1. What is the paper about?
   The guide is mainly describing what is a monolithic application and a microservice-based application. Moreover, the reader can see the differences between them and know when to use which. The last paragraph helps with an analysis.

2. What is a monolithic application?
   A monolithic application is characterized as a “single-tied software app in which different modules are combined into a single program”. In short, this app has all of its components tied and deployed as a single running brain.

3. What is a microservices-based application?
   Unlike monolithic applications, a microservice-based one is defined by the functionality of an app being split between mini-applications, where each is executing its own feature.

4. When looking at both types of application, the biggest difference between them is the way of running components: in a microservice based app the services can be deployed and updated independently, which gives more flexibility, for instance when a bug happens in a microservice, only that one is impacted, while in a monolithic application everything runs as a single and indivisible unit, which means that each bug affects the application as a whole. Another major difference would be the relation between the app and the database: in a monolithic application, it is most common that the whole app works with one db, while a microservice-based one gives the opportunity to use different, specific dbs for each service, that way each service can work and share data on its own, accelerating the work of the application.

5. Why would one want to migrate from one type of application to another?

The first thing that comes to my mind would be scalability. The microservice architecture is easier to update, in terms of adding new features. When developing such an app, u can split the work from one big team to several, smaller teams that would each work on its own microservice. That way the developing process could be done faster. On the other hand, I see the monolithic application being easier to test and deploy. When u have one unit of an app, it is easier and less time-spending to conduct tests and verify its efficiency, than testing each microservice where there might be tens of them in a big project.

6. When starting a new project, which approach is best? Why?
   I see it this way, if the project is a simple one, not requiring a lot of logical executions and extra scalability, then it should be done as a monolithic application. It can be developed with no extra analysis and expertise on splitting the execution logic, designing a gateway for the microservices and so on.
   But if the project is more complex, has many features, requires extra scalability and is prognosed to be updated with new features frequently, then it should have the microservice-based architecture. This way the development can be split among smaller teams, that would work efficiently on their services and not depend as much on sharing resources.

7. How do these approaches compare in the context of maintainability?
   As I previously mentioned, the difference between them is the possibility to split the workload in microservice-based applications. While having separate small teams, they can react faster and more efficiently on solving bugs in their service, while not stopping the work of the other functionalities of the application. On the other hand, in a monolithic app, let's say in a bigger project, a small bug in a single feature could slow down the working process of the other ten features, thus slowing the running app as a whole.