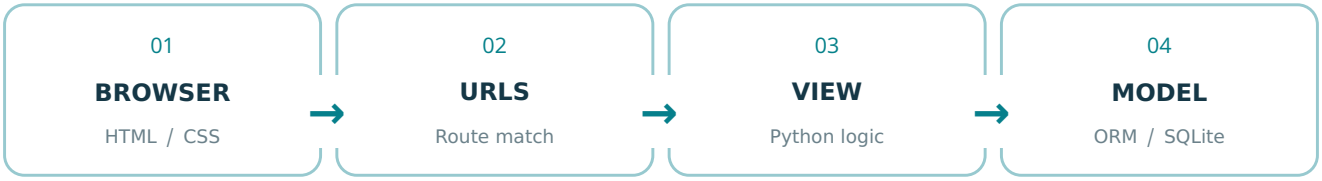
පියවරෙන් පියවර · FOLLOW THESE STEPS

1. project urls.py විවෘත කරන්න.
2. include("tasks.urls") කියන එක හඳුනාගන්න.
3. App URLs වෙනම file එකක ඇති හේතුව තේරුම් ගන්න.

VS CODE / little_star_tasks/urls.py

```python
"""Project-level URLs."""
from django.contrib import admin
from django.contrib.auth import views as auth_views
from django.urls import include, path
from tasks.views import signup

urlpatterns = [
    path("admin/", admin.site.urls),
    path("accounts/signup/", signup, name="signup"),
    path("accounts/login/", auth_views.LoginView.as_view(
        template_name="registration/login.html"
    ), name="login"),
    # Django 6.1's LogoutView logs out on POST only.
    path("accounts/logout/", auth_views.LogoutView.as_view(), name="logout"),
    path("", include("tasks.urls")),
]
```



පුංචි අභ්‍යාසය · TRY IT

Logout URL එක හඳුනාගන්න.

Expected result: /accounts/logout/ → LogoutView.

View Function එකක වැඩ

View එක request එක ලබාගෙන database හෝ form logic ක්‍රියාත්මක කර response එක තීරණය කරන Python function එකක්.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. request එක browser එකෙන් එනවා.
2. render() HTML template එකක් පෙන්වනවා.
3. redirect() තවත් URL එකකට යවනවා.

VS CODE / Python / concept

```python
from django.shortcuts import render

def example_view(request):
    return render(request, "tasks/list.html")
```

පුංචි අභ්‍යාසය · TRY IT

render() සහ redirect() අතර වෙනස කියන්න.

> **Expected result:** render = HTML; redirect = අලුත් URL ඉල්ලීම.

Template එකක HTML

Django templates HTML code එකට data එකතු කරයි. {{ value }} මඟින් value එක පෙන්වයි; {% for %} වැනි tags මඟින් iteration සිදු වෙයි.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. base.html layout එක නිර්මාණය කරන්න.
2. ළමා page එක extends භාවිතා කරයි.
3. block content එකෙන් වෙනස් කොටස පුරවයි.

VS CODE / templates/tasks/list.html

```
{% extends 'base.html' %}
{% block content %}
<h1>My tasks</h1>
{% endblock %}
```

පුංචි අභ්‍යාසය · TRY IT

Django template tags කිහිපයක් හඳුනාගන්න.

> **Expected result:** HTML එකේ පොදු layout එක නැවත නොලියා භාවිතා කළ හැකියි.

Static CSS එක සම්බන්ධ කිරීම

HTML එකේ වර්ණ, spacing, borders සහ mobile layout පාලනය කිරීමට CSS භාවිතා කරනවා. Django static tag එක file path ලබා දෙයි.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. static/css/site.css තැන සොයාගන්න.
2. base.html තුළ load static බලන්න.
3. Browser එක refresh කර CSS එක පෙනෙනවාද බලන්න.

VS CODE / templates/base.html

```
{% load static %}
<link rel="stylesheet"
      href="{% static 'css/site.css' %}">
```

පුංචි අභ්‍යාසය · TRY IT

site.css එකේ --teal variable එක වෙනස් කර local preview බලන්න.

> **Expected result:** Site accent color වෙනස් වේ.

HTTP GET සහ POST

GET request එක දත්ත බලන්න යොදාගන්නවා; POST request එක දත්ත වෙනස් කිරීමට බහුලව යොදාගන්නවා. Delete එකක් GET මඟින් කිරීම නරක practice එකක්.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. GET / → tasks බලන්න.
2. GET /tasks/new/ → form එක පෙන්වන්න.
3. POST /tasks/new/ → form data save කරන්න.

VS CODE / Python / concept

```python
if request.method == "POST":
    # Handle submitted form
    pass
else:
    # Display a blank form
    pass
```

පුංචි අභ්‍යාසය · TRY IT

Delete confirmation සඳහා GET භාවිතා කළත් මකාදැමීම POST වන හේතුව කියන්න.

> **Expected result:** GET එකක් අවදානම් data change එකක් නොකරයි.

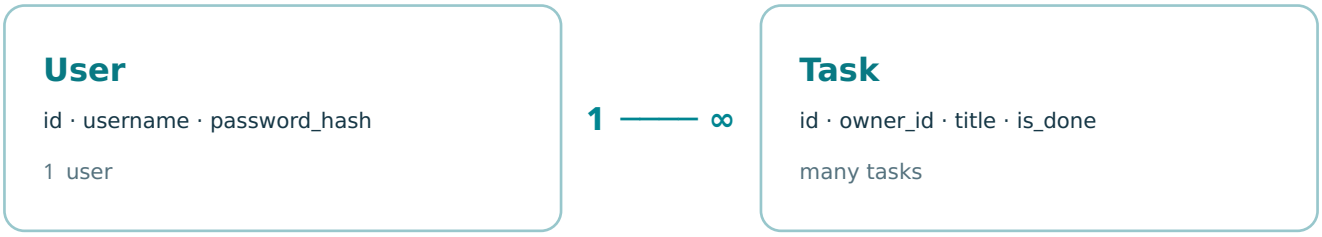
04

Database & Models

මේ කොටස තුළ ඉතා සරල උදාහරණ, සත්‍ය source code සහ ඔබටම අත්හදා බැලිය හැකි අභ්‍යාස තිබෙනවා.

මේ කොටසේ පාඩම්

- ✓ Database එකක් තෝරාගැනීම
- ✓ Model = Database සැලැස්ම
- ✓ Primary Key සහ ID
- ✓ Foreign Key = Relation
- ✓ Migrations: Table සෑදීම
- ✓ Django ORM එකෙන් Data කියවීම
- ✓ Django Admin භාවිතය



Database එකක් තෝරාගැනීම

SQLite යනු Python සමඟම වැඩ කරන සරල database එකකි. ආරම්භක project සඳහා ප්‍රයෝජනවත්; සජීවී විශාල multiuser යෙදුම් සඳහා PostgreSQL වැනි සේවාවක් අනාගතයේ සලකා බැලිය හැකියි.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. SQLite එක .sqlite3 file එකක data තබයි.
2. Django ORM මඟින් SQL අඩුවෙන් ලියන්න පුළුවන්.
3. Migration මඟින් schema එක සෑදේ.

VS CODE / little_star_tasks/settings.py

```
DATABASES = {
  "default": {
    "ENGINE": "django.db.backends.sqlite3",
    "NAME": BASE_DIR / "db.sqlite3",
  }
}
```

User

id · username · password_hash

1 user

1 —— ∞

Task

id · owner_id · title · is_done

many tasks

පුංචි අභ්‍යාසය · TRY IT

db.sqlite3 file එක GitHub push නොකළ යුතු හේතුව හඳුනාගන්න.

> **Expected result:** User-specific/private records leak වීම වැළැක්වේ.

Model = Database සැලැස්ම

Model class එකක් database table එක නියෝජනය කරයි. Task model එකේ title, priority, due_date, status වැනි fields තියෙනවා.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. models.py විවෘත කරන්න.
2. Task class එක හඳුනාගන්න.
3. එක් field එකක් = table එකේ column එකක්.

VS CODE / tasks/models.py

```
class Task(models.Model):
    class Priority(models.TextChoices):
        LOW = "low", "Low"
        MEDIUM = "medium", "Medium"
        HIGH = "high", "High"

    owner = models.ForeignKey(
        settings.AUTH_USER_MODEL,
        on_delete=models.CASCADE,
        related_name="tasks",
    )
    title = models.CharField(max_length=120)
    description = models.TextField(blank=True)
    priority = models.CharField(
        max_length=6, choices=Priority.choices, default=Priority.MEDIUM
    )
    due_date = models.DateField(blank=True, null=True)
    is_done = models.BooleanField(default=False)
    created_at = models.DateTimeField(auto_now_add=True)
    updated_at = models.DateTimeField(auto_now=True)

    class Meta:
        ordering = ["-created_at"]

    def __str__(self):
```

User
id · username · password_hash
1 user

1 —— ∞

Task
id · owner_id · title · is_done
many tasks

පුංචි අභ්‍යාසය · TRY IT

Task table එකට අදාළ තීරු 5ක් ලියා බලන්න.

> **Expected result:** title / description / priority / due_date / is_done.

Primary Key සහ ID

සෑම Task record එකකටම වෙනම ID එකක් Django ලබා දෙනවා. එයින් නිවැරදි row එක සොයාගන්න පුළුවන්; නමුත් ID දැනගැනීම permission එකක් නොවේ.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. Task #1, #2 ලෙස සිතන්න.
2. Update URL එකේ ID එක එනවා.
3. Django හි owner check තවදුරටත් අත්‍යවශ්‍යයි.

VS CODE / tasks/views.py

```
task = get_object_or_404(
    Task, pk=pk, owner=request.user
)
```

USER A · Nimali

- Task 01 · Homework
- Task 02 · Python

NO ACCESS

USER B · Amal

- Task 03 · Testing
- Task 04 · Drawing

පුංචි අභ්‍යාසය · TRY IT

වෙනත් userගේ ID එක දැනගෙන තිබුණත් edit වෙන්නේ නැත්තේ ඇයි?

> **Expected result:** owner=request.user query එක හේතුවයි.

Foreign Key = Relation

User table එකේ එක් person කෙනෙකුට Tasks බොහෝ තිබිය හැකියි. Task.owner යන ForeignKey එකෙන් ඒ සබඳතාවය නිර්මාණය වෙනවා.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. User 1 → Task A, Task B.
2. User 2 → Task C.
3. User 1 ට Task C නොපෙන්වන්න.

VS CODE / tasks/models.py

```
    owner = models.ForeignKey(
        settings.AUTH_USER_MODEL,
        on_delete=models.CASCADE,
        related_name="tasks",
    )
    title = models.CharField(max_length=120)
    description = models.TextField(blank=True)
    priority = models.CharField(
        max_length=6, choices=Priority.choices, default=Priority.MEDIUM
    )
    due_date = models.DateField(blank=True, null=True)
    is_done = models.BooleanField(default=False)
    created_at = models.DateTimeField(auto_now_add=True)
    updated_at = models.DateTimeField(auto_now=True)

    class Meta:
        ordering = ["-created_at"]

    def __str__(self):
        return self.title
```

USER A · Nimali

- Task 01 · Homework
- Task 02 · Python

NO ACCESS

USER B · Amal

- Task 03 · Testing
- Task 04 · Drawing

පුංචි අභ්‍යාසය · TRY IT

User සහ Task අතර 1-to-many diagram එකක් අඳින්න.

> **Expected result:** One user → many tasks.

Migrations: Table සෑදීම

models.py වෙනස් කළ පසු Django ට database schema යාවත්කාලීන කරන්න migrations භාවිතා කරනවා. Project ZIP එකේ initial migration එක තිබෙනවා.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. From scratch නම් makemigrations.
2. migrate මඟින් database tables සෑදේ.
3. Migration files Git එකට commit කරන්න.

VS CODE / Windows CMD

```
python manage.py makemigrations
python manage.py migrate
python manage.py showmigrations tasks
```

Command Prompt · Administrator privileges not required

```
C:\learn\django-little-star> python manage.py runserver
Watching for file changes...
Starting development server at
http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```

පුංචි අභ්‍යාසය · TRY IT

migrate අවසන් වූ පසු db.sqlite3 ගොනුවක් පෙනේද?

> **Expected result:** Local database file එක සෑදී තිබේ.

04 • Database & Models

LESSON 37 / 84

Django ORM එකෙන් Data කියවීම

Object Relational Mapper (ORM) නිසා SQL strings එකින් එක ලිවීම වෙනුවට Python syntax භාවිතා කළ හැකියි.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. Model class එක import කරන්න.
2. Task.objects.filter(...) කියලා query හදන්න.
3. owner filter එක හැම විටම තබාගන්න.

VS CODE / tasks/views.py

```
owned = Task.objects.filter(owner=request.user)
open_tasks = owned.filter(is_done=False)
count = open_tasks.count()
```

පුංචි අභ්‍යාසය · TRY IT

open සහ done tasks දෙක වෙන් කරන්න filter() වලින් කෙසේද?

> **Expected result:** is_done=False / True.

Django Admin භාවිතය

Django admin යනු staff users සඳහා සූදානම් data-management interface එකකි. Normal users ට admin access ස්වයංක්‍රීයව ලැබෙන්නේ නැහැ.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. python manage.py createsuperuser.
2. Admin username සහ password සකස් කරන්න.
3. http://127.0.0.1:8000/admin/ වෙත යන්න.

VS CODE / Windows CMD

```
python manage.py createsuperuser
python manage.py runserver
```

Command Prompt · Administrator privileges not required

```
C:\learn\django-little-star> python manage.py runserver
Watching for file changes...
Starting development server at
http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```

පුංචි අභ්‍යාසය · TRY IT

Admin panel එකේ Task model එක පෙන්වනවාද බලන්න.

> **Expected result:** Admin ට records නිරීක්ෂණය කළ හැකියි.

AUTH & LOGIN

05

Registration & Login

මේ කොටස තුළ ඉතා සරල උදාහරණ, සත්‍ය source code සහ ඔබටම අත්හදා බැලිය හැකි අභ්‍යාස තිබෙනවා.

මේ කොටසේ පාඩම්

- ✓ Authentication සහ Authorization
- ✓ UserCreationForm භාවිතා කරන්න
- ✓ Signup View එකේ POST
- ✓ Signup HTML Template
- ✓ LoginView: සූදානම් Login
- ✓ Login Page එකේ Field දෙක
- ✓ Password Hashing
- ✓ Session සහ Cookie අර්ථය
- ✓ Logout POST request එකක්
- ✓ login_required decorator

USER A · Nimali

Task 01 · Homework

Task 02 · Python

</>

NO ACCESS

USER B · Amal

Task 03 · Testing

Task 04 · Drawing

Authentication සහ Authorization

Authentication කියන්නේ “ඔබ කවුද?” තහවුරු කිරීම. Authorization කියන්නේ “ඔබට මේ task එකට අවසර තියෙනවාද?” පරීක්ෂා කිරීම.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. Signup/Login → authentication.
2. Task.owner match → authorization.
3. වෙනත් userගේ data → 404.

VS CODE / tasks/views.py

```
@login_required
def task_update(request, pk):
    task = get_object_or_404(
        Task, pk=pk, owner=request.user
    )
```

USER A · Nimali

- Task 01 · Homework
- Task 02 · Python

NO ACCESS

USER B · Amal

- Task 03 · Testing
- Task 04 · Drawing

පුංචි අභ්‍යාසය · TRY IT

Login වී තිබීම පමණක් ඔවුන්ගේ data edit කිරීමට ප්‍රමාණවත්ද?

Expected result: නැහැ. Owner/permission check අවශ්‍යයි.

UserCreationForm භාවිතා කරන්න

Signup form එක මුල සිට password logic ලියනවාට වඩා Django හි සූදානම් UserCreationForm භාවිතා කිරීම ආරක්ෂිත සහ පහසුයි.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. Form එක username/password දෙකක් ඉල්ලයි.
2. Validation rules බලයි.
3. save() මඟින් password hash කරයි.

