1. Configure Spring Security on Micro-services Architecture.

Configuring security in Micro-services architecture can be a very complex story. We do found some tutorials online, but they try to simply the process by using InMemoryAuthentication which is not what we wanted in this case study. We should use our OcrUser service application to do the JDBC authentication/ authorization and return a custom UserDetails as a token. This token can be further used for API calls to secure our API services. Currently, the JDBC authentication/authorization is implemented in OcrClient web application and the APIs from all the services are public, which means that the web pages are secure but all the APIs can be accessed by anyone.

1. Redesign some of the models to make full use of the benefits from MongoDB.

For current version, OcrRestaurant, OcrMenu, OcrFood and OcrOrder use MongoDB, and they all use corresponding entities (Restaurant, Menu, Food, Order) for mapping. When we add/remove/modify properties from those entities, there is no conflict in the database/document. A smarter way can be using JSONObject for mapping, however which needs refactoring code for the whole system. A big advantage is that you can choose whatever properties we want to store while creating the JSONObject.

Specifically, for OcrMenu, currently the menus are randomly distributed according to the time when they are created. Menus from one restaurant can be found by looping though the whole document with findMenusByRestaurantId. A better way is to create an JSONObject inside OcrMenu database immediately when a new restaurant is instantiated. All menus created for that restaurant can be put into that JSONObject with the right restaurant id. In this case, all menus from one restaurant are put in a concentrated big entity which can be easily exported for future usage.

1. Redesign native application with Apache Cordova (formerly PhoneGap).

We started from researching and prototyping with PhoneGap, and we found that we can only make application for Android and Windows users due to the limitation of OS of our laptops. Hence, we decided to make an android application on Android Studio. In the future, if macOS can be used, making a native application with PhoneGap can be a better choice. In this case, you may build applications for mobile devices using CSS3, HTML5, and JavaScript instead of relying on platform-specific APIs like those in Android, iOS, or Windows Phone.

1. Implements Other Nice-to-have features.

* Implement real payment system with existing payment API, so that splitting up bills can also be possible.
* Redesign the application so that local network hosting can also be possible.