



Flask ચીટ-શીટ

FLASK CHEATSHEET · PYTHON WEB FRAMEWORK

BEGINNER FRIENDLY

GENERATED BY AI

AIGUJARATI.IN

Python ની સૌથી સરળ web framework — Flask — ની સંપૂર્ણ ગુજરાતી માર્ગદર્શિકા. Installation થી લઈને Forms, Templates, Static Files અને Debug Mode સુધી — બધું એક જ જગ્યાએ!

વિષય-સૂચી:

1. Flask શું છે?

2. Installation

3. App Setup

4. Routes

5. Templates

6. Forms

7. Static Files

8. Redirect & URL

9. Debug Mode



૧. Flask શું છે? (What is Flask?)

Python ની lightweight web framework

Flask એટલે?

Theory

Flask એ Python ની **micro web framework** છે. તે ખૂબ સરળ અને **lightweight** છે. Django કરતાં ઘણો નાનો, પણ powerful!



ઉપયોગ ક્ષેત્ર:

Web apps, REST APIs, Blogs, Portfolios — બધું Flask થી બની શકે!

Flask ના ફાયદા

Benefits

1

સરળ Syntax

Python જ જાણો, Flask આવડી જ ગઈ!

2

Lightweight

ફક્ત જરૂરી tools, બિનજરૂરી ભાર નહીં

3

Flexible

ગમે ત્યારે extensions add કરી શકાય



Prerequisites

Requirements

Flask શીખવા માટે આ વસ્તુઓ જરૂરી છે:

TOOL	ઉપયોગ
Python 3.x	Programming Language
pip	Package Manager
Terminal	Commands ચલાવવા
VS Code	Code Editor (ઐચિછક)



૨. Installation (ઇન્સ્ટોલેશન)

Flask ને Computer માં install કેવી રીતે કરવું



Step-by-Step Installation

Terminal

1

Virtual Environment બનાવો

Project ને isolate રાખવા માટે

```
# Virtual Environment બનાવો
python -m venv venv

# Activate સરો (Windows)
venv\Scripts\activate

# Activate સરો (Mac/Linux)
source venv/bin/activate
```

```
# Flask install સરો
pip install flask

# Check version
flask --version
```



Project Structure

Structure

Flask project ની ફાઇલ structure આ રીતે હોય:

```
my_project/
├── app.py           # Main file
├── venv/            # Virtual env
├── templates/       # HTML files
│   └── index.html
├── static/          # CSS, JS, Images
│   ├── style.css
│   └── script.js
```



યાદ રાખો:

`templates` અને `static` folders ના નામ ચોક્કસ (exact) આ જ હોવા જોઈએ — Flask automatically detect કરે છે.



3. App Setup (App બનાવવી)

Flask application ની basic setup



Minimum Flask App

app.py

આ સૌથી નાનો Flask program છે — Hello World!

```
python

# Step 1: Flask import કરો
from flask import Flask

# Step 2: App object બનાવો
app = Flask(__name__)

# Step 3: Route define કરો
@app.route('/')
def home():
    return 'નમસ્તે Flask! 🚀'

# Step 4: App run કરો
if __name__ == '__main__':
    app.run()
```



App કેવી રીતે Run કરવી

Run



Terminal માં આ command ચલાવો:

Virtual environment activate હોવું જ જોઈએ!



bash

```
# Method 1: Python સે run
python app.py
```

```
# Method 2: Flask CLI
flask run
```

```
# Browser માં ખોલો:
http://127.0.0.1:5000
```



Running on

http://127.0.0.1:5000 message

આવે એટલે app successfully start થઈ ગઈ!



૪. Routes (રૂટ્સ)

URL ને function સાથે connect કરવું



Routes એટલે શું? અને ઉદાહરણ

Core Concept

Route = URL + Function. જ્યારે user specific URL visit કરે, ત્યારે specific Python function ચાલે.

```
python — Basic Routes

# Home Page → /
@app.route('/')
def home():
    return 'Home Page'

# About Page → /about
@app.route('/about')
def about():
    return 'About Page'

# Contact Page → /contact
@app.route('/contact')
def contact():
    return 'Contact Us'
```

```
python — Dynamic Routes

# Dynamic route — URL માં variable
@app.route('/user/<name>')
def user_profile(name):
    return f'નમસ્તે, {name}!'

# Integer variable
@app.route('/post/<int:post_id>')
def show_post(post_id):
    return f'Post #{post_id}'

# Multiple HTTP Methods
@app.route('/submit',
    methods=['GET', 'POST'])
def submit():
    return 'Form Submitted!'
```

URL PATTERN	ગુજરાતીમાં સ્પષ્ટીકરણ	EXAMPLE VISIT
/	Home page (main page)	localhost:5000/
/about	Static page	localhost:5000/about
/user/<name>	Dynamic — name change થાય	localhost:5000/user/Raj
/post/<int:id>	Integer variable	localhost:5000/post/5



૫. Templates (HTML Pages)

Jinja2 template engine — HTML + Python logic



render_template() — HTML

Return કરો

app.py

String return ના કરો — HTML file render કરો:

```
python
from flask import Flask, render_template

app = Flask(__name__)

@app.route('/')
def home():
    # Variable pass કરો template ને
    name = 'Raj'
    items = ['Python', 'Flask', 'HTML']
    return render_template(
        'index.html',
        name=name,
        items=items
    )
```



Jinja2 Template Syntax

index.html

templates/index.html ફાઇલ આ રીતે લખો:

```
html (Jinja2)

<!-- Variable show કરો -->
<h1>નમસ્તે, {{ name }}!</h1>

<!-- If condition -->
{% if name == 'Raj' %}
    <p>Admin User</p>
{% else %}
    <p>Normal User</p>
{% endif %}

<!-- Loop (list items) -->
{% for item in items %}
    <li>{{ item }}</li>
{% endfor %}
```