VS CODE / tasks/views.py

```
def signup(request):
    if request.user.is_authenticated:
        return redirect("task_list")
    form = UserCreationForm(request.POST or None)
    if request.method == "POST" and form.is_valid():
        user = form.save()  # Django hashes the password.
        login(request, user)
        messages.success(request, "Welcome to Little Star Tasks!")
        return redirect("task_list")
    return render(request, "registration/signup.html", {"form": form})

@login_required
def task_list(request):
    # IMPORTANT: filter by logged-in owner before search/filtering.
    owned = Task.objects.filter(owner=request.user)
    q = request.GET.get("q", "").strip()[:100]
    status = request.GET.get("status", "all")
    tasks = owned
    if q:
        tasks = tasks.filter(Q(title__icontains=q) | Q(description__icontains=q))
    if status == "open":
        tasks = tasks.filter(is_done=False)
    elif status == "done":
        tasks = tasks.filter(is_done=True)
    return render(request, "tasks/list.html", {
```

පුංචි අභ්‍යාසය · TRY IT

සාමාන්‍ය text file එකකට password save කිරීම අවදානම් වන්නේ ඇයි?

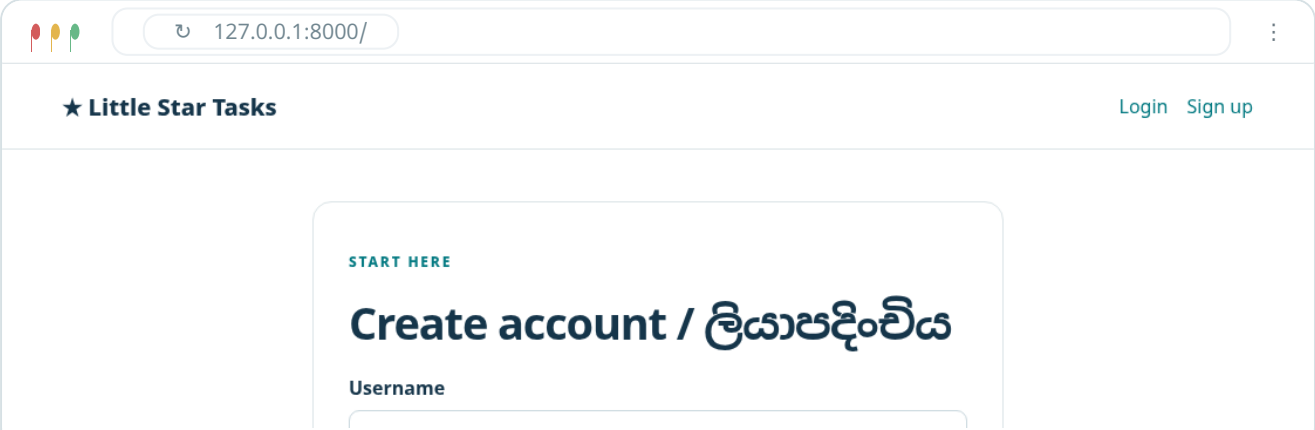
> **Expected result:** Passwords database එකේ hash ලෙසම පවතින්න ඕනේ.

Signup View එකේ POST

User form එක submit කළාම request.method == POST බව පරීක්ෂා කර valid නම් user එක save වෙනවා. අපේ code එකෙන් signup අවසන් වූ පසු user එක login කරයි.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. GET → හිස් form.
2. POST → UserCreationForm validation.
3. Valid → save + login + redirect.



පෙන්වන දර්ශනය අධ්‍යාපනික UI preview එකකි. Actual result එක local browser එකේ බලන්න.

VS CODE / tasks/views.py

```
@require_http_methods(["GET", "POST"])
def signup(request):
    if request.user.is_authenticated:
        return redirect("task_list")
    form = UserCreationForm(request.POST or None)
    if request.method == "POST" and form.is_valid():
        user = form.save()  # Django hashes the password.
        login(request, user)
        messages.success(request, "Welcome to Little Star Tasks!")
        return redirect("task_list")
    return render(request, "registration/signup.html", {"form": form})
```

පුංචි අභ්‍යාසය · TRY IT

වැරදි confirm password යෙදූ විට කුමක් සිදුවිය යුතුද?

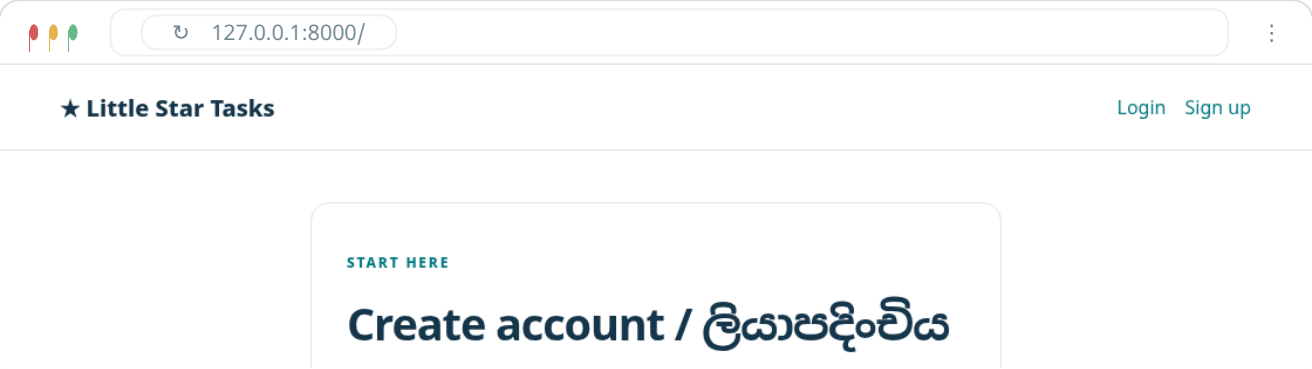
> **Expected result:** Form error පෙන්වා account සෑදීම නවතියි.

Signup HTML Template

Django form.fields loop එකෙන් inputs render කරන්න පුළුවන්. {% csrf_token %} එක POST form වල අත්‍යවශ්‍යයි.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. signup.html විවෘත කරන්න.
2. form method=post බලන්න.
3. csrf_token සහ field.errors සොයාගන්න.



පෙන්වන දර්ශනය අධ්‍යාපනික UI preview එකකි. Actual result එක local browser එකේ බලන්න.

VS CODE / templates/registration/signup.html

```
{% extends 'base.html' %}
{% block title %}Sign up - Little Star Tasks{% endblock %}
{% block content %}
<section class="panel narrow">
  <p class="kicker">START HERE</p>
  <h1>Create account / ලියාපදිංචිය</h1>
  <form method="post">
    {% csrf_token %}
    {% for field in form %}
      <label for="{{ field.id_for_label }}">{{ field.label }}</label>
      {{ field }}
      {% if field.help_text %}<small>{{ field.help_text|safe }}</small>{% endif %}
      {% for error in field.errors %}<div class="error">{{ error }}</div>{% endfor %}
    {% endfor %}
    <button type="submit" class="primary">Create account</button>
  </form>
  <p>ගිණුමක් තිබෙද? <a href="{% url 'login' %}">Login</a></p>
</section>
{% endblock %}
```

පුංචි අභ්‍යාසය · TRY IT

Password fields සඳහා error message පෙනෙන තැන සොයාගන්න.

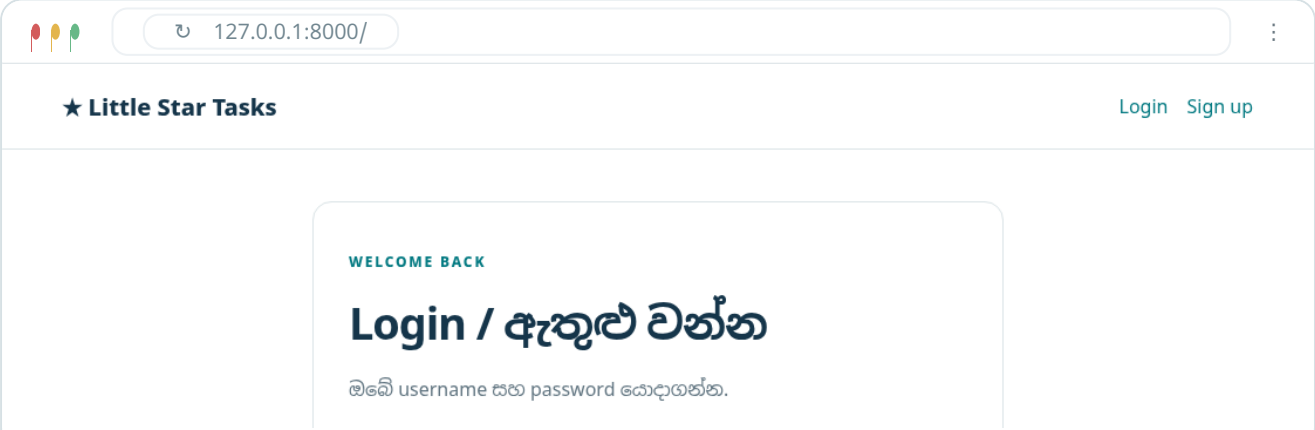
> **Expected result:** form.errors පෙන්වා user ට නිවැරදි කිරීමට හැකි වේ.

LoginView: සූදානම් Login

Django හි LoginView authentication session එක handle කරනවා. Password compare කිරීම ඔබ විසින් ලියනවාට වඩා built-in mechanism භාවිතා කිරීම හොඳයි.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. urlpatterns තුළ LoginView බලන්න.
2. login.html form එක ලියන්න.
3. Success → LOGIN_REDIRECT_URL වෙත.



පෙන්වන දර්ශනය අධ්‍යාපනික UI preview එකකි. Actual result එක local browser එකේ බලන්න.

VS CODE / little_star_tasks/urls.py

```
path(
  "accounts/login/",
  auth_views.LoginView.as_view(
    template_name="registration/login.html"
  ), name="login"
)
```

පුංචි අභ්‍යාසය · TRY IT

වෙනම “if password == ...” code එකක් නොලියන්න.

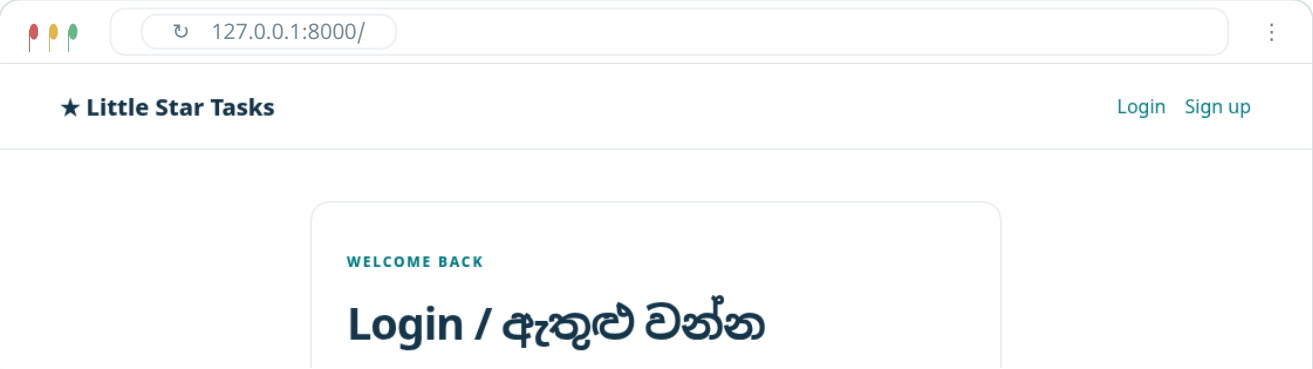
> **Expected result:** Django AuthenticationForm valid user තෝරා දෙයි.

Login Page එකේ Field දෙක

Login page එකේ username/password fields සහ submit button එක තියෙනවා. form.errors වැරදි login පෙන්වයි.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. login.html බලන්න.
2. form.username සහ form.password පෙන්වනවා.
3. hidden next field එක redirect සඳහා ඇත.



පෙන්වන දර්ශනය අධ්‍යාපනික UI preview එකකි. Actual result එක local browser එකේ බලන්න.

VS CODE / templates/registration/login.html

```
{% extends 'base.html' %}
{% block title %}Login - Little Star Tasks{% endblock %}
{% block content %}
<section class="panel narrow">
  <p class="kicker">WELCOME BACK</p>
  <h1>Login / ඇතුළු වන්න</h1>
  <p>ඔබේ username සහ password යොදන්න.</p>
  {% if form.errors %}<div class="error">Username හෝ password වැරදියි.</div>{% endif %}
  <form method="post">
    {% csrf_token %}
    <label for="id_username">Username</label>
    {{ form.username }}
    <label for="id_password">Password</label>
    {{ form.password }}
    <input type="hidden" name="next" value="{{ next }}">
    <button type="submit" class="primary">Login</button>
  </form>
  <p>අලුත් ගිණුමක්? <a href="{% url 'signup' %}">Sign up</a></p>
```

පුංචි අභ්‍යාසය · TRY IT

වැරදි password එකක් යොදා UI error message එක නිරීක්ෂණය කරන්න.

> **Expected result:** Login නොවී එම page එකේ error message පෙනෙයි.

Password Hashing

Passwords යනු plain text ලෙස database එකේ තබන දේවල් නොවේ. Django auth හි password hashing සහ validators සූදානම්ව තිබෙනවා.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. UserCreationForm භාවිතා කරන්න.
2. set_password() හෝ create_user() භාවිතා කරන්න.
3. password field එක compare නොකර authenticate කරන්න.

VS CODE / Python / concept

```
from django.contrib.auth import get_user_model
User = get_user_model()
user = User.objects.create_user(
    username="nimal",
    password="use-a-strong-secret"
)
```

පුංචි අභ්‍යාසය · TRY IT

Plain text password save කරන code එකක් හොඳ නැත්තේ ඇයි?

> **Expected result:** Leak එකකදී සෘජුව password දැකිය හැකි නිසා.

Session සහ Cookie අර්ථය

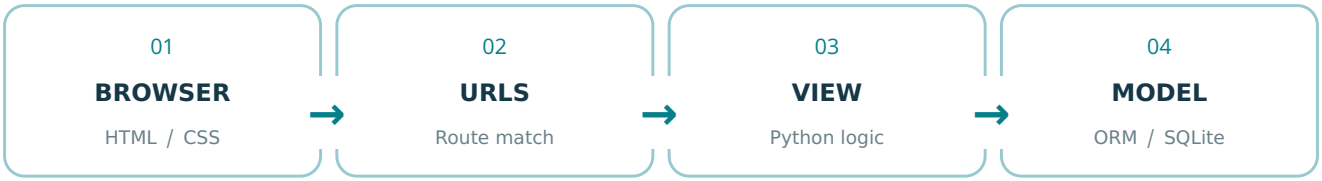
Login වූ පසු Browser එකට session cookie ලැබෙනවා. Django server එක session data භාවිතා කර request.user කවුදැයි හඳුනාගනී.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. Login request එක යන්න.
2. Server session එකක් පවත්වාගනී.
3. Browser cookie යවා පසුව requests සිදු කරයි.

VS CODE / Python / concept

```
if request.user.is_authenticated:
    print(request.user.username)
```



පුංචි අභ්‍යාසය · TRY IT

Browser cookie එකක් password එකමද?

> **Expected result:** නැහැ. එය session identification සඳහා භාවිතා වේ.

