Chapter 12: Decentralized Apps (DApps).

Smart contracts with a web or frontend are called DApps. A DApp is an application that is mostly or entirely decentralized.

It has a:
- Backend Software (Smart Contract).
- Frontend Software.
- Data Storage.
- Message communications.
- Name resolutions.

DApps provide some advantages which includes:
Resiliency: Because the business logic is controlled by a smart contract, a DApp backend will be fully distributed and managed on a blockchain platform.

Trasparency: The on-chain nature of a DApp allows everyone to inspect the code and be more sure about its function.

Censorship resistance: When a smart contract has been deployed, it is impossible to alter the code in the smart contract.

Now, let's take a look at a DApp and its constituents.

Backend (Smart Contract): In a DApp, smart contracts are used to store the business logic (program code) and the related state of your application.

You can think of a smart contract replacing a server-side (aka “backend”) component in a regular application.

Frontend (Web User Interface):
Unlike the business logic of the DApp, which requires a developer to understand the EVM and new languages such as Solidity, the client-side interface of a DApp can use standard web technologies (HTML, CSS, JavaScript, etc.). Interactions with Ethereum, such as signing messages, sending transactions, and managing keys, are often conducted through the web browser, via an extension such as MetaMask.

The frontend is usually linked to Ethereum via the web3.js JavaScript library, which is bundled with the frontend resources and served to a browser by a web server.

Data Storage:
This is where the DApp stores its content, like images, pictures, videos, etc.
IPFS (Inter Planetary File System) is a content-addressable storage system, meaning that whenever a file is uploaded to IPFS, the contents of the file are hashed, and using that hash, the file can be retrieved from any IPFS node.

Swarm:
Just like IPFS, this was built by the Ethereum team and it is a part of the Go-Ethereum set of tools. It saves files by distributing them to all Swarm nodes, and the hash of that file will be used to locate the file.

Message Communications:
Whisper:
This was built by the Ethereum team and it is a part of the Go-Ethereum set of tools. It is the best Message communications platform at the moment.

"You can design the best smart contract in the world, but if you don't provide a good interface for users, they won't be able to access it."

On the traditional internet, the Domain Name System (DNS) allows us to use human-readable names in the browser while resolving those names to IP addresses or other identifiers behind the scenes. On the Ethereum blockchain, the Ethereum Naming System (ENS) solves the same problem, but in a decentralized manner.

For example, the Ethereum Foundation donation address is 0xfB6916095ca1df60 bB79Ce92cE3Ea74c37c5d359; in a wallet that supports ENS, it's simply ethereum.eth.

ENS was launched on Star Wars Day, May 4, 2017 (after a failed attempt to launch it on Pi Day, March 15).

ENS is specified mainly in three Ethereum Improvement Proposals: EIP-137, which specifies the basic functions of ENS; EIP-162, which describes the auction system for the .eth root; and EIP-181, which specifies reverse registration of addresses.

ENS follows a “sandwich” design philosophy, a very simple layer on the bottom, followed by layers of more complex but replaceable code, with a very simple top layer that keeps all the funds in separate accounts.

The ENS operates on “nodes” instead of human-readable names: a human-readable name is converted to a node using the “Namehash” algorithm.

The base layer of ENS is a cleverly simple contract (less than 50 lines of code) defined by ERC137 that allows only nodes’ owners to set information about their names and to create sub-nodes (the ENS equivalent of DNS subdomains).

Namehash is a recursive algorithm that can convert any name into a hash that identifies the name.

```
namehash("my-ether.eth"):
        sha3(sha3('/0'*32) + sha3('eth') + sha3('my-ether'))
```

Resolvers:
The basic ENS contract can’t add metadata to names; that is the job of so-called “resolver contracts.” These are user-created contracts that can answer questions about the name, such as what Swarm address is associated with the app, what address receives payments to the app (in ether or tokens), or what the hash of the app is (to verify its integrity).

With the progress of DApps, as the technology matures further, more and more of our applications can be decentralized, resulting in a more resilient, censorship-resistant, and free web.