A-11) 12 Factors App:-

The twelve-factor app is a methodology used to
develop saas apps or useh apps with the help of
Histoservices. It serves as a practice guide to building
apps when pressed for time or cost to hire new
developers for a project and make them scale up
without significant changes to tools or architecture.
It allows organizing projects effectively and managing
scalable applications.
This methodology was first introduced by the developer
at Heroku in 2012 who created hundreds of
applies and developed this methodology based on their
experience. The methodology incorporates 12 principles

experience. The methodology incorporates 12 princip each applying to a subcet of an app and guiding clevelopens in finding the best way to manage apps as a whole

The principles it suggests that are not constrained to only particular programming language or database

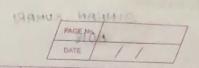
- The 12-factor Principles!-

1. (odebase (one code base tracked in revision control, many deploys)

There should be one codebase with different environments.

The guideline expressed in this first factor suggests the principle of not creating another codebase, just for the cake of setup in different environments, for institute shouldn't be 2 repositories (product 2 develop")

Dy any other days



for one codebase. A distributed version control syscan maintain & monitor the projecture code for developers, this makes a cressibility easier.

Diff. arviron represent diff. states. They all share the same codebase, whatever the date might be.

2. Dependencies (Explicitly declare and isolate the

All dependencies should be declared, with no implicit reliance on sys. tools or libraries.

Including the depen. into the codebase is a bad practice since it could create prob. such as the underlying dependency of the plantorm. For inst, if the uploaded modules were from a windows per perspective, & the codebase was downloaded by a Hac Os developer, they would undoubtedly have prob. running the proj.

It's better to use a package manager-depending on the tech stack-to download the depen. on your respective sys by reading a dep. declarat manifest containing the names & versions of the depen.

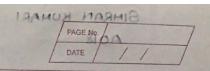
3. config (Store configurate in an environ.)

configurates are an imp part of any apph, especially when there is no a need to support multiple environ or clients.

A configurat that varies bet apployments should be stored in the environ. No one wants configurated ential into to be publicly available, such as database connect into or other sensitive data. The configurat is also large agrendent on the goviron, where it's based.

There should be a strict separate bot config & code. code should remain the same irr. of where the applic is being deployed, but config can vory. 4. Backing services chreat bac ser-resources as attached resources) All backing services are treated as attatched res. 8 attatched 8 detached by the execut environ, Dotabases, external storage , mag. queues, etc, are ext. services & they can be considered as a resource. They may be access so via some web-addresses / URLs or similar req. , then specified in the contig. That way, the sources of the ser can be modified w/s affecting the core of the app. 5. Build, Release, Run (Strictly seperate build & run stages) The 12-fact app restricts req. a strict seperate bet Build, Release 4 Run stages. · First, the build phase. It's the assembly of the sc into an executable bundle while localing depend. & creating assests. The boild phase starts every time a new code needs to be deployed. o 2", the release stage: Here the code produced during the build stage is combined, with the deploys current config. It contains both build & config & o b, the run stage: The final stage runs the appli in the execut environ of should not be intervened by any other stago.

	SINRAN KUMARI AOK
G.	Processes Cerecute the app as 1 or more stateless processes).
ad unua	Applis should be pel deployed as one or more stateless processes with persisted data stored on a backing, service, typically adb.
	The key principle of this factor is that the process is Stateless and shares absolutely nothing. While many web sys rely on sticky sessions storing data is in the session expecting that the next
5000	this approach.



7 Port Binding (Export services via port binding)

A tivelve factor app should be independent of additional application every function should be its own process.

This factor reters to an application binding itself a particular part and responding to all the requests from that part. The part is declared as an environment variable and provided during execution.

Applications built on the basis of principle do not he depend on a web server. The application is self-conting self-continued and hins standalone. The web server to prekaged as a Library and bundled with the application.

8. Concurrency (Scare out via the process moder)

Based on the requirement, each process in your 3 application should be able to scale, restart, or clone itself. Doing this can enhance scalability.

Thatead of making a single process even larger olevelopers can create multiple processes and then distribute the load of their application among those processes. Using this approach, developers can build apporthan can handle volious workloads by assigning each workloads to a process type. For instance, it is possible to delegate HTTP request to a web process and ipng background tasks to the worker process.

9. Disposa bility (maximize the robustness with fast
Startup and graceful snutdown)

A 12-factor app should be built on simple process
that start quickly work quickly and finish
correctly. This helps developers to scale up
processes while still allowing them to restart
them if anything goes wrong.
Building disposability into app means graceful
shutdown. It should clean up all utilized
resources and shut clown seamlessly. An
app can be easily relaunched when designed
in this way likewise when processes terminate
they should finish their current request

10. Dev-Prod Parity (Keep development, staging, and production as similar as possible)

The twelve-factor article proposes keeping the difference between development and production environment minimal. This reduces the likelihood of bugs turning up in a particular environment Developers should strive to use the same third party service between development and production. To keep differences between development and production minimal, teams collaborating on a project should use the same operating system, backing services, and alependencies. As a result continuous clevelo deployment takes less time. This also encarrages the idea of rapid app development

SIHRAN KUHARI ADI8 PAGE NO. DATE

(RAD)

The process of continuous deployment becomes happing free when developers reduce the humber of differences between the development and production stages

11) Logo (Treat 1095 as event streams)

or understanding beer behavior logs become extremely important. logs provide visibility into a running application's behavior.

A 12 factor app should not concern itselt with routing or storage of its and output stream. Neither should it not attempt to write to a manage logilies. All logilies should be written to statut and the environment decides how to process this attempt to console or saving and analysis system.

SIMRAN KUMARI ADI8

PAGE NO. DATE

12. Admin processos:

This tector states that any admin processes should be run in an identical environ. as the regular long-running processes of the app. In a local deploy, deva. use a direct shell command inside the apple checkout directory to perform one-time admin processes.

- Advantags:-

- Speed up effeciency
- bosides, the time spont on learning & applying these guidelines can help companies cave a great deal of money.
- trom some of the guid. such as logs, butter 1j's better to tollow the 12- Fac guidling.