

Dinein.co.uk - System Architecture Description

Document Information

|  |  |
| --- | --- |
| **Current Version** | 1.0 |
| **Document Owner** | Alex Popov,  Sergey Vodotyka,  Sasha Anisimov |
| **Status** | Draft |

Version History

|  |  |  |  |
| --- | --- | --- | --- |
| **Version #** | **Created Date** | **Description** | **Author** |
| 1.0 | 06/01/2015 | Initial Release | Alex Popov,  Sergey Vodotyka,  Sasha Anisimov |

# Gateway project

The Gateway project encapsulates all REST API services in the system. There are 2 packages: 3rd party services and public services.

External Services package implements all functionality to communicate with a third party components like WorldPay, Twilio, Mandrill, Micros, Dispatch console, etc.

Internal Services package contains the list of interfaces which will used from different parts of the system like authorization, reporting, etc.

## 3rd party services

External services package includes following services:

### Twilio Service

#### Functionality

Provides interface with Twilio for Text notifications.

#### Methods

**Send** – send order text status updates to the user.

**Receive** – send order requests from user to the restaurant.

#### Interfaces

Twilio Service implements ITextNotificationService interface so it can be replaced with a different text notification system at any time.

#### Relations

No relations.

### Mandril Service

#### Functionality

Provides interface with Mandrill (by MailChimp) for email notifications.

#### Methods

**Send** – send email notification to the user.

#### Interfaces

Mandrill Service implements IEmailNotificationService interface so it can be replaced with a different email system at any time.

#### Relations

No relations.

### Micros Service

#### Functionality

Provides interface with Micros management system.

#### Methods

**FetchMenuByRestaurant** – fetch menu information for specific restaurants.

**FetchMenuOptionsByRestaurant** – fetch menu options’ information for specific restaurants.

**FetchRestaurantInfo** – fetch restaurant information.

**FetchRestaurantSchedule** – fetch restaurant schedule.

#### Relations

No relations.

### MicrosKDS Service

#### Functionality

Provides interface with Micros KDS system.

#### Methods

**FetchFoodPreparationTimeByRestaurant** – fetch food preparation time information for specific restaurants.

**UpdateOrderStatus** – update from the KDS system order status.

**SendOrder** – send an order to the KDS system.

**GetFoodPreparationTimeByOrder** – get a response from the KDS system about order

preparation estimate time (or rejection).

#### Relations

No relations.

### Dispatch Service

#### Functionality

Provides interface to interact with Dispatch console.

#### Methods

**SendOrder** – send order to Dispatch system.

**GetOrderStatus** – get order status from the Dispatch system.

**UpdateOrderStatus** – update order status and send it to Dispatch system.

**GetDriverInfo** – get driver info and location from Dispatch system.

#### Relations

No relations.

### WorldPay Service

#### Functionality

Provides interface with payment processing system - WorldPay.

#### Methods

**Autorize** – authorize payment

**Capture** – capture payment

**Void** – void payment

**Refund** – refund payment

#### Interfaces

WorldPay Service implements IPayService interface so it can be replaced with a different payment system at any time.

#### Relations

No relations.

### TwilioIVR Service

#### Functionality

Provides interface with Twilio for IVR calls.

#### Methods

**Send** – send IVR order requests to the restaurant.

#### Interfaces

TwilioIVR Service implements IIVRService interface so it can be replaced with a different IVR system at any time.

#### Relations

No relations.

## Public services

Public services package includes following services:

### Restaurant Service

#### Functionality

Provides interface for managing restaurants in the system

#### Methods

**SetRestaurantAsFavourite** – set restaurant as favorite.

**AddRestaurant** – add new restaurant to the system.

**SetRestaurantAsNewest -** set restaurant as newest.

**SetRestaurantStatus** **–** set active/inactive to specific restaurant.

#### Relations

No relations.

### Order Service

#### Functionality

Provides interface for managing orders in the system

#### Methods

**CreateOrder** – create order in the system

**SetStatus** – update status of specific order.

**GetOrders –** get list of orders in the system.

**GetOrderHistory –** get order history by specific user.

**Cancel –** cancel specific order.

#### Relations

No relations.

### Report Service

#### Functionality

Provides interface for getting reports data.

#### Methods

**RestaurantReport** – get report by specific restaurant.

**ProfitAndLossReport** – get profit and loss report.

**CorporateReport** – get report by company.

#### Relations

No relations.