Jinja2 Quick Reference

Reference

SYNTAX	ઉપયોગ	EXAMPLE
{{ variable }}	Variable ના value show કરો	{{ name }} → Raj
{% if %} {% endif %}	Condition check કરો	{% if age > 18 %}
{% for %} {% endfor %}	List loop	{% for x in list %}
{% extends %}	Base template inherit	{% extends 'base.html' %}
{% block %} {% endblock %}	Content block define	{% block content %}
{{ variable filter }}	Filter apply	{{ name upper }}



૬. Forms (ફોર્મ)

User input handle કરવું — GET & POST method

HTML Form (Template)

templates/form.html

```
html

<form method="POST"
      action="/submit">

  <!-- Text Input -->
  <input type="text"
        name="username"
        placeholder="નામ લખો">

  <!-- Email Input -->
  <input type="email"
        name="email"
        placeholder="Email">

  <!-- Submit Button -->
  <button type="submit">
    Submit
  </button>

</form>
```

Form Data Handle (Python)

app.py

```
python

from flask import Flask, request, render_template

app = Flask(__name__)

@app.route('/submit',
          methods=['GET', 'POST'])
def submit():
    if request.method == 'POST':
        # Form data મેળવો
        name = request.form['username']
        email = request.form['email']

        return f'નમસ્તે {name}!'

    # GET request - form show
    return render_template('form.html')
```



GET vs POST:

GET = data URL માં (search), POST =
data hide થઈ (login, forms)



9. Static Files (CSS, JS, Images)

Styling અને scripts add કરવું



CSS/JS Link કરવું

HTML Template

`url_for('static', filename='...')` use કરો:

```
html (Jinja2)

<head>
  <!-- CSS file link -->
  <link rel="stylesheet"
        href={{ url_for('static',
                          filename='style.css') }}>
</head>

<body>
  <!-- Image tag -->
  <img src={{ url_for('static',
                      filename='logo.png') }}>

  <!-- JS file link -->
  <script src={{ url_for('static',
                        filename='script.js') }}></script>
</body>
```



Static Folder Structure

Files

```
structure

static/
├── css/
│   └── style.css
├── js/
│   └── script.js
└── images/
    └── logo.png
```



`url_for()` ના ફાયદા:

Hardcoded path ના લખો.

`url_for()` automatically correct path generate કરે — project structure change થાય તો પણ.

FILE TYPE

FILENAME= PARAMETER

CSS

'css/style.css'

JavaScript

'js/script.js'

Image

'images/logo.png'



૮. Redirect & URL (redirect & url_for)

User ને બીજા page પર send કરવું



redirect() ઉપયોગ

app.py

```
python

from flask import Flask, redirect, url_for

app = Flask(__name__)

@app.route('/')
def home():
    # /old-page visit → /new-page send કરવું
    return redirect(url_for('new_page'))

@app.route('/old')
def old_page():
    # Redirect to /new
    return redirect(url_for('new_page'))

@app.route('/new')
def new_page():
    return 'New Page!'
```



url_for() — URL Generate

Best Practice

`url_for()` function name આપે, URL automatically generate!

```
python

# url_for ના ઉપયોગ
url_for('home')
# → '/' generate કરે

url_for('user_profile',
        name='Raj')
# → '/user/Raj' generate કરે

# Template માં
{{ url_for('home') }}
{{ url_for('static',
            filename='style.css') }}
```

FUNCTION

ઉપયોગ

`redirect(url)`

User ને બીજા
URL પર send

`url_for('fn')`

Function
name → URL
generate

`url_for('static',
filename= ...)`

Static file URL



૯. Debug Mode (ડિબગ મોડ)

Development time error tracking



Debug Mode Enable

Development Only

```
python

# Method 1: app.run() માં
if __name__ == '__main__':
    app.run(debug=True)

# Method 2: Config set
app.config['DEBUG'] = True

# Method 3: Environment variable
# Terminal માં:
# export FLASK_DEBUG=1 (Mac)
# set FLASK_DEBUG=1 (Windows)
app.run(host='0.0.0.0',
        port=5000,
        debug=True)
```



Debug Mode — ફાયદા & ચેતવણી

Important



Debug Mode ON = ના ફાયદા:

Code save કરે → App automatically restart. Error messages browser માં. Line-by-line error tracking.



Production Server (Live Website) પર Debug Mode NEVER ON!

Debug mode ON = Security risk. Hackers server information જોઈ શકે. Production માં `debug=False` રાખો.

MODE	DEBUG=	ઉપયોગ
Development	True	Coding time
Production	False	Live website



Quick Reference — Important Imports

Flask ની most used functions

```
python - All Imports

from flask import (
    Flask,           # App create
    render_template, # HTML render
    request,         # Form data
    redirect,        # Redirect
    url_for,         # URL generate
    jsonify,         # JSON response
    session,         # User session
    flash            # Flash messages
)
```

FUNCTION/OBJECT	ગુજરાતી ઉપયોગ
<code>Flask(__name__)</code>	App object બનાવો
<code>render_template()</code>	HTML file return
<code>request.form[]</code>	POST form data
<code>request.args[]</code>	GET URL parameters
<code>request.method</code>	GET/POST check
<code>redirect()</code>	User redirect
<code>url_for()</code>	URL generate
<code>jsonify()</code>	JSON API response
<code>session[]</code>	User login info save
<code>flash()</code>	One-time message show

aigujarati.in

ગુજરાતીમાં Technology શીખો ✨

Generated by AI

Powered by Artificial Intelligence

aigujarati.in

Flask Cheatsheet v1.0

Beginner Friendly · Gujarati

© 2025 aigujarati.in