Online Food Ordering System

1. Introduction

An online food ordering system. Every restaurant, diner etc. can use this system. It's build to manage products and orders being placed. Customers can use it to find restaurants, diners etc. to order food. The company will have a nice overview of all the orders that are placed and they can view and edit all the products they offer. The customers of these businesses can view which companies are in their area to order food and they can place companies as their favorites.

Expected List of Features
As a company I should be able to…

- create an account, so that I can login.
- recover my account if I forget my password.
- add and edit my accounts information.
- insert products in my product catalog.
- view, edit and delete my products.
- place a product out of stock.
- view all of my orders.
- report a customer, so their account can be viewed and blocked if needed.

As a customer I should be able to…

- create an account, so that I can login.
- recover my account if I forget my password.
- browse products (dishes etc).
- browse restaurants.
- search for specific products or restaurants.
- view the details of a product.
- view the profile of a restaurant.
- place restaurants as my favorite.
- place products as my favorite.
- place products in my shopping cart.
- view my shopping cart and checkout.
- share products and restaurants to social media.

As an admin I should be able to…

- login.
- view, block and delete every restaurant and customer.
- add a restaurant to the list of featured restaurants

2. Design and Implementation

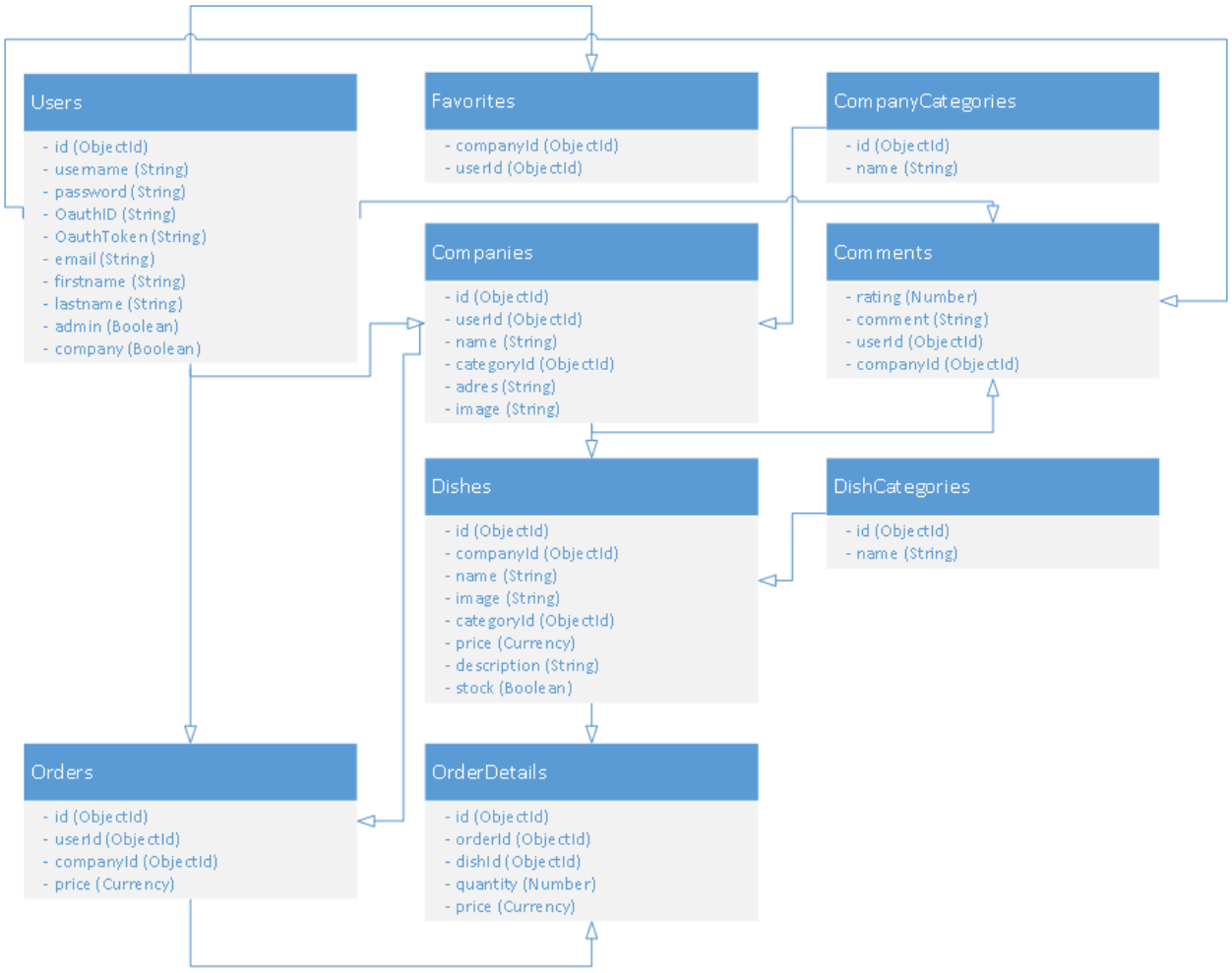
2.1 The REST API Specification

Routes	**HTTP Methods**
/companies	get - post
/companies/:companyId	get - put - delete
/companies/:companyId/comments	get - post
/companies/:companyId/comments/:commentId	put - delete
/companies/:companyId/dish-categories	get - post
/companies/:companyId/dishes	get - post
/companies/:companyId/orders	get
/companies/:companyId/orders/:orderId	get
/dishes	get - post
/dishes/:dishId	get - put - delete
/dishes/company-categories	get - post
/dishes/company-categories/:categoryId	put - delete
/users	get - post
/users/:userId	get - put - delete
/users/:userId/favorites	get - post
/users/:userId/favorites/:dishId	get - put - delete
/users/:userId/orders	get - post
/users/:userId/orders/:orderId	get - post
/users/register	post
/users/login	post
/users/logout	get
/users/facebook	get
/users/facebook/callback	get

