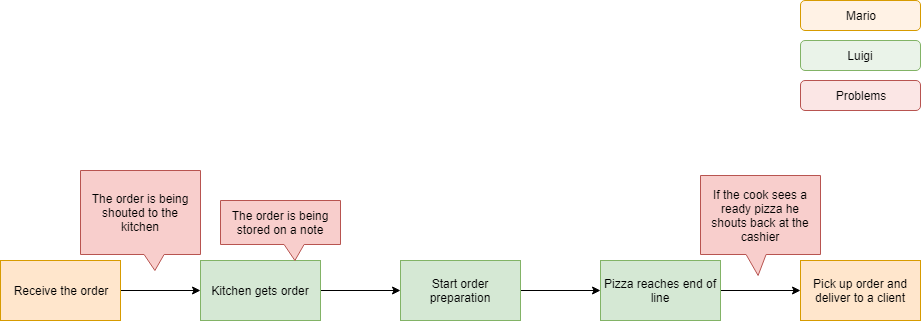
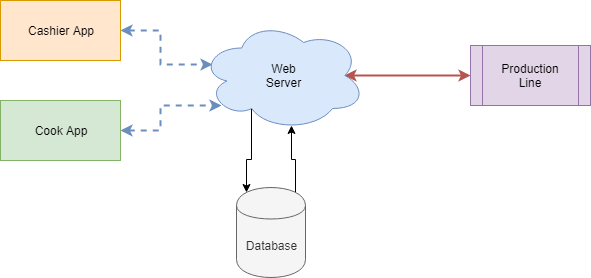
The pizza show has several problems, which are indicated in the following flowchart, depicting the process of ordering:



Firstly, there is an obvious issue with the communication between the cashier and the cook. The shouting back and forth is not an efficient way to transfer information to the kitchen and back. Furthermore, the order itself is stored on a flying note, if heard by the cook at all. That note can get lost or damaged, rendering the order non-existent. Additionally, when the order is ready for serving, there’s an unnecessary waiting time for the cook to see the baked pizza at the end of the line, followed by a shout to the cashier, informing him he can complete an order. And that’s the main bottleneck in the process.

Our solution to all of these problems is developing a system to manage and keep track of all orders. It looks something like this:



We have two distinct applications for the cashier and the cook, connected to our custom web server, which processes all requests and stores the orders in a database. The server also has a wired connection to the production line, which is automated to check for ready orders.

|  |  |  |
| --- | --- | --- |
| Component | Implemented with | Responsible for |
| Cashier App | C# Application | * Creating orders * Finishing orders |
| Cook App | C# Application | * Reading orders * Editing orders |
| Web Server | NodeMCU ESP826 board | * Establishing communication |
| Database | Memory Expansion | * Storing orders |
| Production Line | Arduino Uno | * Keeping track of the orders * Editing orders |