### Autorization Service

#### Functionality

Provides interface to authorize user in the system. This service will entry for all parts in the system.

#### Methods

**Login –** log in user to the system.

**Logout –** log out user from the system.

**TokenVerification –** log in user to the system with security token.

**ChangePassword –** change password for specific user.

**ForgotPassword/ResetPassword –** reset password for specific user.

#### Relations

No relations.

### Email Service

#### Functionality

Provides public interface to send email notifications.

#### Methods

**SendTo –** send email status updates to the user.

**SendCustomEmailTo –** send custom emails to specific system users. The HTML template and the list of users should be provided.

**SendScheduleEmailTo –** send scheduled email notifications to selected users. The HTML template, the list of users and schedule should be provided.

#### Relations

Gateway MandrillService – call REST API method Send

### Notification Service

#### Functionality

Provides public interface to send IVR requests.

#### Methods

**Send –** send IVR order requests to the restaurant.

#### Relations

Gateway TwilioIVRService– call REST API method Send

### Menu Service

#### Functionality

Provides public interface to integrate restaurants using Micros management system.

#### Methods

**FetchMenuByRestaurant** – fetch menu information for specific restaurants.

**FetchMenuOptionsByRestaurant** – fetch menu options’ information for specific restaurants.

#### Relations

Gateway MicrosService– call REST API method FetchMenuByRestaurant

Gateway MicrosService– call REST API method FetchMenuOptionsByRestaurant

### FoodPreparation Service

#### Functionality

Provides public interface to integrate restaurants using Micros KDS management system.

#### Methods

**FetchFoodPreparationTimeByRestaurant** – fetch food preparation time information for specific restaurants.

**UpdateOrderStatus** – update from the KDS system order status.

**SendOrder** – send an order to the KDS system.

**GetFoodPreparationTimeByOrder** – get a response from the KDS system about order

preparation estimate time (or rejection).

#### Relations

Gateway MicrosKDSService– call REST API method FetchFoodPreparationTimeByRestaurant

Gateway MicrosKDSService– call REST API method UpdateOrderStatus

Gateway MicrosKDSService– call REST API method SendOrder

Gateway MicrosKDSService– call REST API method GetFoodPreparationTimeByOrder

### Payment Service

#### Functionality

Provides public interface to integrate payment processing system - WorldPay.

#### Methods

**Autorize** – authorize payment

**Capture** – capture payment

**Void** – void payment

**Refund** – refund payment

#### Relations

Gateway WorldPayService– call REST API method Autorize

Gateway WorldPayService – call REST API method Capture

Gateway WorldPayService – call REST API method Void

Gateway WorldPayService – call REST API method Refund

### Delivery Service

#### Functionality

Provides public interface to integrate with Dispatch.

#### Methods

**SendOrder** – send order to Dispatch system.

**GetOrderStatus** – get order status from the Dispatch system.

**UpdateOrderStatus** – update order status and send it to Dispatch system.

#### Relations

Gateway DispatchService– call REST API method SendOrder

Gateway DispatchService – call REST API method GetOrderStatus

Gateway DispatchService – call REST API method UpdateOrderStatus

# Dinein Website project

The main Dinein.co.uk web portal.

## Controllers

### UserController

#### Functionality

Provides interface to login/logout user in the web site.

#### Methods

**Login –** log in user to the web site.

**Logout –** log out user from the web site.

**ChangePassword –** change password for specific user.

**ForgotPassword/ResetPassword –** reset password for specific user.

#### Relations

Gateway AutorizationService – call REST API methods Login,

Gateway AutorizationService – call REST API methods Logout,

Gateway AutorizationService – call REST API methods ChangePassword,

Gateway AutorizationService – call REST API methods ResetPassword,

Uses User, UserGroup model,

Updates User view

### RestaurantController

#### Functionality

Provides interface for managing restaurants in the web site.

#### Methods

**SetRestaurantAsFavourite** – set restaurant as favorite

#### Relations

Gateway RestaurantService – call REST API method SetRestaurantAsFavourite,

Uses Restaurant model,

Updates Restaurant view

### MenuController

#### Functionality

Provides interface for displaying menu on the web site.

