***APPLICATION LAYERS***

Mendukung untuk *network application* untuk mengirimkan data dari satu tempat ke tempat yang lain. Seperti browser yang kita buka mengarahkan kita ke HTTP, email yang kita pakai menggunakan SMTP.

***NETWORK APPS***

*Network apps* yang sudah digunakan karena memang paling dekat dengan manusia. Berikut terdapat beberapa *network apps.*

* E-mail
* Web
* Text messaging
* Remote login
* P2P file sharing
* multi-user network games
* Streaming stored video (YouTube, Hulu, Netflix)

Beberapa *network app* berjalan di *end system* yang berbeda dan melakukan komunikasi dengan *network*  menggunakan *socket.*

***APPLICATION ARCHITECTURES***

* *Client Server* :
  + Memiliki *host* yang memiliki IP *Address* permanenyang akan diakses oleh beberapa *client.*
  + *Client* berkomunikasi dengan server dan memiliki IP *Address* yang dinamis.
* *Peer-to-peer :* saling berbagi antar anggota jaringan
  + Tidak selalu sebagai server.
  + Masing - masing anggota bisa bertindak sebagai *client/server.*
  + Terkoneksi satu sama lain dan dapat mengubah IP *Address.*

***PROCESSES COMMUNICATING***

Host yang berbeda saling bertukar pesan. Semua aplikasi berjalan diatas *Operating System* dan *Operating System* tersebut akan mengirimkan *resources*nya. Pada P2P *Architectures* memiliki 2 proses, yaitu memproses sebagai *server* atau sebagai *client.*

***SOCKET***

Membantu untuk mengirimkan pesan ke *host. Socket* berada di tengah *process layer* dan *transport layer.* Karena dekat dengan *transport layer,* terdapat beberapa *service* pada *transport layer.*

* TCP : menjaga keandalan (contoh : pada data)
* UDP : toleran terhadap *loss* (contoh : pada video)

***ADDRESSING PROCESSES***

Untuk mengirimkan pesan, kita harus mengetahui tujuannya. *Host* mempunyai 32-bit IP *Address* yang unik. Namun *IP Address* tidak cukup untuk mengidentifikasi sebuah proses. Oleh karena itu, diperlukan *port,* dimana satu IP *Address* dapat memiliki banyak *port number.*

***TRANSPORT PROTOCOLS***

* TCP
  + *Reliable transport* : pada saat mengirim dan menerima proses
  + *Flow control* : tidak membebani penerima
  + *Congestion control* : mengatur kepadatan
  + *Connection - oriented* : ada *setup* antara proses *client* dan *server*

Agar TCP aman, digunakan SSL. Jadi paket - paket yang ingin dikirimkan dienkripsi. SSL berada di *application layer.*

* UDP
  + *Unreliable transport*, karena ada di suatu kondisi kita membutuhkan kecepatan

***WEB & HTTP***

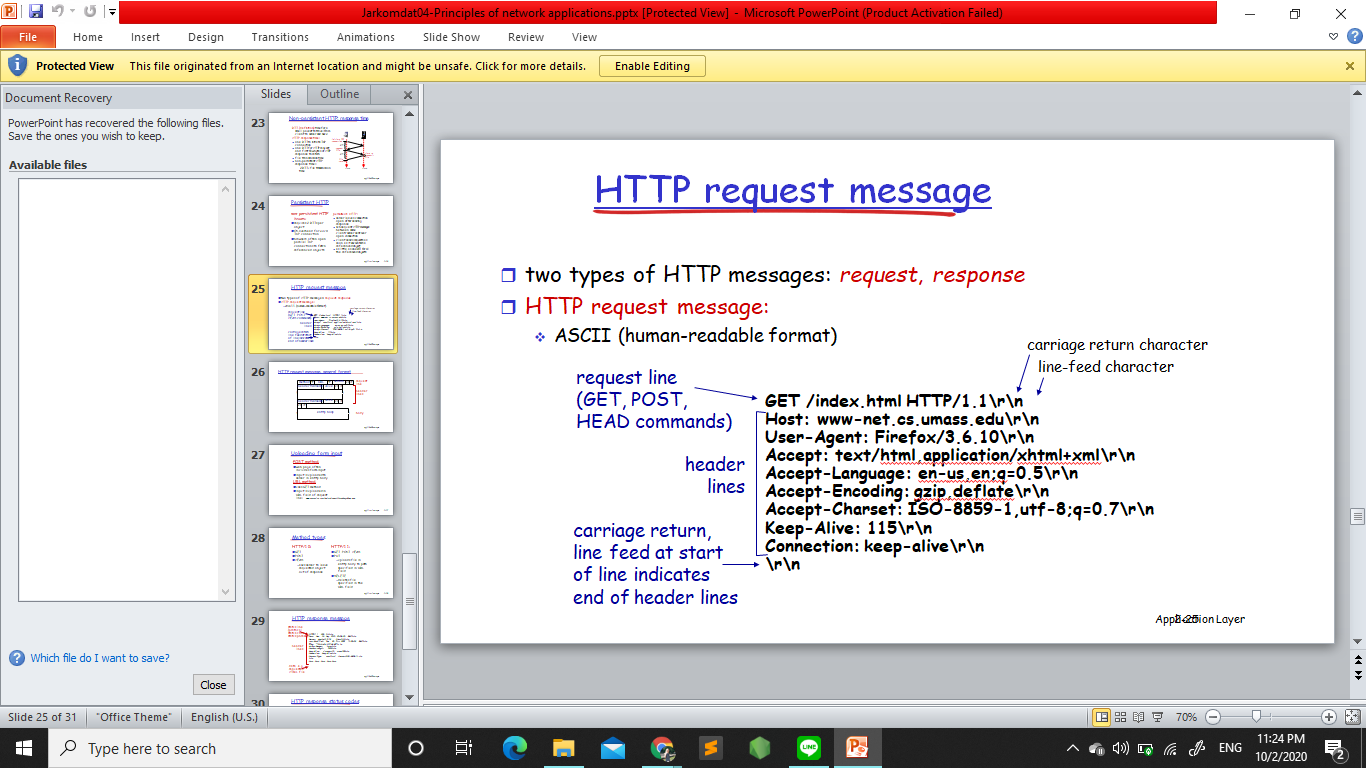
HTTP biasanya menggunakan TCP, karena tidak ingin ada data yang loss. Setiap kita melakukan *request* ke server, maka akan ada HTTP *message.*

***HTTP CONNECTIONS***

* HTTP yang persisten : melakukan *close* hanya di akhir
* HTTP yang tidak persisten : pada saat mengirimkan *request,* kemudian mendapat *respons,* kemudian di *close,* begitu seterusnya.

***HTTP REQUEST***

* Ada *method GET* atau *POST*

****

***HTTP RESPONSE***

* 200 : OK
* 301 : *Moved Permanently*
* 400 : *Bad Request*
* 404 : *Not Found*
* 505 : HTTP *Version Not Supported*