Logout POST request එකක්

Django 6.1 LogoutView POST requests මත logout සිදු කරයි. Logout සඳහා සරල GET link එකක් වෙනුවට csrf_token සහිත POST form එකක් භාවිතා කරන්න.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. base.html logout form සොයාගන්න.
2. method=post තහවුරු කරන්න.
3. csrf_token ඇතුළත් බව පරීක්ෂා කරන්න.

VS CODE / templates/base.html

```
          <form method="post" action="{% url 'logout' %}" class="inline-form">
            {% csrf_token %}
            <button class="link-button" type="submit">Logout</button>
          </form>
        {% else %}
          <a href="{% url 'login' %}">Login</a>
          <a href="{% url 'signup' %}">Sign up</a>
        {% endif %}
      </nav>
    </div>
  </header>
  <main class="wrap main-space">
    {% if messages %}
      {% for message in messages %}<div class="notice" role="status">{{ message }}</div>{% endfor %}
    {% endif %}
    {% block content %}{% endblock %}
  </main>
  <footer class="footer"><div class="wrap">Little Star learning project · egotechworld.com</div></footer>
</body>
```

පුංචි අභ්‍යාසය · TRY IT

Logout පසු back button එකෙන් private tasks දකින්න පුළුවන්ද?

> **Expected result:** New request එක login page වෙත යා යුතුයි.

login_required decorator

Task pages අමුත්තන්ට විවෘත නොවීමට @login_required යොදනවා. Login නැත්නම් Django අදාළ login page වෙත redirect කරයි.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. function එකට කලින් decorator එක දාන්න.
2. LOGIN_URL settings පරීක්ෂා කරන්න.
3. Anonymous browser tab එකෙන් / visit කරන්න.

VS CODE / tasks/views.py

```
@login_required
def task_list(request):
    owned = Task.objects.filter(
        owner=request.user
    )
```

පුංචි අභ්‍යාසය · TRY IT

Login නැති tab එකක /tasks/new/ open කරන්න.

> **Expected result:** Login page වෙත යවයි.

CREATE · READ · UPDATE · DELETE

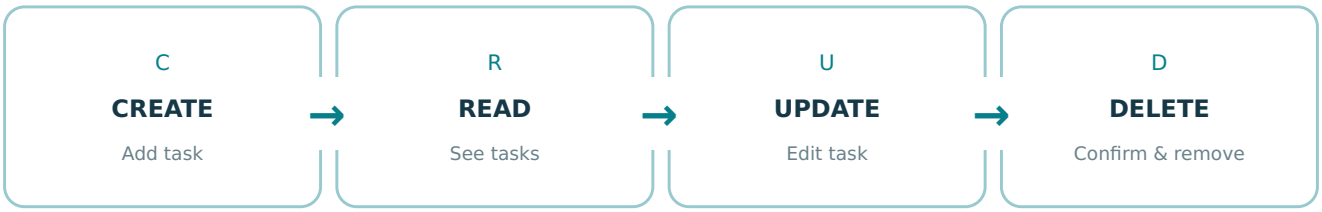
06

CRUD Tasks

මේ කොටස තුළ ඉතා සරල උදාහරණ, සත්‍ය source code සහ ඔබටම අත්හදා බැලිය හැකි අභ්‍යාස තිබෙනවා.

මේ කොටසේ පාඩම්

- ✓ ModelForm එකෙන් Form හදමු
- ✓ Create Form UI
- ✓ Create View එකේ save(commit=False)
- ✓ READ: තමන්ගේ tasks පමණක්
- ✓ Template එකේ for loop
- ✓ UPDATE: Task එක වෙනස් කරමු
- ✓ DELETE: පළමුව Confirm
- ✓ DELETE: POST මත පමණයි
- ✓ CRUD අනුපිළිවෙළ යළි බලමු



ModelForm එකෙන් Form හදමු

Form class එක model fields වලට ගැළපෙන input fields සකස් කරයි. Browser එකෙන් පැමිණෙන values server-side validate කරයි.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. forms.py විවෘත කරන්න.
2. class TaskForm(forms.ModelForm) හඳුනාගන්න.
3. Meta.fields හි fields පහ පරීක්ෂා කරන්න.

VS CODE / tasks/forms.py

```
"""Validate user input with Django ModelForm."""
from django import forms
from .models import Task

class TaskForm(forms.ModelForm):
    class Meta:
        model = Task
        fields = ["title", "description", "priority", "due_date", "is_done"]
        widgets = {
            "title": forms.TextInput(attrs={"placeholder": "e.g. Read a Django lesson"}),
            "description": forms.Textarea(attrs={"rows": 3}),
            "due_date": forms.DateInput(attrs={"type": "date"}),
        }
```

පුංචි අභ්‍යාසය · TRY IT

owner field එක form.fields තුළ නැත්තේ ඇයි?

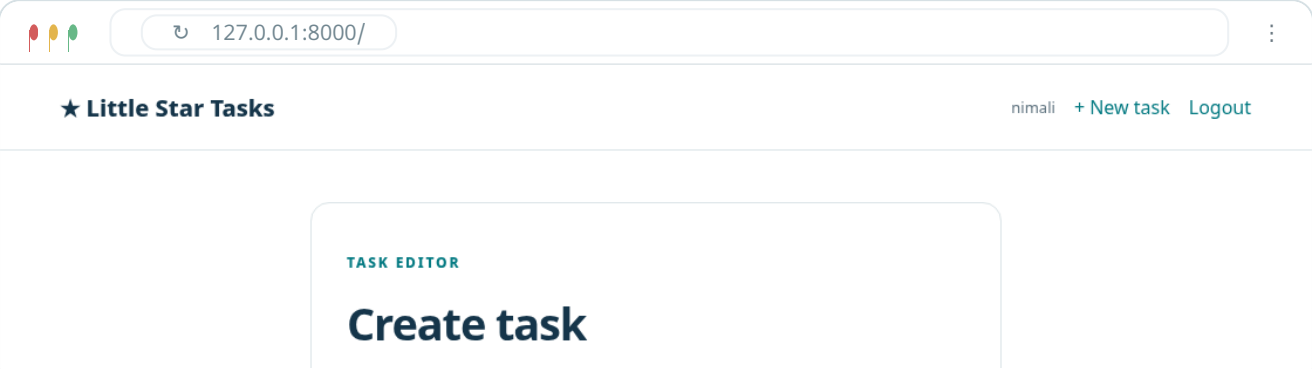
> **Expected result:** Owner සෑම විටම server එකෙන් request.user මඟින් දෙයි.

Create Form UI

Task title, description, priority, due date සහ done status පුරවා “Save task” කරන්න. Browser preview එකේ පෙන්නන්නේ UI එකේ ඉලක්ක layout එකයි.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. New task link එක click කරන්න.
2. Form fields හඳුනාගන්න.
3. Title එකක් ඇතුළත් කර Submit කරන්න.



පෙන්වන දර්ශනය අධ්‍යාපනික UI preview එකකි. Actual result එක local browser එකේ බලන්න.

VS CODE / templates/tasks/form.html

```
{% extends 'base.html' %}
{% block title %}{{ mode }} task - Little Star{% endblock %}
{% block content %}
<section class="panel narrow">
  <p class="kicker">TASK EDITOR</p>
  <h1>{{ mode }} task</h1>
  <form method="post">
    {% csrf_token %}
    {% if form.non_field_errors %}<div class="error">{{ form.non_field_errors }}</div>{% endif %}
    {% for field in form %}
      <label for="{{ field.id_for_label }}">{{ field.label }}</label>
      {{ field }}
      {% for error in field.errors %}<div class="error">{{ error }}</div>{% endfor %}
    {% endfor %}
    <button type="submit" class="primary">Save task</button>
    <a class="cancel" href="{% url 'task_list' %}">Cancel</a>
  </form>
</section>
{% endblock %}
```

පුංචි අභ්‍යාසය · TRY IT

Title එක හිස් තබා submit කළොත් කුමක් වෙයිද?

> **Expected result:** Django validation error පෙන්වයි.

Create View එකේ save(commit=False)

Task එක තවම database එකට save නොකර owner field එක සුරක්ෂිතව set කිරීමට commit=False යොදාගන්නවා.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. Form is_valid() පරීක්ෂා කරන්න.
2. form.save(commit=False) ලබාගන්න.
3. task.owner=request.user දා save කරන්න.

VS CODE / tasks/views.py

```python
def task_create(request):
    form = TaskForm(request.POST or None)
    if request.method == "POST" and form.is_valid():
        task = form.save(commit=False)
        task.owner = request.user  # NEVER trust an owner field from the browser.
        task.save()
        messages.success(request, "Task saved.")
        return redirect("task_list")
    return render(request, "tasks/form.html", {"form": form, "mode": "Create"})

@login_required
def task_update(request, pk):
    task = get_object_or_404(Task, pk=pk, owner=request.user)
    form = TaskForm(request.POST or None, instance=task)
```

පුංචි අභ්‍යාසය · TRY IT

Browser එකෙන් owner=3 යැව්වත් server owner කවුද?

> **Expected result:** Login කළ request.user පමණයි.

READ: තමන්ගේ tasks පමණක්

Database එකේ සියලුම records පෙන්වන්න එපා. Task.objects.filter(owner=request.user) නිසා user-specific dashboard එකක් ලැබෙනවා.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. task_list view එක සොයාගන්න.
2. owned කියන queryset එක බලන්න.
3. Template එකට tasks යවනවා.



VS CODE / tasks/views.py

```python
def task_list(request):
    # IMPORTANT: filter by logged-in owner before search/filtering.
    owned = Task.objects.filter(owner=request.user)
    q = request.GET.get("q", "").strip()[:100]
    status = request.GET.get("status", "all")
    tasks = owned
    if q:
        tasks = tasks.filter(Q(title__icontains=q) | Q(description__icontains=q))
    if status == "open":
        tasks = tasks.filter(is_done=False)
    elif status == "done":
        tasks = tasks.filter(is_done=True)
    return render(request, "tasks/list.html", {
        "tasks": tasks,
        "q": q,
        "status": status,
        "total_count": owned.count(),
        "done_count": owned.filter(is_done=True).count(),
    })
```

පුංචි අභ්‍යාසය · TRY IT

All Task.objects.all() යෙදීමේ අවදානම පැහැදිලි කරන්න.

> **Expected result:** වෙනත් usersගේ තොරතුරු leak විය හැකියි.

Template එකේ for loop

tasks ලැයිස්තුවෙන් සෑම task එකක්ම HTML card එකක් ලෙස පෙන්වන්න {% for %} භාවිතා කරනවා. Empty නම් වෙනම message එකක්.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. list.html විවෘත කරන්න.
2. {% for task in tasks %} සොයාගන්න.
3. {% empty %} state එකත් හඳුනාගන්න.



පෙන්වන දර්ශනය අධ්‍යාපනික UI preview එකකි. Actual result එක local browser එකේ බලන්න.

VS CODE / templates/tasks/list.html

```
  {% for task in tasks %}
    <article class="task-card">
      <div><h2>{{ task.title }}</h2><p>{{ task.description|default:'No description yet' }}</p>
      <small>{{ task.get_priority_display }} priority · {% if task.is_done %}Done{% else %}Open{% en
dif %}{% if task.due_date %} · Due {{ task.due_date }}{% endif %}</small></div>
      <div class="actions"><a href="{% url 'task_update' task.pk %}">Edit</a><a href="{% url 'task_d
elete' task.pk %}">Delete</a></div>
    </article>
  {% empty %}<div class="panel">No tasks yet. Create your first one!</div>
  {% endfor %}
</div>
{% endblock %}
```

පුංචි අභ්‍යාසය · TRY IT

Tasks 0ක් නම් “No tasks yet” පෙන්වෙනවාද බලන්න.

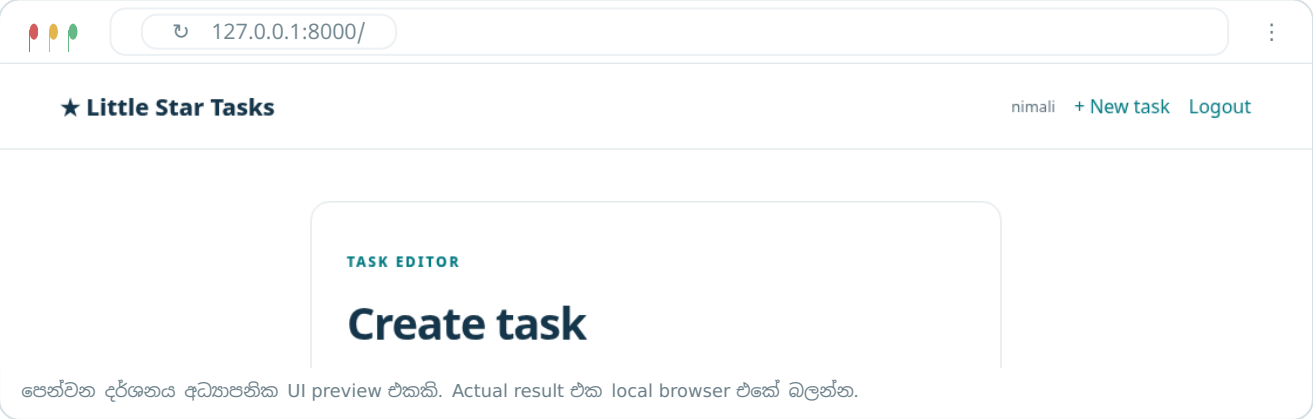
> **Expected result:** Empty-state message එක පෙනේ.

UPDATE: Task එක වෙනස් කරමු

Edit page එකට Task object එක instance ලෙස දුන්නාම form එකේ කලින් values පෙනෙනවා. POST valid නම් update save වෙනවා.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. Edit link එකෙන් ID එක URL එකට යනවා.
2. owner check එකෙන් task එක සොයයි.
3. TaskForm(instance=task) යොදා form පෙන්වයි.



පෙන්වන දර්ශනය අධ්‍යාපනික UI preview එකකි. Actual result එක local browser එකේ බලන්න.

VS CODE / tasks/views.py

```python
def task_update(request, pk):
    task = get_object_or_404(Task, pk=pk, owner=request.user)
    form = TaskForm(request.POST or None, instance=task)
    if request.method == "POST" and form.is_valid():
        form.save()
        messages.success(request, "Task updated.")
        return redirect("task_list")
    return render(request, "tasks/form.html", {"form": form, "mode": "Edit"})

@login_required
@require_http_methods(["GET", "POST"])
def task_delete(request, pk):
```

පුංචි අභ්‍යාසය · TRY IT

Task title “Homework” → “Homework done” කරන්න.

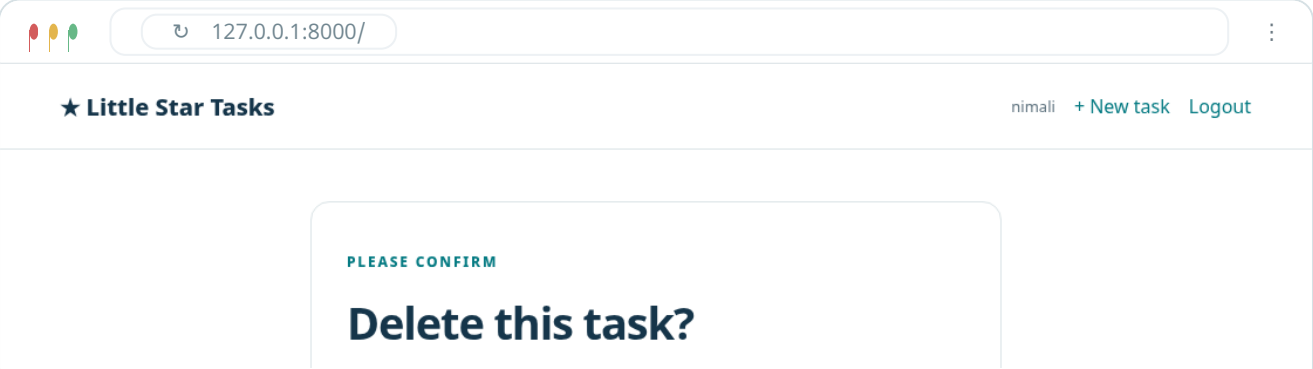
> **Expected result:** Dashboard එකේ නව title එක පෙනෙයි.