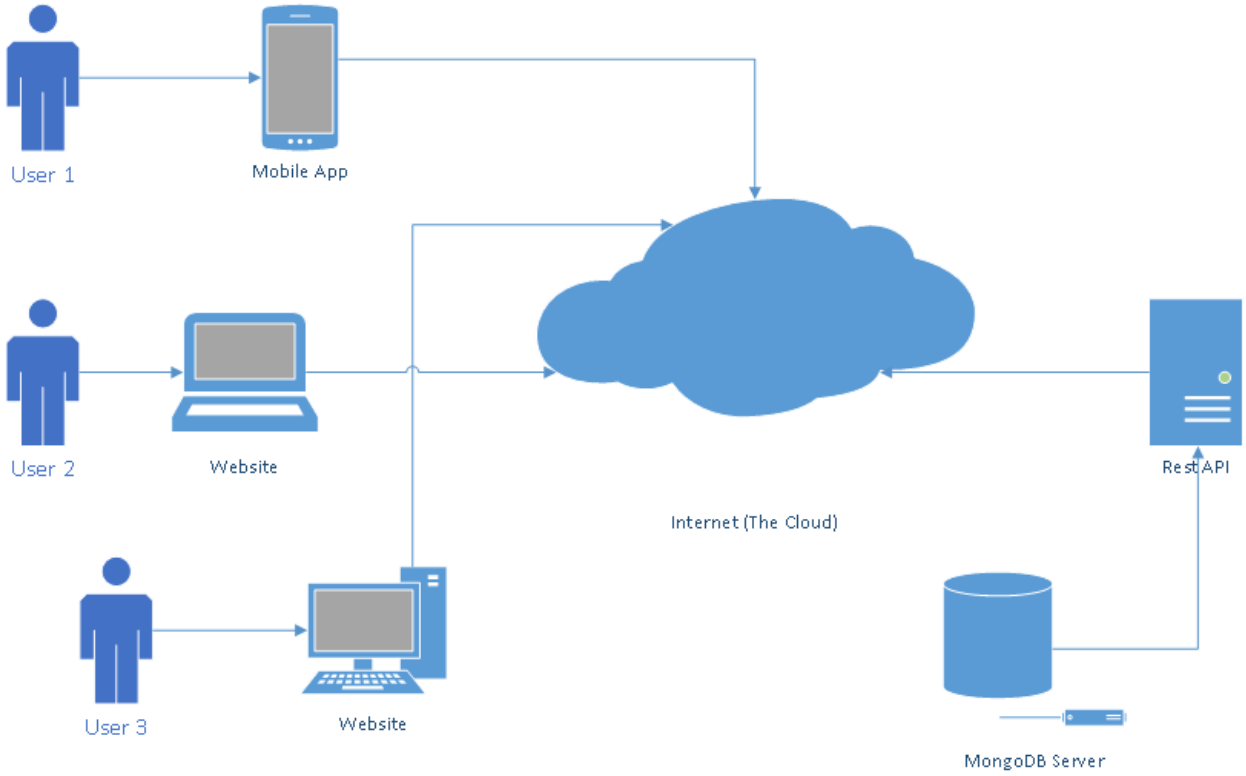
2.2 Front-end Architecture Design

URL	TemplateUrl	Controller
/	views/front/home.html	FrontHomeCtrl
/aboutus	views/front/aboutus.html	FrontAboutCtrl
/contactus	views/front/contactus.html	FrontContactCtrl
/companies	views/front/companies.html	FrontCompaniesCtrl
/company-profile	views/front/company-profile.html	FrontCompanyProfileCtrl
/company-menu	views/front/company-menu.html	FrontCompanyMenuCtrl
/company-about	views/front/company-about.html	FrontCompanyAboutCtrl
/company-contact	views/front/company-contact.html	FrontCompanyContactCtrl
/products	views/front/products.html	FrontProductsCtrl
/cart	views/front/cart.html	FrontCartCtrl
/profile	views/front/profile.html	FrontProfileCtrl
/favorites	views/front/favorites.html	FrontFavoritesCtrl
/orders	views/front/orders.html	FrontOrdersCtrl
/order-details	views/front/order-details.html	FrontOrderDetailsCtrl
/admin/	views/admin/home.html	AdminHomeCtrl
/admin/category	views/admin/category.html	AdminCategoryCtrl
/admin/category-details	views/admin/category-details.html	AdminCategoryDetailsCtrl
/admin/products	views/admin/products.html	AdminProductsCtrl
/admin/product-details	views/admin/product-details.html	AdminProductDetailsCtrl
/admin/profile	views/admin/profile.html	AdminProfileCtrl
/admin/settings	views/admin/settings.html	AdminSettingsCtrl
/admin/orders	views/admin/orders.html	AdminOrdersCtrl
/admin/order-details	views/admin/order-details.html	AdminOrderDetailsCtrl

2.3 Database Schemas, Design and Structure



2.3 Communication



3. Conclusions

Now that I've documented the architecture design and structure of my project I expect the development to go much smoother. The REST specifications are clear, the front end architecture design is also clear and the database structure is designed and all is clear. I think that the design and architecture will get tweaked here and there while testing the app, website and REST API to meet the requirements of the original idea.