I spent a total 4 hours on this project, mainly focus on the scalability and performance of the system.

Functions:

* User can log in.
* User can view, add, edit, delete to do task.

System Designs:

* Sign in. It is a synchronous call, not fully implemented, missing JWT token.
* Authorization. Not implemented. User can view others to do lists.
* To do related actions. The system is designed to read and write from nosql database for better performance. Service bus is used to handle request asynchronously and improve scalability.
* Cache layer. This could be done in either Redis or browser cookies. Due to time restrictions, this is not implemented. Ideally the system will persist data in these two ways synchronously, and these will send offline request separately to persist permanently. Over night job will be used for reconciliation reasons.
* Unit test is missing, due to time constraints and as business logic is quite limited. Only performance test is done, as the main benefit of such system design is performance.
* Easy plug in to NoSql db, scalable, able to handle a great work load of users and requests per minute. Easy to scale up.
* Service bus will offload the overload to different machines, as a demo, everything is within one single solution.

Improvement/Further development:

* More attractive UI. Obviously.
* Command/Query Segregation.
* Add base repository.
* Drag and drop to reprioritise tasks.
* AutoMapper to map between different layers of objects.
* Ajax call to make UI more user friendly.
* Mobile first design.
* Validation and constraints.
* Use cookie to handle network latency problems. Save and read from cookies, as actions are async, with bad network, UI could out of sync.
* Need a cache layer to improve performance. Like Redis.

Load test result attached. 20 threads, 400 runs, average time 383ms, no crash, no memory leak, system can easily handle the workload.

I think the most important design thinking for this system is how to handle a great load of user requests.