DELETE: පළමුව Confirm

"Delete" link click කළ විට ක්ෂණිකව data මැකීම වෙනුවට confirm page එකක් පෙන්වීම user-friendly සහ ආරක්ෂිතයි.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. GET request එක confirm page පෙන්වයි.
2. Yes, delete button POST යවයි.
3. Cancel link data නොවෙනස් කර ආපසු යයි.



පෙන්වන දර්ශනය අධ්‍යාපනික UI preview එකකි. Actual result එක local browser එකේ බලන්න.

VS CODE / templates/tasks/confirm_delete.html

```
{% extends 'base.html' %}
{% block title %}Delete task - Little Star{% endblock %}
{% block content %}
<section class="panel narrow">
  <p class="kicker">PLEASE CONFIRM</p>
  <h1>Delete this task?</h1>
  <p>"{{ task.title }}" යන්න මකා දැමීමට අවශ්‍යද?</p>
  <form method="post">
    {% csrf_token %}
    <button type="submit" class="danger">Yes, delete</button>
    <a class="cancel" href="{% url 'task_list' %}">No, go back</a>
  </form>
</section>
{% endblock %}
```

පුංචි අභ්‍යාසය · TRY IT

Cancel තෝරා data ඉතිරි වෙනවාද පරීක්ෂා කරන්න.

> **Expected result:** Task එක තවමත් database එකේ ඇත.

DELETE: POST මත පමණයි

Delete action එක GET request එකකින් සිදු නොවිය යුතුයි. POST සහ owner permission check එකක් සහිතව පමණක් record එක මකා දමනවා.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. get_object_or_404(owner=request.user) යොදාගන්න.
2. request.method == POST නම් task.delete() කරන්න.
3. ඊට පසුව dashboard වෙත redirect කරන්න.

VS CODE / tasks/views.py

```
def task_delete(request, pk):
    task = get_object_or_404(Task, pk=pk, owner=request.user)
    if request.method == "POST":
        task.delete()
        messages.success(request, "Task deleted.")
        return redirect("task_list")
    # GET shows confirmation; it never deletes data.
    return render(request, "tasks/confirm_delete.html", {"task": task})
```

පුංචි අභ්‍යාසය · TRY IT

Delete URL එක browser address bar එකෙන් බලන විට මැකෙනවාද?

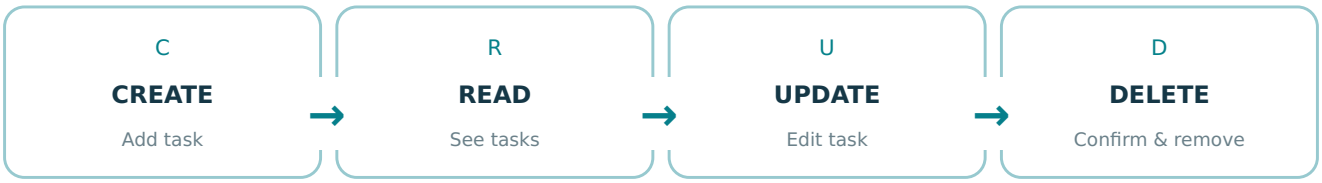
> **Expected result:** නැහැ. Confirm page එක පමණක් පෙනෙනවා.

CRUD අනුපිළිවෙළ යළි බලමු

දැන් C/R/U/D හතරම එකම Task model එකත්, user login system එකත් සමඟ ක්‍රියා කරනවා. Server-side ownership logic එකත් අමතක කරන්න එපා.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. C → new form → save.
2. R → filtered queryset → cards.
3. U → existing instance → save.
4. D → confirm POST → delete.



පුංචි අභ්‍යාසය · TRY IT

Task එකක් සාදා, title වෙනස් කර, list එකෙන් බලා, අවසානයේ මකා දමන්න.

> **Expected result:** වැඩ හතරම browser එකෙන් තහවුරු වේ.

POLISHED UI

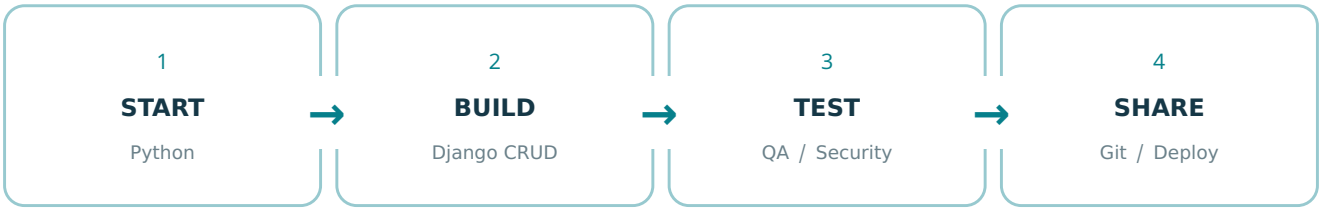
07

Search & UI

මේ කොටස තුළ ඉතා සරල උදාහරණ, සත්‍ය source code සහ ඔබටම අත්හදා බැලිය හැකි අභ්‍යාස තිබෙනවා.

මේ කොටසේ පාඩම්

- ✓ Dashboard එකේ Counts
- ✓ Search Box එක
- ✓ Status Filter එක
- ✓ Messages සහ User Feedback
- ✓ Responsive UI කියන්නේ?
- ✓ Chrome DevTools භාවිතා කරන්න



Dashboard එකේ Counts

Dashboard එකේ Total සහ Done card දෙකෙන් ඔබේ වැඩවල තත්ත්වය ඉක්මනින් දැනගන්න පුළුවන්. මේ counts ගන්නේ ඔබට අයිති queryset එකෙන්මයි.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. owned.count() මඟින් total.
2. owned.filter(is_done=True).count() මඟින් done.
3. දත්ත view එකෙන් template එකට යවයි.



VS CODE / tasks/views.py

```
total_count = owned.count()
done_count = owned.filter(
    is_done=True
).count()
```

පුංචි අභ්‍යාසය · TRY IT

Tasks 4ක් සහ done 2ක් තිබේ නම් cards අගයන් මොනවාද?

> **Expected result:** Total 4, Done 2.

Search Box එක

Search form එකෙන් GET q parameter එක යවනවා. Django ORM හි icontains භාවිතා කර title/ description තුළ ගැළපෙන records හොයන්න පුළුවන්.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. request.GET.get("q") කියවාගන්න.
2. Text එක trim කරන්න.
3. owner-scoped tasks වලට search filter යොදන්න.



පෙන්වන දර්ශනය අධ්‍යාපනික UI preview එකකි. Actual result එක local browser එකේ බලන්න.

VS CODE / tasks/views.py

```
q = request.GET.get("q", "").strip()[:100]
if q:
    tasks = tasks.filter(
        Q(title__icontains=q) |
        Q(description__icontains=q)
    )
```

පුංචි අභ්‍යාසය · TRY IT

“Python” යන search එකෙන් title match වෙන records බලන්න.

> **Expected result:** තමන්ගේ tasks වලින් ගැළපෙන ඒවා පමණක්.

Status Filter එක

Tasks Completed/Open ලෙස filter කරන්න පුළුවන්. User input ලෙස එන status අගය පරීක්ෂා කර තේරුම්ගත් values වලට පමණක් logic යොදනවා.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. status="open" නම් is_done=False.
2. status="done" නම් is_done=True.
3. වෙනත් value නම් සියල්ල.

VS CODE / tasks/views.py

```
if status == "open":
    tasks = tasks.filter(is_done=False)
elif status == "done":
    tasks = tasks.filter(is_done=True)
```

පුංචි අභ්‍යාසය · TRY IT

Done filter එකෙන් completed tasks පමණක් පෙනෙනවාද?

> **Expected result:** Done records පමණයි.

Messages සහ User Feedback

Save, update, delete වූ පසු user ට confirmation message එකක් පෙන්වීම හොඳ UX practice එකක්. Django messages framework එක මේ සඳහා භාවිතා කළ හැකියි.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. messages.success() දාන්න.
2. redirect() වෙත යන්න.
3. base.html messages loop එකෙන් text පෙන්වන්න.

VS CODE / tasks/views.py

```
messages.success(request, "Task saved.")
return redirect("task_list")
```

පුංචි අභ්‍යාසය · TRY IT

Save කළ පසු confirmation message එක පෙනෙනවාද බලන්න.

> **Expected result:** Task saved. පෙනෙයි.

Responsive UI කියන්නේ?

Desktop, tablet සහ mobile screens වෙනස් පළලක් ඇත. CSS media queries භාවිතා කර forms/ card layout mobile එකට ගැළපිය හැකියි.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. Chrome DevTools open කරන්න.
2. Device toolbar සක්‍රිය කරන්න.
3. 375px පළලේ UI එක පරීක්ෂා කරන්න.



VS CODE / static/css/site.css

```
@media(max-width:650px) {
  .topbar, .heading-row, .task-card {
    flex-direction:column;
  }
}
```

පුංචි අභ්‍යාසය · TRY IT

Mobile view එකේ horizontal scrollbar මතුවෙනවාද බලන්න.

> **Expected result:** Buttons සහ inputs screen එකට ගැළපෙයි.

Chrome DevTools භාවිතා කරන්න

UI එක වැඩ නොකරන්නේ නම් Console සහ Network tabs බලන්න. Browser console එකේ errors, network response codes, loaded CSS හඳුනාගන්න.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. F12 හෝ Ctrl+Shift+I ඔබන්න.
2. Console tab එකෙන් errors බලන්න.
3. Network tab එකෙන් page request එකේ status බලන්න.

VS CODE / Chrome DevTools / note

```
HTTP 200 = OK
HTTP 302 = Redirect
HTTP 403 = Forbidden
HTTP 404 = Not Found
```

Command Prompt · Administrator privileges not required

```
C:\learn\django-little-star> python manage.py runserver
Watching for file changes...
Starting development server at
http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```

පුංචි අභ්‍යාසය · TRY IT

Anonymous user කෙනෙකු / යද්දී 302 redirect එකක් පෙනේද?

> **Expected result:** Login page වෙත redirect වේ.

SECURITY & QUALITY

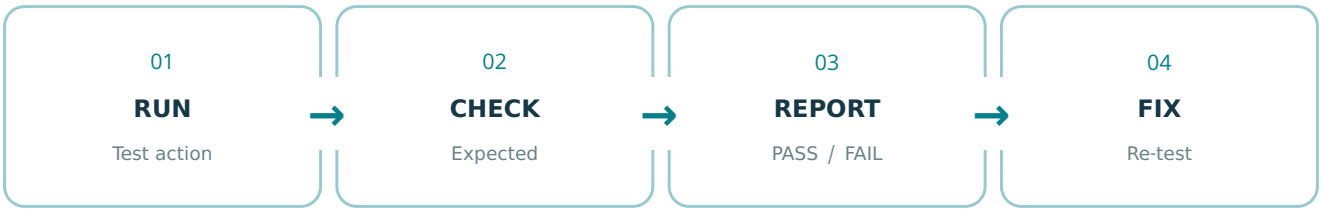
08

Security & Testing

මේ කොටස තුළ ඉතා සරල උදාහරණ, සත්‍ය source code සහ ඔබටම අත්හදා බැලිය හැකි අභ්‍යාස තිබෙනවා.

මේ කොටසේ පාඩම්

- ✓ CSRF Token එක වැදගත්
- ✓ SQL Injection ගැන දැනගමු
- ✓ XSS: User Text පෙන්වන විට
- ✓ Access Control: Owner Check
- ✓ Testing කියන්නේ මොකක්ද?
- ✓ Django TestCase හැඳින්වීම
- ✓ Automated Tests Run කරන්න
- ✓ QA Test Case ලිවීම
- ✓ Bug Report සහ Regression



CSRF Token එක වැදගත්

Browser එකේ login වී සිටින user කෙනෙකුට අදාළ data වෙනස් කිරීම වෙනත් website එකක් මඟින් බලහත්කාරයෙන් සිදු නොවීමට Django CSRF protection උදව් කරනවා.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. POST form සෑම තැනකම {% csrf_token %}.
2. CSRF middleware enabled.
3. Delete POST සඳහාත් token අවශ්‍යයි.

VS CODE / Django Template

```
<form method="post">
  {% csrf_token %}
  <button type="submit">Save</button>
</form>
```

පුංචි අභ්‍යාසය · TRY IT

CSRF token ඉවත් කර test කළොත් expected status මොකක්ද?

> **Expected result:** CSRF enforcement යටතේ 403 Forbidden.

SQL Injection ගැන දැනගමු

Browser එකෙන් text එන විට SQL strings එකට අමු අගය concatenate කිරීම අවදානම්. Django ORM භාවිතා කර parameterized queries මඟින් සාමාන්‍ය queries වඩා ආරක්ෂිතව සෑදිය හැකියි.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. Raw SQL එකට user input එක නොඑකතු කරන්න.
2. Task.objects.filter(...) භාවිතා කරන්න.
3. Input validation කරන්න.

VS CODE / tasks/views.py

```
Task.objects.filter(
    owner=request.user,
    title__icontains=q
)
```

පුංචි අභ්‍යාසය · TRY IT

“SQL injection safe” කියලා ඕනෑම SQL string එකක් concatenate කළ හැකිද?

> **Expected result:** නැහැ. Raw SQL ලියන්නේ නම් safe parameters අවශ්‍යයි.

XSS: User Text පෙන්වන විට

Task title එකට HTML tags යෙදුවත් ඒවා executable code ලෙස නොව text ලෙස පෙන්වීම වැදගත්. Django templates සාමාන්‍යයෙන් variables autoescape කරනවා.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. {{ task.title }} මඟින් text පෙන්වන්න.
2. Untrusted content මත |safe යොදන්න එපා.
3. HTML injection example එකක් text ලෙස නිරීක්ෂණය කරන්න.

VS CODE / templates/tasks/list.html

```
<h2>{{ task.title }}</h2>
<p>{{ task.description }}</p>
```

පුංචි අභ්‍යාසය · TRY IT

Task title එකට Test ඇතුළත් කර HTML නොව text ලෙස පෙනේද බලන්න.

> **Expected result:** Markup render නොවී text ලෙස පෙන්විය යුතුයි.

Access Control: Owner Check

User A login වූ විට User Bගේ task ID එක URL එකට දී try කළත් Django 404 ලබා දිය යුතුයි. Login check එකට අමතරව මේ record-level check එක අවශ්‍යයි.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. User A සහ User B account දෙකක් හදන්න.
2. A task URL එක B ගිණුමෙන් try කරන්න.
3. Edit/Delete දෙකම 404 බව සටහන් කරන්න.

VS CODE / tasks/views.py

