**Experience with testing**

Software testing is the process of evaluating and verifying that a software product or application does what it is supposed to do. The benefits of testing include preventing bugs, reducing development costs and improving performance.

There are many different types of software tests, each with specific objectives and strategies:

* **Acceptance testing:** Verifying whether the whole system works as intended.

**Integration testing:** Ensuring that software components or functions operate together.

**Unit testing:** Validating that each software unit performs as expected. A unit is the smallest testable component of an application.

**Functional testing:** Checking functions by emulating business scenarios, based on functional requirements. Black-box testing is a common way to verify functions.

**Performance testing:** Testing how the software performs under different workloads. Load testing, for example, is used to evaluate performance under real-life load conditions.

**Regression testing:** Checking whether new features break or degrade functionality. Sanity testing can be used to verify menus, functions and commands at the surface level, when there is no time for a full regression test.

**Stress testing:** Testing how much strain the system can take before it fails. Considered to be a type of non-functional testing.

**Usability testing:** Validating how well a customer can use a system or web application to complete a task.

**Experience in developing distributed / high-load systems**  
**A distributed system** is a computing environment in which various components are spread across multiple computers (or other computing devices) on a network. These devices split up the work, coordinating their efforts to complete the job more efficiently than if a single device had been responsible for the task.

Distributed systems are an important development for IT and computer science as an increasing number of related jobs are so massive and complex that it would be impossible for a single computer to handle them alone.  
  
**High-load systems**. High load means the resistance of an Internet project to high loads. But wait, there’s more. This is not some universal piece of code that we copy-paste and after which everything flies. This is setting up the site architecture: working with databases, a server, using modern technologies and programming languages.

**Experience in building microservice architecture**

Microservices - also known as the microservice architecture - is an architectural style that structures an application as a collection of services that are

Highly maintainable and testable  
Loosely coupled  
Independently deployable  
Organized around business capabilities  
Owned by a small team

The microservice architecture enables the rapid, frequent and reliable delivery of large, complex applications. It also enables an organization to evolve its technology stack.