1. The two application layer protocols needed for this example are DNS and HTTP. At the Transport layers HTTP usually uses TCP whereas DNS primarily uses UDP with occasional use of TCP. So UDP and TCP are the only protocols needed at the transport layer
2. 2RTTs  + dobj + RTT1 + RTT2 + …. + RTTn
   1. 22RTTs + RTT1 + RTT2 + …. + RTTn
   2. 10RTTs + RTT1 + RTT2 + …. + RTTn
   3. 3RTTs + RTT1 + RTT2 + …. + RTTn
3. Yes, parallel instances of non-persistent HTTP would be useful in this scenario. Persistent HTTP you would give a very slight gain but not enough to be noticeable (Like .1% gain). The reason the gain would be so low is that transmission would take so long that the propagation speed would be an insignificant amount of time in comparison (Microseconds) so even if less connections have to be made back and forth between the sender and receiver the time saved would be almost non with such a small propagation delay.
   1. Yes, David’s use of parallel connections improves response time because he is not forced to use only one connection unlike the other people on the link, he can use five connections at once to speed up his response time.
   2. Yes, it is still beneficial for David to be using parallel connections. Although it is not as beneficial as it was before if David decided to stop using parallel connections the response time would drop. All users on the link should now have an equal size bandwidth.
   3. 240,000 (Average Object Size) / 15,000,000 (Bits Per Second) = 0.016 Seconds (Transmission Time)  
      62 (Number of Request a Second) \* .016 (Transmission Time) = 0.992  
      0.016 / 1 - 0.992 = 2 (Average Access Delay)  
      2 (Average Access Delay) + 1.5 (Average Response Delay) = **3.5 Seconds** (Average Response Time)
   4. 0.016 / 1 – (0.3 \* 0.992) = 0.022  
      Average response time for cache misses 0.016 / 1 – (0.3 \* 0.992) + 0.022 = 1.522

(0.7 \* 0) + (0.3 \* 1.522) = **1.24 Seconds**