```
task = get_object_or_404(
    Task, pk=pk, owner=request.user
)
```

USER A · Nimali

- Task 01 · Homework
- Task 02 · Python

NO ACCESS

USER B · Amal

- Task 03 · Testing
- Task 04 · Drawing

පුංචි අභ්‍යාසය · TRY IT

ID increment වෙන බව අනුමාන කළොත් permission ලැබෙනවාද?

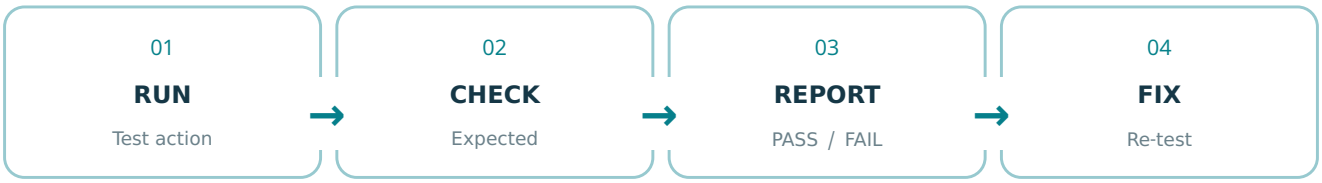
> **Expected result:** නැහැ. Owner mismatch → 404.

Testing කියන්නේ මොකක්ද?

Software එක හදලා ඉවරයි කියලා කියන්න නම් buttons පමණක් click කිරීම ප්‍රමාණවත් නැහැ. Functional, negative, security සහ usability tests කරන්න ඕනේ.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. Positive: valid task save වේ.
2. Negative: blank title reject වේ.
3. Security: B userගේ data A ට නොපෙන්වයි.
4. Usability: phone UI readable.



පුංචි අභ්‍යාසය · TRY IT

මෙම categories හතරට test එකක් බැගින් ලියන්න.

> **Expected result:** QA checklist එකක් සෑදේ.

Django TestCase හැඳින්වීම

Django හි TestCase තාවකාලික test database එකක් භාවිතා කර assertions පරීක්ෂා කරනවා. Test records සැබෑ user data සමඟ මිශ්‍ර නොකරයි.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. tests.py බලන්න.
2. self.client.get / post භඳුනාගන්න.
3. assertEqual වැනි checks බලන්න.

VS CODE / tasks/tests.py

```
class TaskAppTests(TestCase):
    def setUp(self):
        self.alex = User.objects.create_user(username="alex", password="SafePass123!abc")
        self.maya = User.objects.create_user(username="maya", password="SafePass123!abc")
        self.a_task = Task.objects.create(owner=self.alex, title="Alex private task")
        self.m_task = Task.objects.create(owner=self.maya, title="Maya secret task")

    def test_login_required(self):
        response = self.client.get(reverse("task_list"))
        self.assertEqual(response.status_code, 302)
        self.assertIn("/accounts/login/", response.url)

    def test_login_success(self):
        response = self.client.post(reverse("login"), {
            "username": "alex", "password": "SafePass123!abc"
        })
        self.assertRedirects(response, reverse("task_list"))
```

පුංචි අභ්‍යාසය · TRY IT

TestCase එකක් run වූ පසු test data ගබඩා කර තබනවාද?

> **Expected result:** සාමාන්‍යයෙන් test database එක නිකුත් කෙරේ.

Automated Tests Run කරන්න

Tests project එකේ ඇතුළත්, නමුත් අවසන් සැබෑ test result ඔබේ පරිගණකයේ Django install කර run කිරීමෙන් තහවුරු කළ යුතුයි.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. .venv activate කරගන්න.
2. python manage.py test run කරන්න.
3. Failures තිබුණොත් traceback කියවන්න.

VS CODE / Windows CMD

```
python manage.py check
python manage.py test
python manage.py test tasks
```

Command Prompt · Administrator privileges not required

```
C:\learn\django-little-star> python manage.py runserver
Watching for file changes...
Starting development server at
http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```

පුංචි අභ්‍යාසය · TRY IT

Test output එකේ tests count සහ PASS/FAIL ලියන්න.

> **Expected result:** Tests 18ක් ඇතුළත්; local result තහවුරු කරන්න.

QA Test Case ලිවීම

සෑම test එකකටම ID, steps, expected, actual සහ status තියෙන්න ඕනේ. Test report එකකට screenshots ද වැදගත්.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. TC01: guest dashboard → login redirect.
2. TC05: empty title → error.
3. TC09: other user edit → 404.

VS CODE / QA / test-case template

```
TC09 | A tries editing B task
Expected: 404 Not Found
Actual: ______
Status: PASS / FAIL
```

පුංචි අභ්‍යාසය · TRY IT

TC10 සඳහා delete-other-user test එකක් ලියන්න.

> **Expected result:** Expected 404; database unchanged.

Bug Report සහ Regression

Bug එකක් හමු වුණොත් නැවත සිදු කරන steps සටහන් කරන්න. Bug fix කළ පසු පැරණි වැඩ තවමත් හොඳින් ක්‍රියා කරනවාද යන්න regression testing මඟින් බලන්න.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. Bug ID හා summary.
2. Reproduce කරන steps.
3. Expected vs actual result.
4. Screenshot, priority, status.

VS CODE / QA / bug template

```
BUG-004: Delete button not working
Steps: login > open task > delete
Expected: confirmation page
Actual: 404 error
Severity: Medium
```

පුංචි අභ්‍යාසය · TRY IT

Bug fix කළ පසු create/update/list නැවත test කරන්නේ ඇයි?

> **Expected result:** New change එකෙන් පරණ features කැඩිය හැකි නිසා.

GIT & LIVE DEPLOYMENT

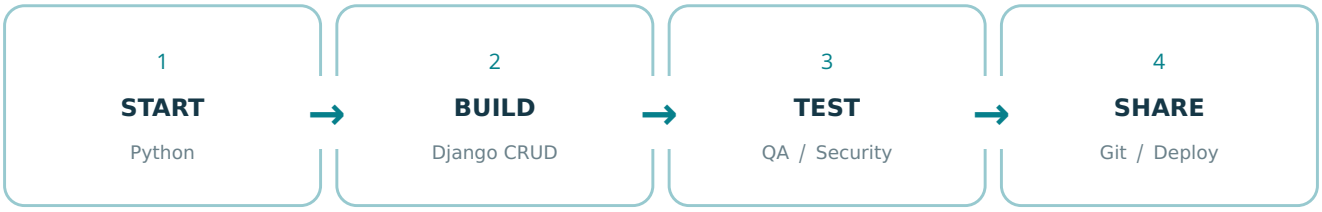
09

Git & Deployment

මේ කොටස තුළ ඉතා සරල උදාහරණ, සත්‍ය source code සහ ඔබටම අත්හදා බැලිය හැකි අභ්‍යාස තිබෙනවා.

මේ කොටසේ පාඩම්

- ✓ Git එකෙන් code සුරකිමු
- ✓ .gitignore එක වැදගත්
- ✓ Deployment කියන්නේ මොකක්ද?
- ✓ Database Backup සහ Restore
- ✓ Live App එකට තව මොනවාද?
- ✓ 2026 Django Learning Roadmap



Git එකෙන් code සුරකිමු

Git කියන්නේ code එකේ versions තබාගන්න ක්‍රමයක්. Django project එකේ source files track කළත් passwords, user database සහ virtualenv commit නොකළ යුතුයි.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. git init කරන්න.
2. git status බලන්න.
3. git add . සහ commit කරන්න.

VS CODE / Windows CMD

```
git init
git add .
git commit -m "Start Django CRUD app"
```

Command Prompt · Administrator privileges not required

```
C:\learn\django-little-star> python manage.py runserver
Watching for file changes...
Starting development server at
http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```

පුංචි අභ්‍යාසය · TRY IT

Commit එකට පෙර git status බලන්න.

> **Expected result:** Source files commit වෙනවා; .venv/db.sqlite3 නැහැ.

.gitignore එක වැදගත්

.gitignore ගොනුව Git වෙත යා නොයුතු files සඳහන් කරනවා. Local database සහ secrets අන්තර්ජාලයට නොපළ කළ යුතුයි.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. .venv/ ignore කරන්න.
2. db.sqlite3 ignore කරන්න.
3. .env සහ secret keys ignore කරන්න.

VS CODE / .gitignore

```
# Local-only files and secrets
.venv/
__pycache__/
*.py[cod]
db.sqlite3
.env
staticfiles/
.idea/
.vscode/
*.log
.DS_Store
```

පුංචි අභ්‍යාසය · TRY IT

git status එකේ db.sqlite3 නැති බව පරීක්ෂා කරන්න.

> **Expected result:** User data accidental push වීමේ අවදානම අඩු වෙයි.

Deployment කියන්නේ මොකක්ද?

Localhost යනු ඔබේ computer එකේම server එකයි. Real users ට ලබාදීමට production hosting සහ HTTPS අවශ්‍යයි. Django runserver production සඳහා නොවේ.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. Production web server/WSGI හෝ ASGI තෝරන්න.
2. DEBUG=False සහ SECRET_KEY env setting.
3. HTTPS, static files, ALLOWED_HOSTS සකස් කරන්න.

VS CODE / Windows CMD

```
python manage.py check --deploy
python manage.py collectstatic
```

පුංචි අභ්‍යාසය · TRY IT

Local runserver සහ production server අතර වෙනස පැහැදිලි කරන්න.

> **Expected result:** Production security + operational setup අවශ්‍යයි.

Database Backup සහ Restore

වැඩ කරන database එකක් අහම්බෙන් නැති වුණොත් backup අවශ්‍යයි. SQLite file copy කිරීමේදී app එක වසා හෝ safe online backup ක්‍රමයක් භාවිතා කරන්න.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. Testing data එකකට backup plan එකක් හදන්න.
2. Backup file එක වෙනම location එකක තබන්න.
3. Restore drill එකක් සිදු කරලා තහවුරු කරන්න.

VS CODE / Windows CMD / example

```
# Stop local server before copying SQLite.
copy db.sqlite3 backups\tasks-backup.sqlite3
```

පුංචි අභ්‍යාසය · TRY IT

Backup file එක ඇත්තටම restore කළ හැකි බව පරීක්ෂා කළේද?

> **Expected result:** Backup තිබීම පමණක් නොව restore test වැදගත්.

Live App එකට තව මොනවාද?

මේ project එක learning prototype එකක්. Production එකකට password recovery, email verification, login rate limiting, logs, monitoring, backups සහ server hardening අවශ්‍ය විය හැකියි.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. Production checklist එක කියවන්න.
2. check --deploy run කරන්න.
3. එක් එක් security item එකට owner කෙනෙක් තබන්න.

VS CODE / Settings / example

```
DJANGO_DEBUG=0
DJANGO_ALLOWED_HOSTS=yourdomain.com
# Set DJANGO_SECRET_KEY securely
```

පුංචි අභ්‍යාසය · TRY IT

DEBUG=True පමණක් ඇති server එක internet වෙත දාන්නේ ඇයි අවදානම්?

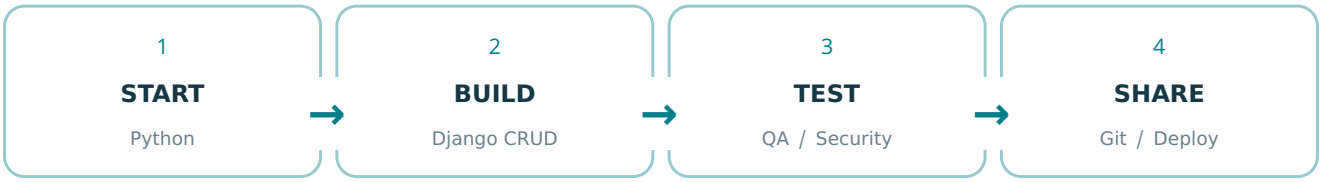
> **Expected result:** Internal errors / debug information leak විය හැකියි.

2026 Django Learning Roadmap

Django 6.1.2 තුළ අපි built-in auth, views, templates, ORM, forms, tests භාවිතා කරනවා. ඉදිරියට REST APIs, background tasks, PostgreSQL, caching සහ secure deployment ඉගෙනගන්න පුළුවන්.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. Beginner: HTML + Django template.
2. Intermediate: APIs + pagination + PostgreSQL.
3. Advanced: async tasks + monitoring + CI/CD.



පුංචි අභ්‍යාසය · TRY IT

ඊළඟ සතියේ ඉගෙනගන්න මාතෘකා තුනක් තෝරන්න.

> **Expected result:** Practical roadmap එකක් ලැබේ.

PRACTICE LABS

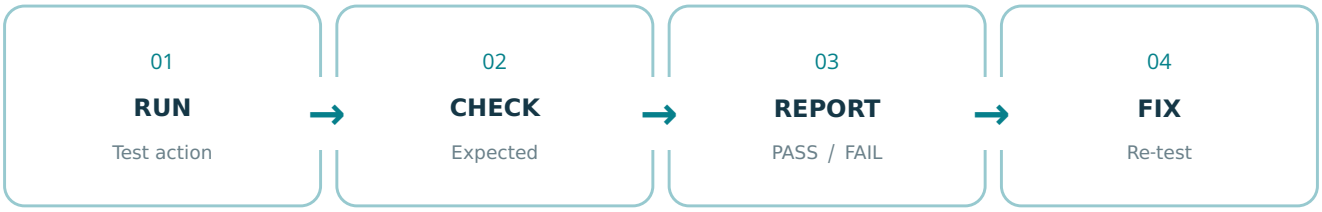
10

Practice & Revision

මේ කොටස තුළ ඉතා සරල උදාහරණ, සත්‍ය source code සහ ඔබටම අත්හදා බැලිය හැකි අභ්‍යාස තිබෙනවා.

මේ කොටසේ පාඩම්

- ✓ Lab 01: First Login
- ✓ Lab 02: Create / Read
- ✓ Lab 03: Update / Delete
- ✓ Lab 04: User Privacy
- ✓ Lab 05: Code Review
- ✓ Final Mini Project Challenge

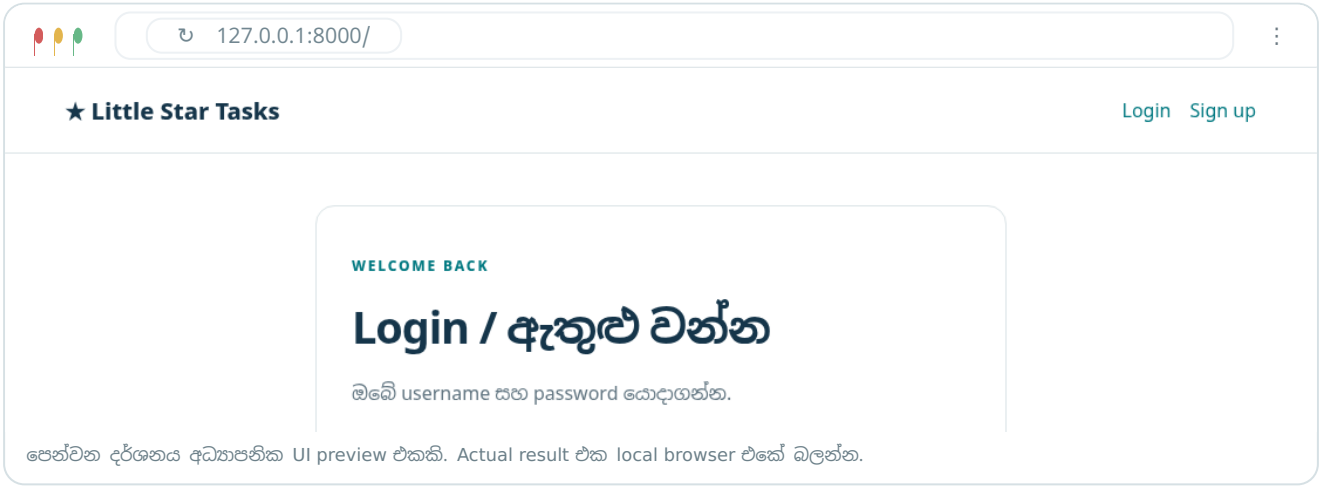


Lab 01: First Login

ලබා දී ඇති ZIP project එක ආරම්භ කර Login සහ Signup pages දෙකම පරීක්ෂා කරන්න. මෙය සාර්ථක වුණොත් පසු පාඩම්වල CRUD තේරුම් ගැනීමට පහසුයි.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. venv + pip + migrate.
2. runserver.
3. Signup → Login → Dashboard.



පෙන්වන දර්ශනය අධ්‍යාපනික UI preview එකකි. Actual result එක local browser එකේ බලන්න.

VS CODE / Windows CMD

