1) what is Flask?

Flask is a one of the Framework in Python. its a micro level framework.

2) List differences between Flask an Django

Flask	Django.
Micro framework	Full-Stack Framework
Minimal (add manually)	Many features included.
High flexibility	Less flexible due to conventions
user-defined	pre-defined structure
Easy for beginners	Steeper learning curve

Page No.
Date

3) Research and write 3 pros and 3 cons of using Flask?

Pros:
1) Easy to learn and use.
2) Flexible and minimal
3) Great for small apps and APIs

Cons:
1) Not ideal for large complex apps.
2) Fewer built-in tools than Django
3) Requires more manual setup

4) Best use cases For Flask (with Examples)
- Small webapps like a calculator, portfolio site

Page No.
Date

- REST APIs for mobile or frontend apps.
- Prototypes/MVPs (e.g, to-do app)

Example
A startup can use Flask to quickly launch a to-do list app with login and task management features.

5) Flask vs Django - Project setup complexity :-

* Flask has a simple setup Just install Flask and create app.py.
* Django has more boilerplate (project folders, settings, apps) but it's powerful for big apps.

Page No.
Date

6. Role of WSGI in Flask

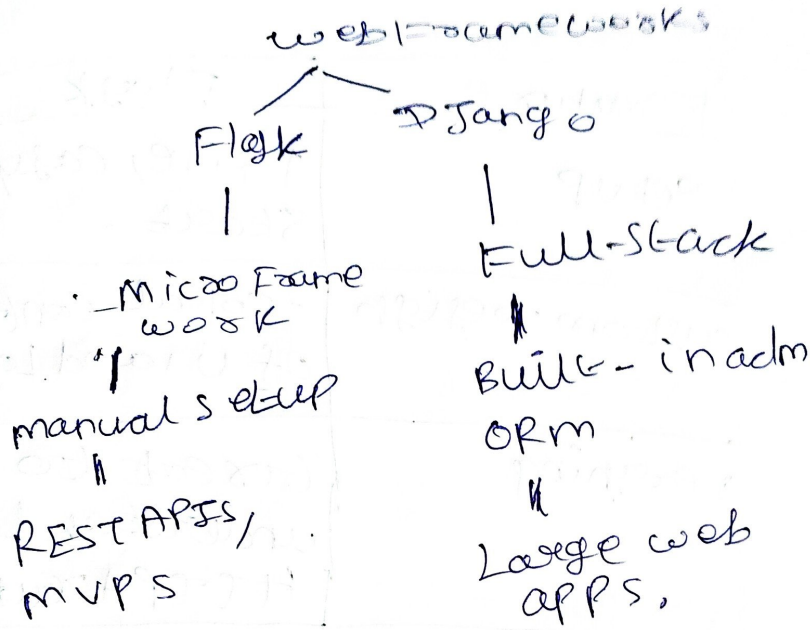
Ans
WSGI (Web Server Gateway Interface) acts as bridge between web servers (like Gunicorn) and your Flask app. It helps the server understand Python code and deliver responses

7) companies using Flask

Company/ Platform	Why Flask used.
Netflix	For internal data tools
Reddit	For lightweight services
Airbnb	For micro services
Lyft	For internal dashboard.

Page No:
Date:

8) mind map - Flask vs Django Architecture.



9) Why Flask is Great for Beginners.

1) simple to install and run
2) Fewer files, esy to understand.
3) Great community + beginner tutorials.
4) Flexible to experiments.
5) Focuses one core concepts for web development.

Page No.
Date

10) Sample use-case :-

TO-DO APP

Feature	Flask
Setup	1 file, easy to start.
custom design	total control of UI and Logic
Learning	Great to understand HTTP + routes
Example	to-do APP with CRUD operations

11) List down 5 popular Flask-related packages :-

1) Flask-WTF
- Adds support for web forms using WTForms

Page No.
Date

- Helps with CSRF protection, validation and rendering.

use
pip install flask-wtf

2) Flask-Login
- manages user Authentication (login / logout / session)
- Helps protect routes and manage logged-in states.

use
pip install flask-login

3) Flask-mail
- send emails from your Flask application.
- useful for registration, password reset etc.

use
pip Install flask-mail

Page No.
Date

4. Flask-SQLAlchemy

- Adds SQLAlchemy (ORM) support to Flask.
- Lets you define and use database easily with Python classes.

use

Pip install flask-sqlalchemy

5. Flask-migrate

- Database migration tool using Alembic.
- Works with Flask-SQLAlchemy to upgrade/downgrade DB schema.

use

Pip install flask-migrate

Page No.
Date

12) bash script.

1) create virtual environment
2) Activate it
3) install Flask
4) Freeze dependencies requirements.txt.

13) Explain debug mode.

* Auto reload - server restarts on code change
* Debug console - shown errors in browser.
* Dev-friendly - Great for testing and learning.

Page No.
Date

14) app.py	flask run
Requires --main--	No
No uses environment vars	yes (FLASK_APP FLASK_ENV)
if debug=True	if FLASK_ENV=development
Not ideal Production setup	Still Not ideal Production setup
Easier to start with	Slightly more setup needed
~~Flas~~ No CLI Features	Yes (flask shell, etc

Page No.
Date

15) common issues faced while running the server (port conflicts, missing env vars).

issue	cause	solution
Port already in use.	Another app uses port 5000	change to port 8000 or higher.
APP not found	missing FLASK_APP env var	set it manually or use .env file.
unreachable from phone	Firewall or wrong host	use host='0.0.0.0' allow in firewall
NO auto-reload	missing debug mode	use debug=True or FLASK_ENV=development