1. Explain your understanding of CI, CD, DevOps practices in software development. What is it? What benefits it provides? Why do we need to spread it widely across the world?

DevOps is a set of methods that engineers use to exploit, develop, and design the entire life cycle of the software development process. One of method is Continuous Integration and Continuous delivery. Continuous Integration (CI) is the process of automating the build and testing of code every time when developers add code to repository several times a day. Continuous delivery (CD) is a way where teams release quality products frequently and predictably from source code repository to production automatically. The key benefit of it is that you can detect errors quickly and fix them more easily. Also, you always have a working version of the product that you can show to the customer. We need to spread it across the world because it allows developers to develop programs easier, testers to test faster, and the customer to receive the product faster, cheaper and more reliable.

2. There's an application that grew quite big. It's been decided to migrate it from monolithic app to microservices.
What will be the advantages of its containerisation?
How often both solutions (ms and mono) may be updated under live traffic?

A monolithic application is a single unit. It consists of parts: a database, a client-side user interface, and a server-side application. I mean, it’s about building an application consisting of many small services that can be independently deployed and maintained, don’t have any dependencies but rather communicate with each other through lightweight mechanisms and lack a centralized infrastructure. It is even possible to write these small (micro-) services each in its own language.The benefits are: to easily deployment, debug or even replace quickly. With microservices, you can’t break an application: If something goes wrong, it will go wrong only within its own microspace, while the rest of the application will continue working.
A monolithic application describes a software application which is designed without modularity. Monolithic, in this context, means composed all in one piece. Monolithic software is designed to be stand-alone; components of the program are interconnected. If any program component must be updated, the whole application has to be rewritten. The benefit of monolithic architectures is that monolithic programs have better throughput than modular way.

3. Software Pipeline. What is it? Why do we need to use it?
Describe mandatory steps of PPL and application lifecycle.

Software Pipeline is a concept for avoiding waste in the software development process, and it is used for providing quick feedback to the team during deployment.
We can picture it graphically in detail like this:

Git push->build->parallel tasks(code analysis, unit tests, integration /API tests)->smoke test> parallel tasks->(regression tests, visual regression tests)->production

We can picture **Application lifecycle** graphically like this:
Plan->Develop->Test->Deploy->Maintain->Plan->Develop->Test->Deploy->Maintain->...

4. You have application sources in remote repository. And production environment under live traffic.

How source code transformed to running service?
Please share steps of release process.

The CI server checks the updates for subversion. If it detects there is an update it gets a copy of the project then it runs a build script that compiles the source code, integrates the database, runs the tests, deploys the software and the environment. After that, the Server sends a status message to the developer who made the comit. If this works correctly, the product is sent to the production server.

5. Share known solutions which are used in software development.

What mostly used for orchestration?
What is known Configuration Management tools?
How we can establish code review process and why

The most popular **orchestration applications** are Jenkins, TeamCity and travisLC.

Configuration Management tools allow automatically us to environments, deploying applications, maintaining infrastructures.
The most common:

- Puppet
- Chef
- Docker

Code review is analiz of computer source code. It is intended to find mistakes that surfaced in software development. it improves the overall quality of software.
The most common:

- Collaborator
- Codebrag
- Gerrit
- Codestriker