```
python -m pip install -r requirements.txt
python manage.py migrate
python manage.py runserver
```

පුංචි අභ්‍යාසය · TRY IT

Screenshot 3ක් ගන්න: Signup, Login, Dashboard.

> **Expected result:** Registration සහ Login pages ක්‍රියාත්මක.

Lab 02: Create / Read

Task දෙකක් එකතු කර list එකේ පෙනෙනවාද බලන්න. එයින් database save සහ query යන දෙකම පරීක්ෂා වෙනවා.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. Task A: Study Django.
2. Task B: Draw a robot.
3. Dashboard එකේ total = 2 බලන්න.



පුංචි අභ්‍යාසය · TRY IT

ඔබේ tasks දෙකෙහි priority වෙනස් කරන්න.

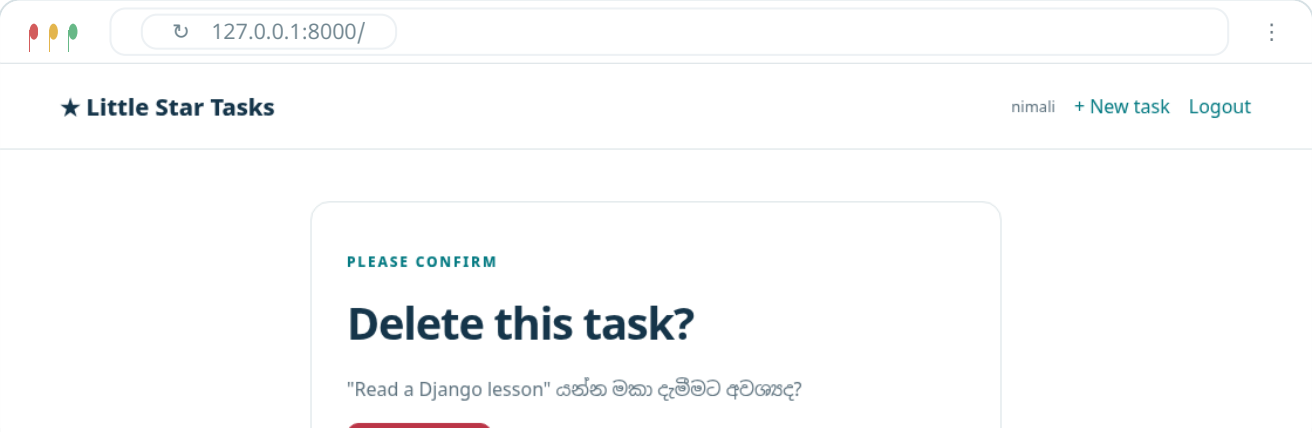
> **Expected result:** Task දෙකම list එකේ පෙනේ.

Lab 03: Update / Delete

Task A එකට due date එකක් සහ Done status යොදා update කරන්න. Task B මකා දමන්න. එයින් Update සහ Delete තහවුරු වෙනවා.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. Task A → Edit → Done.
2. Task B → Delete → Confirm.
3. Total සහ Done counts නැවත බලන්න.



පෙන්වන දර්ශනය අධ්‍යාපනික UI preview එකකි. Actual result එක local browser එකේ බලන්න.

පුංචි අභ්‍යාසය · TRY IT

Expected totals පළමුව ලියා පසුව browser එකෙන් තහවුරු කරන්න.

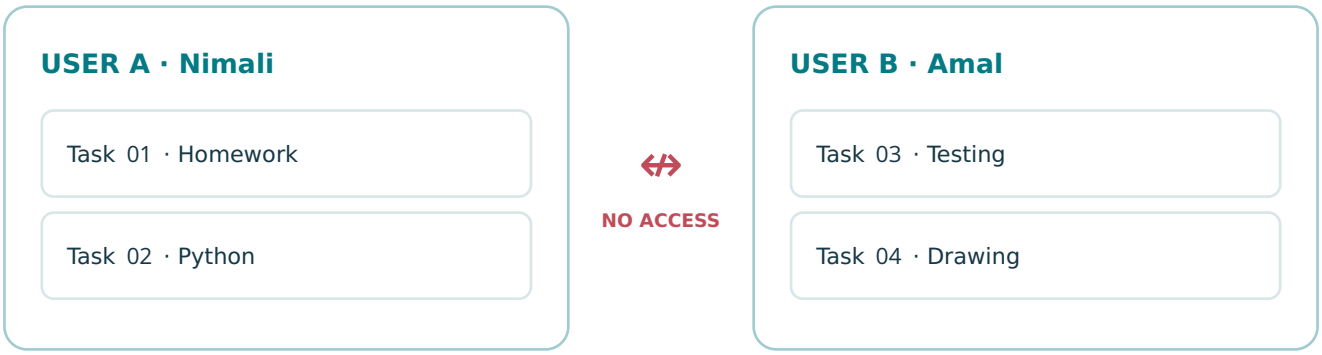
> **Expected result:** Total 1; Done 1.

Lab 04: User Privacy

Accounts දෙකක් හදා එකිනෙකාගේ private tasks පෙනෙන්නේ නැති බව verify කරන්න. Security testing එක වැදගත්ම lab එකක්.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. Nimali account එකට Task X.
2. Amal account එකට Task Y.
3. Amal ගිණුමෙන් Nimaliගේ edit URL try කරන්න.



පුංචි අභ්‍යාසය · TRY IT

Test result එක Expected/Actual ලෙස ලියන්න.

> **Expected result:** Other user's task request = 404.

Lab 05: Code Review

ඔබේ project එකේ security-critical code lines හඳුනාගන්න. Source code කියවීමත් development skill එකක්.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. @login_required සොයාගන්න.
2. owner=request.user දෙතැනක සොයාගන්න.
3. {% csrf_token %} forms තුළ සොයාගන්න.

VS CODE / Review checklist / mixed

```
@login_required
Task.objects.filter(owner=request.user)
{% csrf_token %}
```

පුංචි අභ්‍යාසය · TRY IT

මේ එක් එක් line එක නැති නම් ඇතිවන අවදානම් ලියන්න.

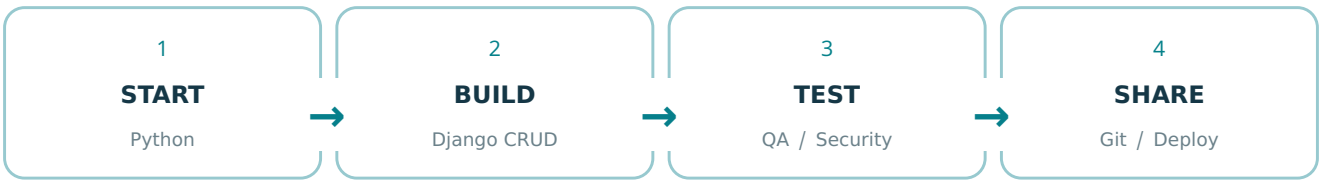
> **Expected result:** Authentication / Authorization / CSRF වෙන වෙනම පැහැදිලි වේ.

Final Mini Project Challenge

දැන් අපේ Task Manager එක පුළුල් කරලා School Homework Tracker එකක් කරන්න. අලුත් UI labels, priority options සහ filter එකක් එක්කර විස්තරාත්මක QA checklist එකක් ලියන්න.

පියවරෙන් පියවර · FOLLOW THESE STEPS

1. Project එක Git repository එකකට save කරන්න.
2. Task label Homework ලෙස වෙනස් කරන්න.
3. දින 7ක development plan එකක් හදන්න.
4. Tests නැවත run කරන්න.



පුංචි අභ්‍යාසය · TRY IT

Feature, Screenshot, Test Result ඇතුළත් report එකක් සකස් කරන්න.

> **Expected result:** ඔබේම customization සහ documentation.

QUIZ 01 / KNOWLEDGE CHECK

Python & Django – ඔබට මතකද?

මුලින්ම පිළිතුරු බලන්නේ නැතිව ඔබටම ලියන්න. පසුව පිටුවේ පහළ answer key එක සමඟ ගළපන්න.

01. Python වලින් web app හදන්න භාවිතා කරන framework එක කුමක්ද?

02. print("Hello") මඟින් සිදුවන්නේ?

03. True / False යන data type එක?

04. Function එකක් හදන Python keyword එක?

පිළිතුරු · ANSWER KEY

1. Django | 2. Terminal එකේ text පෙන්වීම | 3. Boolean | 4. def

> සියල්ල නිවැරදි නොවුණත් අදාළ පාඩමට ආපසු ගොස් කුඩා code එක නැවත run කරන්න.

QUIZ 02 / KNOWLEDGE CHECK

URLs, Views & Templates – ඔබට මතකද?

මුලින්ම පිළිතුරු බලන්නේ නැතිව ඔබටම ලියන්න. පසුව පිටුවේ පහළ answer key එක සමඟ ගළපන්න.

01. Browser URL එක view එකකට සම්බන්ධ කරන්නේ කුමක්ද?

02. HTML file එකට data ලබා දෙන්නේ කුමන layer එකද?

03. CSS files සාමාන්‍යයෙන් තබන්නේ කොහේද?

04. POST form එකක අවශ්‍ය security token එක?

පිළිතුරු · ANSWER KEY

1. urls.py | 2. view | 3. static/ | 4. csrf_token

> සියල්ල නිවැරදි නොවුණත් අදාළ පාඩමට ආපසු ගොස් කුඩා code එක නැවත run කරන්න.

QUIZ 03 / KNOWLEDGE CHECK

Database & CRUD – ඔබට මතකද?

මුලින්ම පිළිතුරු බලන්නේ නැතිව ඔබටම ලියන්න. පසුව පිටුවේ පහළ answer key එක සමඟ ගළපන්න.

01. Task model එකේ user relation field එක කුමක්ද?

02. Database schema වෙනස් කිරීමට commands දෙක?

03. CRUD හි U කියන්නේ කුමක්ද?

04. DELETE එක GET හෝ POST කුමන method එකෙන්ද?

පිළිතුරු · ANSWER KEY

1. owner | 2. makemigrations / migrate | 3. Update | 4. POST

> සියල්ල නිවැරදි නොවුණත් අදාළ පාඩමට ආපසු ගොස් කුඩා code එක නැවත run කරන්න.

QUIZ 04 / KNOWLEDGE CHECK

Security & QA – ඔබට මතකද?

මුලින්ම පිළිතුරු බලන්නේ නැතිව ඔබටම ලියන්න. පසුව පිටුවේ පහළ answer key එක සමඟ ගළපන්න.

01. Login වීම තහවුරු කරන security පියවර?

02. වෙනත් user data edit නොකිරීමේ check එක?

03. Django tests run කරන command එක?

04. Production සඳහා DEBUG අගය කුමක්ද?

පිළිතුරු · ANSWER KEY

1. Authentication | 2. Authorization / owner | 3. python manage.py test | 4. False

> සියල්ල නිවැරදි නොවුණත් අදාළ පාඩමට ආපසු ගොස් කුඩා code එක නැවත run කරන්න.

දින 7ක Django Learning Plan

දවසකට මිනිත්තු 45–60ක් වගේ කාලයකින් මේ project එක ඉගෙනගන්න පුළුවන්. තේරුම් නොගත් කොටසට ආපසු යන්න.

DAY 01	Python basics + variables + functions
DAY 02	Python venv + Django + runserver
DAY 03	URLs + Views + Templates + Static CSS
DAY 04	SQLite + Models + Migrations
DAY 05	Signup + Login + Logout + Sessions
DAY 06	CRUD + Search + UI testing
DAY 07	QA + Security + Git + review

අවසන් ඉලක්කය

ඔබේම account එකක් සකස් කර tasks add/edit/delete කරන, වෙනත් usersගේ data නොපෙන්වන Little Star Task Manager එකක් local browser එකේ ධාවනය කරන්න.

THE REAL FILES USED IN YOUR PROJECT

A

සම්පූර්ණ Source Files කියවමු

මෙතැන් සිට පෙන්වන්නේ ලබා දෙන ZIP project එකේ සැබෑ files වල code කොටස්. මුළු code එක copy/edit කිරීමට source ZIP එක භාවිතා කරන්න.

EXPLORER · LITTLE-STAR-TASKS

```
▾ little_star_tasks/
      ├── manage.py
      ├── little_star_tasks/
      ├── tasks/
      ├── templates/
      ├── static/css/
      └── requirements.txt
```

වැදගත්

පොතේ source appendix එක review සඳහා. Install කරන project එකේ files දැනටමත් තිබෙන නිසා එකම lines දෙවරක් paste කරන්න එපා.

manage.py

LINES 1–9

manage.py

මෙම code කොටස ඔබේ project ZIP එකේ **manage.py** file එකෙන් ලබාගත් සැබෑ කොටසකි.

VS CODE / manage.py

```
#!/usr/bin/env python
"""Django command-line utility."""
import os
import sys

if __name__ == "__main__":
    os.environ.setdefault("DJANGO_SETTINGS_MODULE", "little_star_tasks.settings")
    from django.core.management import execute_from_command_line
    execute_from_command_line(sys.argv)
```

ඔබේ code reading task

මෙම කොටසේ class, function, URL, form field, template tag හෝ test assertion එකක් හඳුනාගන්න. එය ක්‍රියාකරන වැඩේ සිංහලෙන් වාක්‍යයක් ලියන්න.

little_star_tasks/settings.py

LINES 1–22

settings.py

මෙම code කොටස ඔබේ project ZIP එකේ **little_star_tasks/settings.py** file එකෙන් ලබාගත් සැබෑ කොටසකි.

VS CODE / little_star_tasks/settings.py

```
"""Local tutorial settings for Little Star Tasks (Django 6.1)."""
import os
from pathlib import Path

BASE_DIR = Path(__file__).resolve().parent.parent
# DEVELOPMENT ONLY. Provide a strong secret through the environment for production.
SECRET_KEY = os.getenv("DJANGO_SECRET_KEY", "django-insecure-local-tutorial-only-change-me")
DEBUG = os.getenv("DJANGO_DEBUG", "1") == "1"
ALLOWED_HOSTS = [host.strip() for host in os.getenv(
    "DJANGO_ALLOWED_HOSTS", "127.0.0.1,localhost"
).split(",") if host.strip()]

INSTALLED_APPS = [
    "django.contrib.admin",
    "django.contrib.auth",
    "django.contrib.contenttypes",
    "django.contrib.sessions",
    "django.contrib.messages",
    "django.contrib.staticfiles",
    "tasks.apps.TasksConfig",
]
```

ඔබේ code reading task

මෙම කොටසේ class, function, URL, form field, template tag හෝ test assertion එකක් හඳුනාගන්න. එය ක්‍රියාකරන වැඩේ සිංහලෙන් වාක්‍යයක් ලියන්න.

little_star_tasks/settings.py

LINES 23–44

settings.py

මෙම code කොටස ඔබේ project ZIP එකේ **little_star_tasks/settings.py** file එකෙන් ලබාගත් සැබෑ කොටසකි.

VS CODE / little_star_tasks/settings.py

```
MIDDLEWARE = [
    "django.middleware.security.SecurityMiddleware",
    "django.contrib.sessions.middleware.SessionMiddleware",
    "django.middleware.common.CommonMiddleware",
    "django.middleware.csrf.CsrfViewMiddleware",
    "django.contrib.auth.middleware.AuthenticationMiddleware",
    "django.contrib.messages.middleware.MessageMiddleware",
    "django.middleware.clickjacking.XFrameOptionsMiddleware",
]

ROOT_URLCONF = "little_star_tasks.urls"
TEMPLATES = [
    {
        "BACKEND": "django.template.backends.django.DjangoTemplates",
        "DIRS": [BASE_DIR / "templates"],
        "APP_DIRS": True,
        "OPTIONS": {
            "context_processors": [
                "django.template.context_processors.debug",
                "django.template.context_processors.request",
                "django.contrib.auth.context_processors.auth",
                "django.contrib.messages.context_processors.messages",
```

ඔබේ code reading task

මෙම කොටසේ class, function, URL, form field, template tag හෝ test assertion එකක් හඳුනාගන්න. එය ක්‍රියාකරන වැඩේ සිංහලෙන් වාක්‍යයක් ලියන්න.

little_star_tasks/settings.py

LINES 45–66

settings.py

මෙම code කොටස ඔබේ project ZIP එකේ **little_star_tasks/settings.py** file එකෙන් ලබාගත් සැබෑ කොටසකි.

VS CODE / little_star_tasks/settings.py

```
            ],
        },
    },
]
WSGI_APPLICATION = "little_star_tasks.wsgi.application"

DATABASES = {
    "default": {"ENGINE": "django.db.backends.sqlite3", "NAME": BASE_DIR / "db.sqlite3"}
}

AUTH_PASSWORD_VALIDATORS = [
    {"NAME": "django.contrib.auth.password_validation.UserAttributeSimilarityValidator"},
    {"NAME": "django.contrib.auth.password_validation.MinimumLengthValidator"},
    {"NAME": "django.contrib.auth.password_validation.CommonPasswordValidator"},
    {"NAME": "django.contrib.auth.password_validation.NumericPasswordValidator"},
]

LANGUAGE_CODE = "en-us"
TIME_ZONE = "Asia/Colombo"
USE_I18N = True
USE_TZ = True
STATIC_URL = "static/"
```

ඔබේ code reading task

මෙම කොටසේ class, function, URL, form field, template tag හෝ test assertion එකක් හඳුනාගන්න. එය ක්‍රියාකරන වැඩේ සිංහලෙන් වාක්‍යයක් ලියන්න.

little_star_tasks/settings.py

LINES 67–75

settings.py

මෙම code කොටස ඔබේ project ZIP එකේ **little_star_tasks/settings.py** file එකෙන් ලබාගත් සැබෑ කොටසකි.

VS CODE / little_star_tasks/settings.py

```
STATIC_ROOT = BASE_DIR / "staticfiles"
STATICFILES_DIRS = [BASE_DIR / "static"]
DEFAULT_AUTO_FIELD = "django.db.models.BigAutoField"
LOGIN_URL = "login"
LOGIN_REDIRECT_URL = "task_list"
LOGOUT_REDIRECT_URL = "login"

# For production, set DEBUG=False and use the deployment checklist.
# Review SESSION_COOKIE_SECURE / CSRF_COOKIE_SECURE after enabling HTTPS.
```

ඔබේ code reading task

මෙම කොටසේ class, function, URL, form field, template tag හෝ test assertion එකක් හඳුනාගන්න. එය ක්‍රියාකරන වැඩේ සිංහලෙන් වාක්‍යයක් ලියන්න.

little_star_tasks/urls.py

LINES 1–16

urls.py

මෙම code කොටස ඔබේ project ZIP එකේ **little_star_tasks/urls.py** file එකෙන් ලබාගත් සැබෑ කොටසකි.

VS CODE / little_star_tasks/urls.py

```python
"""Project-level URLs."""
from django.contrib import admin
from django.contrib.auth import views as auth_views
from django.urls import include, path
from tasks.views import signup

urlpatterns = [
    path("admin/", admin.site.urls),
    path("accounts/signup/", signup, name="signup"),
    path("accounts/login/", auth_views.LoginView.as_view(
        template_name="registration/login.html"
    ), name="login"),
    # Django 6.1's LogoutView logs out on POST only.
    path("accounts/logout/", auth_views.LogoutView.as_view(), name="logout"),
    path("", include("tasks.urls")),
]
```

ඔබේ code reading task

මෙම කොටසේ class, function, URL, form field, template tag හෝ test assertion එකක් හඳුනාගන්න. එය ක්‍රියාකරන වැඩේ සිංහලෙන් වාක්‍යයක් ලියන්න.

tasks/models.py

LINES 1–22

models.py

මෙම code කොටස ඔබේ project ZIP එකේ **tasks/models.py** file එකෙන් ලබාගත් සැබෑ කොටසකි.

VS CODE / tasks/models.py

```python
"""A task belongs to exactly one signed-in user."""
from django.conf import settings
from django.db import models

class Task(models.Model):
    class Priority(models.TextChoices):
        LOW = "low", "Low"
        MEDIUM = "medium", "Medium"
        HIGH = "high", "High"

    owner = models.ForeignKey(
        settings.AUTH_USER_MODEL,
        on_delete=models.CASCADE,
        related_name="tasks",
    )
    title = models.CharField(max_length=120)
    description = models.TextField(blank=True)
    priority = models.CharField(
        max_length=6, choices=Priority.choices, default=Priority.MEDIUM
    )
    due_date = models.DateField(blank=True, null=True)
    is_done = models.BooleanField(default=False)
```

ඔබේ code reading task

මෙම කොටසේ class, function, URL, form field, template tag හෝ test assertion එකක් හඳුනාගන්න. එය ක්‍රියාකරන වැඩේ සිංහලෙන් වාක්‍යයක් ලියන්න.

tasks/models.py

LINES 23–30

models.py

මෙම code කොටස ඔබේ project ZIP එකේ **tasks/models.py** file එකෙන් ලබාගත් සැබෑ කොටසකි.

VS CODE / tasks/models.py

```
    created_at = models.DateTimeField(auto_now_add=True)
    updated_at = models.DateTimeField(auto_now=True)

    class Meta:
        ordering = ["-created_at"]

    def __str__(self):
        return self.title
```

ඔබේ code reading task

මෙම කොටසේ class, function, URL, form field, template tag හෝ test assertion එකක් හඳුනාගන්න. එය ක්‍රියාකරන වැඩේ සිංහලෙන් වාක්‍යයක් ලියන්න.

tasks/forms.py

LINES 1–13

forms.py

මෙම code කොටස ඔබේ project ZIP එකේ **tasks/forms.py** file එකෙන් ලබාගත් සැබෑ කොටසකි.

VS CODE / tasks/forms.py

```
"""Validate user input with Django ModelForm."""
from django import forms
from .models import Task

class TaskForm(forms.ModelForm):
    class Meta:
        model = Task
        fields = ["title", "description", "priority", "due_date", "is_done"]
        widgets = {
            "title": forms.TextInput(attrs={"placeholder": "e.g. Read a Django lesson"}),
            "description": forms.Textarea(attrs={"rows": 3}),
            "due_date": forms.DateInput(attrs={"type": "date"}),
        }
```

ඔබේ code reading task

මෙම කොටසේ class, function, URL, form field, template tag හෝ test assertion එකක් හඳුනාගන්න. එය ක්‍රියාකරන වැඩේ සිංහලෙන් වාක්‍යයක් ලියන්න.

tasks/views.py

LINES 1–22

views.py

මෙම code කොටස ඔබේ project ZIP එකේ **tasks/views.py** file එකෙන් ලබාගත් සැබෑ කොටසකි.

VS CODE / tasks/views.py

```
"""Login, registration, and owner-protected task CRUD views."""
from django.contrib import messages
from django.contrib.auth import login
from django.contrib.auth.decorators import login_required
from django.contrib.auth.forms import UserCreationForm
from django.db.models import Q
from django.shortcuts import get_object_or_404, redirect, render
from django.views.decorators.http import require_http_methods
from .forms import TaskForm
from .models import Task

@require_http_methods(["GET", "POST"])
def signup(request):
    if request.user.is_authenticated:
        return redirect("task_list")
    form = UserCreationForm(request.POST or None)
    if request.method == "POST" and form.is_valid():
        user = form.save()  # Django hashes the password.
        login(request, user)
        messages.success(request, "Welcome to Little Star Tasks!")
        return redirect("task_list")
    return render(request, "registration/signup.html", {"form": form})
```

ඔබේ code reading task

මෙම කොටසේ class, function, URL, form field, template tag හෝ test assertion එකක් හඳුනාගන්න. එය ක්‍රියාකරන වැඩේ සිංහලෙන් වාක්‍යයක් ලියන්න.

tasks/views.py

LINES 23–44

views.py

මෙම code කොටස ඔබේ project ZIP එකේ **tasks/views.py** file එකෙන් ලබාගත් සැබෑ කොටසකි.

VS CODE / tasks/views.py

```python
@login_required
def task_list(request):
    # IMPORTANT: filter by logged-in owner before search/filtering.
    owned = Task.objects.filter(owner=request.user)
    q = request.GET.get("q", "").strip()[:100]
    status = request.GET.get("status", "all")
    tasks = owned
    if q:
        tasks = tasks.filter(Q(title__icontains=q) | Q(description__icontains=q))
    if status == "open":
        tasks = tasks.filter(is_done=False)
    elif status == "done":
        tasks = tasks.filter(is_done=True)
    return render(request, "tasks/list.html", {
        "tasks": tasks,
        "q": q,
        "status": status,
        "total_count": owned.count(),
        "done_count": owned.filter(is_done=True).count(),
    })
```

ඔබේ code reading task

මෙම කොටසේ class, function, URL, form field, template tag හෝ test assertion එකක් හඳුනාගන්න. එය ක්‍රියාකරන වැඩේ සිංහලෙන් වාක්‍යයක් ලියන්න.

tasks/views.py

LINES 45–66

views.py

මෙම code කොටස ඔබේ project ZIP එකේ **tasks/views.py** file එකෙන් ලබාගත් සැබෑ කොටසකි.

VS CODE / tasks/views.py

```
@login_required
def task_create(request):
    form = TaskForm(request.POST or None)
    if request.method == "POST" and form.is_valid():
        task = form.save(commit=False)
        task.owner = request.user  # NEVER trust an owner field from the browser.
        task.save()
        messages.success(request, "Task saved.")
        return redirect("task_list")
    return render(request, "tasks/form.html", {"form": form, "mode": "Create"})

@login_required
def task_update(request, pk):
    task = get_object_or_404(Task, pk=pk, owner=request.user)
    form = TaskForm(request.POST or None, instance=task)
    if request.method == "POST" and form.is_valid():
        form.save()
        messages.success(request, "Task updated.")
        return redirect("task_list")
    return render(request, "tasks/form.html", {"form": form, "mode": "Edit"})

@login_required
```

ඔබේ code reading task

මෙම කොටසේ class, function, URL, form field, template tag හෝ test assertion එකක් හඳුනාගන්න. එය ක්‍රියාකරන වැඩේ සිංහලෙන් වාක්‍යයක් ලියන්න.

tasks/views.py

LINES 67–75

views.py

මෙම code කොටස ඔබේ project ZIP එකේ **tasks/views.py** file එකෙන් ලබාගත් සැබෑ කොටසකි.

VS CODE / tasks/views.py

```
@require_http_methods(["GET", "POST"])
def task_delete(request, pk):
    task = get_object_or_404(Task, pk=pk, owner=request.user)
    if request.method == "POST":
        task.delete()
        messages.success(request, "Task deleted.")
        return redirect("task_list")
    # GET shows confirmation; it never deletes data.
    return render(request, "tasks/confirm_delete.html", {"task": task})
```

ඔබේ code reading task

මෙම කොටසේ class, function, URL, form field, template tag හෝ test assertion එකක් හඳුනාගන්න. එය ක්‍රියාකරන වැඩේ සිංහලෙන් වාක්‍යයක් ලියන්න.

tasks/urls.py

LINES 1–10

urls.py

මෙම code කොටස ඔබේ project ZIP එකේ **tasks/urls.py** file එකෙන් ලබාගත් සැබෑ කොටසකි.

VS CODE / tasks/urls.py

```python
"""URL names keep templates independent of hard-coded paths."""
from django.urls import path
from . import views

urlpatterns = [
    path("", views.task_list, name="task_list"),
    path("tasks/new/", views.task_create, name="task_create"),
    path("tasks/<int:pk>/edit/", views.task_update, name="task_update"),
    path("tasks/<int:pk>/delete/", views.task_delete, name="task_delete"),
]
```

ඔබේ code reading task

මෙම කොටසේ class, function, URL, form field, template tag හෝ test assertion එකක් හඳුනාගන්න. එය ක්‍රියාකරන වැඩේ සිංහලෙන් වාක්‍යයක් ලියන්න.

templates/base.html

LINES 1–22

base.html

මෙම code කොටස ඔබේ project ZIP එකේ **templates/base.html** file එකෙන් ලබාගත් සැබෑ කොටසකි.

VS CODE / templates/base.html

```
{% load static %}
<!doctype html>
<html lang="si">
<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <title>{% block title %}Little Star Tasks{% endblock %}</title>
  <link rel="stylesheet" href="{% static 'css/site.css' %}">
</head>
<body>
  <header class="site-header">
    <div class="wrap topbar">
      <a class="brand" href="{% url 'task_list' %}">★ Little Star Tasks</a>
      <nav aria-label="Main navigation">
        {% if user.is_authenticated %}
          <span class="username">{{ user.username }}</span>
          <a href="{% url 'task_create' %}">+ New task</a>
          <form method="post" action="{% url 'logout' %}" class="inline-form">
            {% csrf_token %}
            <button class="link-button" type="submit">Logout</button>
          </form>
        {% else %}
```

ඔබේ code reading task

මෙම කොටසේ class, function, URL, form field, template tag හෝ test assertion එකක් හඳුනාගන්න. එය ක්‍රියාකරන වැඩේ සිංහලෙන් වාක්‍යයක් ලියන්න.

templates/base.html

LINES 23–37

base.html

මෙම code කොටස ඔබේ project ZIP එකේ **templates/base.html** file එකෙන් ලබාගත් සැබෑ කොටසකි.

VS CODE / templates/base.html

```
          <a href="{% url 'login' %}">Login</a>
          <a href="{% url 'signup' %}">Sign up</a>
        {% endif %}
      </nav>
    </div>
  </header>
  <main class="wrap main-space">
    {% if messages %}
      {% for message in messages %}<div class="notice" role="status">{{ message }}</div>{% endfor %}
    {% endif %}
    {% block content %}{% endblock %}
  </main>
  <footer class="footer"><div class="wrap">Little Star learning project · egotechworld.com</div></foo
ter>
</body>
</html>
```

ඔබේ code reading task

මෙම කොටසේ class, function, URL, form field, template tag හෝ test assertion එකක් හඳුනාගන්න. එය ක්‍රියාකරන වැඩේ සිංහලෙන් වාක්‍යයක් ලියන්න.

templates/registration/login.html

LINES 1–20

login.html

මෙම code කොටස ඔබේ project ZIP එකේ **templates/registration/login.html** file එකෙන් ලබාගත් සැබෑ කොටසකි.

VS CODE / templates/registration/login.html

```
{% extends 'base.html' %}
{% block title %}Login - Little Star Tasks{% endblock %}
{% block content %}
<section class="panel narrow">
  <p class="kicker">WELCOME BACK</p>
  <h1>Login / ඇතුළු වන්න</h1>
  <p>ඔබේ username සහ password යොදන්න.</p>
  {% if form.errors %}<div class="error">Username හෝ password වැරදියි.</div>{% endif %}
  <form method="post">
    {% csrf_token %}
    <label for="id_username">Username</label>
    {{ form.username }}
    <label for="id_password">Password</label>
    {{ form.password }}
    <input type="hidden" name="next" value="{{ next }}">
    <button type="submit" class="primary">Login</button>
  </form>
  <p>අලුත් ගිණුමක්? <a href="{% url 'signup' %}">Sign up</a></p>
</section>
{% endblock %}
```

ඔබේ code reading task

මෙම කොටසේ class, function, URL, form field, template tag හෝ test assertion එකක් හඳුනාගන්න. එය ක්‍රියාකරන වැඩේ සිංහලෙන් වාක්‍යයක් ලියන්න.

templates/registration/signup.html

LINES 1–19

signup.html

මෙම code කොටස ඔබේ project ZIP එකේ **templates/registration/signup.html** file එකෙන් ලබාගත් සැබෑ කොටසකි.

VS CODE / templates/registration/signup.html

```
{% extends 'base.html' %}
{% block title %}Sign up - Little Star Tasks{% endblock %}
{% block content %}
<section class="panel narrow">
  <p class="kicker">START HERE</p>
  <h1>Create account / ලියාපදිංචිය</h1>
  <form method="post">
    {% csrf_token %}
    {% for field in form %}
      <label for="{{ field.id_for_label }}">{{ field.label }}</label>
      {{ field }}
      {% if field.help_text %}<small>{{ field.help_text|safe }}</small>{% endif %}
      {% for error in field.errors %}<div class="error">{{ error }}</div>{% endfor %}
    {% endfor %}
    <button type="submit" class="primary">Create account</button>
  </form>
  <p>ගිණුමක් තිබෙද? <a href="{% url 'login' %}">Login</a></p>
</section>
{% endblock %}
```

ඔබේ code reading task

මෙම කොටසේ class, function, URL, form field, template tag හෝ test assertion එකක් හඳුනාගන්න. එය ක්‍රියාකරන වැඩේ සිංහලෙන් වාක්‍යයක් ලියන්න.

templates/tasks/list.html

LINES 1–22

list.html

මෙම code කොටස ඔබේ project ZIP එකේ **templates/tasks/list.html** file එකෙන් ලබාගත් සැබෑ කොටසකි.

VS CODE / templates/tasks/list.html

```
{% extends 'base.html' %}
{% block title %}My tasks - Little Star{% endblock %}
{% block content %}
<div class="heading-row">
  <div>
    <p class="kicker">MY WORKSPACE</p>
    <h1>My tasks / මගේ වැඩ</h1>
    <p>ඔබ සෑදූ tasks පමණක් මෙහි පෙනේ.</p>
  </div>
  <a class="primary button-link" href="{% url 'task_create' %}">+ Add task</a>
</div>
<div class="stats">
  <article><strong>{{ total_count }}</strong><span>Total tasks</span></article>
  <article><strong>{{ done_count }}</strong><span>Done</span></article>
</div>
<form class="filters" method="get">
  <label for="q">Search tasks</label>
  <input id="q" name="q" type="search" value="{{ q }}" placeholder="Search title or notes">
  <label for="status">Status</label>
  <select id="status" name="status">
    <option value="all" {% if status == 'all' %}selected{% endif %}>All</option>
    <option value="open" {% if status == 'open' %}selected{% endif %}>Open</option>
```

ඔබේ code reading task

මෙම කොටසේ class, function, URL, form field, template tag හෝ test assertion එකක් හඳුනාගන්න. එය ක්‍රියාකරන වැඩේ සිංහලෙන් වාක්‍යයක් ලියන්න.

templates/tasks/list.html

LINES 23–37

list.html

මෙම code කොටස ඔබේ project ZIP එකේ **templates/tasks/list.html** file එකෙන් ලබාගත් සැබෑ කොටසකි.

VS CODE / templates/tasks/list.html

```
    <option value="done" {% if status == 'done' %}selected{% endif %}>Done</option>
  </select>
  <button type="submit">Filter</button>
</form>
<div class="task-list">
  {% for task in tasks %}
    <article class="task-card">
      <div><h2>{{ task.title }}</h2><p>{{ task.description|default:'No description yet' }}</p>
      <small>{{ task.get_priority_display }} priority · {% if task.is_done %}Done{% else %}Open{% end
if %}{% if task.due_date %} · Due {{ task.due_date }}{% endif %}</small></div>
      <div class="actions"><a href="{% url 'task_update' task.pk %}">Edit</a><a href="{% url 'task_de
lete' task.pk %}">Delete</a></div>
    </article>
  {% empty %}<div class="panel">No tasks yet. Create your first one!</div>
  {% endfor %}
</div>
{% endblock %}
```

ඔබේ code reading task

මෙම කොටසේ class, function, URL, form field, template tag හෝ test assertion එකක් හඳුනාගන්න. එය ක්‍රියාකරන වැඩේ සිංහලෙන් වාක්‍යයක් ලියන්න.

templates/tasks/form.html

LINES 1–19

form.html

මෙම code කොටස ඔබේ project ZIP එකේ **templates/tasks/form.html** file එකෙන් ලබාගත් සැබෑ කොටසකි.

VS CODE / templates/tasks/form.html

```
{% extends 'base.html' %}
{% block title %}{{ mode }} task - Little Star{% endblock %}
{% block content %}
<section class="panel narrow">
  <p class="kicker">TASK EDITOR</p>
  <h1>{{ mode }} task</h1>
  <form method="post">
    {% csrf_token %}
    {% if form.non_field_errors %}<div class="error">{{ form.non_field_errors }}</div>{% endif %}
    {% for field in form %}
      <label for="{{ field.id_for_label }}">{{ field.label }}</label>
      {{ field }}
      {% for error in field.errors %}<div class="error">{{ error }}</div>{% endfor %}
    {% endfor %}
    <button type="submit" class="primary">Save task</button>
    <a class="cancel" href="{% url 'task_list' %}">Cancel</a>
  </form>
</section>
{% endblock %}
```

ඔබේ code reading task

මෙම කොටසේ class, function, URL, form field, template tag හෝ test assertion එකක් හඳුනාගන්න. එය ක්‍රියාකරන වැඩේ සිංහලෙන් වාක්‍යයක් ලියන්න.

templates/tasks/confirm_delete.html

LINES 1–14

confirm_delete.html

මෙම code කොටස ඔබේ project ZIP එකේ **templates/tasks/confirm_delete.html** file එකෙන් ලබාගත් සැබෑ කොටසකි.

VS CODE / templates/tasks/confirm_delete.html

```
{% extends 'base.html' %}
{% block title %}Delete task - Little Star{% endblock %}
{% block content %}
<section class="panel narrow">
  <p class="kicker">PLEASE CONFIRM</p>
  <h1>Delete this task?</h1>
  <p>"{{ task.title }}" යන්න මකා දැමීමට අවශ්‍යද?</p>
  <form method="post">
    {% csrf_token %}
    <button type="submit" class="danger">Yes, delete</button>
    <a class="cancel" href="{% url 'task_list' %}">No, go back</a>
  </form>
</section>
{% endblock %}
```

ඔබේ code reading task

මෙම කොටසේ class, function, URL, form field, template tag හෝ test assertion එකක් හඳුනාගන්න. එය ක්‍රියාකරන වැඩේ සිංහලෙන් වාක්‍යයක් ලියන්න.

tasks/tests.py

LINES 1–22

tests.py

මෙම code කොටස ඔබේ project ZIP එකේ **tasks/tests.py** file එකෙන් ලබාගත් සැබෑ කොටසකි.

VS CODE / tasks/tests.py

```
"""Run with: py manage.py test"""
from django.contrib.auth import get_user_model
from django.test import Client, TestCase
from django.urls import reverse
from .models import Task

User = get_user_model()

class TaskAppTests(TestCase):
    def setUp(self):
        self.alex = User.objects.create_user(username="alex", password="SafePass123!abc")
        self.maya = User.objects.create_user(username="maya", password="SafePass123!abc")
        self.a_task = Task.objects.create(owner=self.alex, title="Alex private task")
        self.m_task = Task.objects.create(owner=self.maya, title="Maya secret task")

    def test_login_required(self):
        response = self.client.get(reverse("task_list"))
        self.assertEqual(response.status_code, 302)
        self.assertIn("/accounts/login/", response.url)

    def test_login_success(self):
        response = self.client.post(reverse("login"), {
```

ඔබේ code reading task

මෙම කොටසේ class, function, URL, form field, template tag හෝ test assertion එකක් හඳුනාගන්න. එය ක්‍රියාකරන වැඩේ සිංහලෙන් වාක්‍යයක් ලියන්න.

tasks/tests.py

LINES 23–44

tests.py

මෙම code කොටස ඔබේ project ZIP එකේ **tasks/tests.py** file එකෙන් ලබාගත් සැබෑ කොටසකි.

VS CODE / tasks/tests.py

```
            "username": "alex", "password": "SafePass123!abc"
        })
        self.assertRedirects(response, reverse("task_list"))

    def test_login_rejects_wrong_password(self):
        response = self.client.post(reverse("login"), {
            "username": "alex", "password": "wrong"
        })
        self.assertEqual(response.status_code, 200)

    def test_signup(self):
        response = self.client.post(reverse("signup"), {
            "username": "nimal", "password1": "MuchSaferPass123!",
            "password2": "MuchSaferPass123!"
        })
        self.assertRedirects(response, reverse("task_list"))
        self.assertTrue(User.objects.filter(username="nimal").exists())

    def test_list_is_private(self):
        self.client.force_login(self.alex)
        content = self.client.get(reverse("task_list")).content.decode()
        self.assertIn("Alex private task", content)
```

ඔබේ code reading task

මෙම කොටසේ class, function, URL, form field, template tag හෝ test assertion එකක් හඳුනාගන්න. එය ක්‍රියාකරන වැඩේ සිංහලෙන් වාක්‍යයක් ලියන්න.

tasks/tests.py

LINES 45–66

tests.py

මෙම code කොටස ඔබේ project ZIP එකේ **tasks/tests.py** file එකෙන් ලබාගත් සැබෑ කොටසකි.

VS CODE / tasks/tests.py

```
        self.assertNotIn("Maya secret task", content)

    def test_create_sets_owner(self):
        self.client.force_login(self.alex)
        self.client.post(reverse("task_create"), {"title": "Read book", "priority": "high"})
        self.assertTrue(Task.objects.filter(owner=self.alex, title="Read book").exists())

    def test_owner_spoof_ignored(self):
        self.client.force_login(self.alex)
        self.client.post(reverse("task_create"), {
            "title": "Private", "owner": self.maya.pk, "priority": "low"
        })
        self.assertEqual(Task.objects.get(title="Private").owner, self.alex)

    def test_empty_title_invalid(self):
        self.client.force_login(self.alex)
        response = self.client.post(reverse("task_create"), {"title": ""})
        self.assertEqual(response.status_code, 200)
        self.assertFalse(Task.objects.filter(title="").exists())

    def test_edit_own(self):
        self.client.force_login(self.alex)
```

ඔබේ code reading task

මෙම කොටසේ class, function, URL, form field, template tag හෝ test assertion එකක් හඳුනාගන්න. එය ක්‍රියාකරන වැඩේ සිංහලෙන් වාක්‍යයක් ලියන්න.

tasks/tests.py

LINES 67–88

tests.py

මෙම code කොටස ඔබේ project ZIP එකේ **tasks/tests.py** file එකෙන් ලබාගත් සැබෑ කොටසකි.

VS CODE / tasks/tests.py

```
        self.client.post(reverse("task_update", args=[self.a_task.pk]), {
            "title": "Updated", "priority": "medium", "is_done": "on"
        })
        self.a_task.refresh_from_db()
        self.assertEqual(self.a_task.title, "Updated")
        self.assertTrue(self.a_task.is_done)

    def test_cannot_edit_other(self):
        self.client.force_login(self.alex)
        response = self.client.post(reverse("task_update", args=[self.m_task.pk]), {
            "title": "Hacked", "priority": "high"
        })
        self.assertEqual(response.status_code, 404)
        self.m_task.refresh_from_db()
        self.assertEqual(self.m_task.title, "Maya secret task")

    def test_get_delete_only_confirms(self):
        self.client.force_login(self.alex)
        response = self.client.get(reverse("task_delete", args=[self.a_task.pk]))
        self.assertEqual(response.status_code, 200)
        self.assertTrue(Task.objects.filter(pk=self.a_task.pk).exists())
```

ඔබේ code reading task

මෙම කොටසේ class, function, URL, form field, template tag හෝ test assertion එකක් හඳුනාගන්න. එය ක්‍රියාකරන වැඩේ සිංහලෙන් වාක්‍යයක් ලියන්න.

tasks/tests.py

LINES 89–110

tests.py

මෙම code කොටස ඔබේ project ZIP එකේ **tasks/tests.py** file එකෙන් ලබාගත් සැබෑ කොටසකි.

VS CODE / tasks/tests.py

```
    def test_post_deletes_own(self):
        self.client.force_login(self.alex)
        self.client.post(reverse("task_delete", args=[self.a_task.pk]))
        self.assertFalse(Task.objects.filter(pk=self.a_task.pk).exists())

    def test_cannot_delete_other(self):
        self.client.force_login(self.alex)
        response = self.client.post(reverse("task_delete", args=[self.m_task.pk]))
        self.assertEqual(response.status_code, 404)
        self.assertTrue(Task.objects.filter(pk=self.m_task.pk).exists())

    def test_search_keeps_owner_scope(self):
        self.client.force_login(self.alex)
        response = self.client.get(reverse("task_list"), {"q": "secret"})
        self.assertNotContains(response, "Maya secret task")

    def test_filter_done(self):
        Task.objects.filter(pk=self.a_task.pk).update(is_done=True)
        self.client.force_login(self.alex)
        response = self.client.get(reverse("task_list"), {"status": "done"})
        self.assertContains(response, "Alex private task")
```

ඔබේ code reading task

මෙම කොටසේ class, function, URL, form field, template tag හෝ test assertion එකක් හඳුනාගන්න. එය ක්‍රියාකරන වැඩේ සිංහලෙන් වාක්‍යයක් ලියන්න.

tasks/tests.py

LINES 111–125

tests.py

මෙම code කොටස ඔබේ project ZIP එකේ **tasks/tests.py** file එකෙන් ලබාගත් සැබෑ කොටසකි.

VS CODE / tasks/tests.py

```
    def test_logout_post(self):
        self.client.force_login(self.alex)
        response = self.client.post(reverse("logout"))
        self.assertRedirects(response, reverse("login"))

    def test_logout_get_not_allowed(self):
        response = self.client.get(reverse("logout"))
        self.assertEqual(response.status_code, 405)

    def test_csrf_protects_delete(self):
        client = Client(enforce_csrf_checks=True)
        client.force_login(self.alex)
        response = client.post(reverse("task_delete", args=[self.a_task.pk]))
        self.assertEqual(response.status_code, 403)
        self.assertTrue(Task.objects.filter(pk=self.a_task.pk).exists())
```

ඔබේ code reading task

මෙම කොටසේ class, function, URL, form field, template tag හෝ test assertion එකක් හඳුනාගන්න. එය ක්‍රියාකරන වැඩේ සිංහලෙන් වාක්‍යයක් ලියන්න.

නිල 2026 මූලාශ්‍ර සහ අවසාන පරීක්ෂාව

Version එක: **Django 6.1.2** (2026-10-06 patch release). Python 3.12/3.13/3.14 supported. Beginner project සඳහා SQLite භාවිතා කරලා ඇත.

Django download: https://www.djangoproject.com/download/

Python versions: https://docs.djangoproject.com/en/6.1/faq/install/

Login / Logout: https://docs.djangoproject.com/en/6.1/topics/auth/default/

Deployment checklist: https://docs.djangoproject.com/en/6.1/howto/deployment/checklist/

Official Django tutorial: https://docs.djangoproject.com/en/6.1/intro/tutorial01/

අවසන් Checklist

☐ Venv active ☐ Django install ☐ Migration success ☐ Signup/Login works ☐ CRUD works ☐ Other-user edit/delete blocked ☐ CSRF present ☐ Tests run ☐ Git ignore secrets.

මෙය ආරම්භක අධ්‍යාපනික project එකකි. Source code සඳහා Python syntax/structure checks සම්පූර්ණ කර ඇත. Authoring environment එකේ Django runtime install නොවූ නිසා database migration, live server සහ Django TestCase execution මෙහි ධාවනය කළ බව කියන්නේ නැහැ. ඔබේ local machine එකේ පරීක්ෂා කරන්න.