#### Methods

**GetMenuByRestaurant** – get menu by specific restaurant from MenuService

#### Relations

Gateway MenuService – call REST API method FetchMenuByRestaurant,

Uses MenuCategory, MenuOption, MenuType models,

Updates Menu view

### OrderController

#### Functionality

Provides interface for order management from user perspective.

#### Methods

**CreateOrder** – create order by calling CreateOrder from OrderService

**UpdateOrderStatus** – update order status by calling SetStatusfrom OrderService

**GetOrderHistory** – get order history by calling GetOrderHistory fromOrderService

#### Relations

Gateway OrderService,

Gateway DeliveryService,

Gateway FoodPreparationService,

Uses Order model,

Updates Order view

### PaymentController

#### Functionality

Provides interface for payment processing.

#### Methods

**Autorize** – authorize payment

**Capture** – capture payment

**Void** – void payment

**Refund** – refund payment

#### Relations

Gateway PaymentService – call REST API method Autorize,

Gateway PaymentService – call REST API method Capture,

Gateway PaymentService – call REST API method Void,

Gateway PaymentService – call REST API method Refund,

Updates Payment view

# Admin portal project

The admin portal for managing Dinein system.

## Controllers

### UserController

#### Functionality

Provides interface to manage access for user in the admin portal.

#### Methods

**Login –** log in user to the web site.

**Logout –** log out user from the web site.

**ChangePassword –** change password for specific user.

**ForgotPassword/ResetPassword –** reset password for specific user.

**CreateUser –** create specific user.

**AssignUserToRole –** assign user to specific role.

**GetUserList –** get a list of users in the system.

**DisableUser –** disable user in the system.

#### Relations

Gateway AutorizationService – call REST API methods Login,

Gateway AutorizationService – call REST API methods Logout,

Gateway AutorizationService – call REST API methods ChangePassword,

Gateway AutorizationService – call REST API methods ResetPassword,

Uses User, UserGroup model,

Updates User view

### RestaurantController

#### Functionality

Provides interface for managing restaurants in the admin portal.

#### Methods

**AddRestaurant** – add new restaurant to the system.

**SetRestaurnatAsNewest** – set restaurant as newest.

**SetActivityToRestaurant** – enable/disable specific restaurant.

**UpdateRestaurant** – update information about specific restaurant in the system.

**GetRestaurantList** – get a list of restaurants in the system.

#### Relations

Gateway RestaurantService – call REST API method **AddRestaurant**,

Gateway RestaurantService – call REST API method **SetRestaurnatAsNewest**

Gateway RestaurantService – call REST API method **SetActivityToRestaurant**

Uses Restaurant model,

Updates Restaurant view

### ReportController

#### Functionality

Provides interface to display/print different reports in the admin portal.

#### Methods

**RestaurantReport** – get report by specific restaurant.

**ProfitAndLossReport** – get profit and loss report.

**CorporateReport** – get report by company.

#### Relations

Gateway ReportService – call REST API method **RestaurantReport**,

Gateway ReportService – call REST API method **ProfitAndLossReport**

Gateway ReportService – call REST API method **CorporateReport**

Updates Report view

### OrderController

#### Functionality

Provides interface to manage orders the admin portal.

#### Methods

**CreateOrder** – create new order in the system.

**UpdateOrderStatus** – change status of specific order.

**GetOrders** – get list of orders.

**CancelOrder** – cancel order.

#### Relations

Gateway OrderService – call REST API method **CreateOrder**,

Gateway OrderService – call REST API method **UpdateOrderStatus,**

Gateway OrderService – call REST API method **GetOrders,**

Gateway OrderService – call REST API method **CancelOrder**

Updates Order view

### MenuController

#### Functionality

Provides interface to manage menu in the admin portal.

#### Methods

**CreateMenu** – create new menu for specific restaurant.

**UpdateMenu** – update menu.

**DeleteMenu** – delete menu for specific restaurant.

**GetMenuList** – get menu list of specific restaurant.

#### Relations

Gateway OrderService – call REST API method **CreateOrder**,

Gateway OrderService – call REST API method **UpdateOrderStatus,**

Gateway OrderService – call REST API method **GetOrders,**

Gateway OrderService – call REST API method **CancelOrder**

Updates Order view

### CMSController

#### Functionality

Provides interface to manage static content in the system.

#### Methods

**CreatePage** – create new static page.

**CreateNavigationItem** – assign static page to specific navigation item.

**UpdatePage** – update content of specific page.

**DeletePage** – soft deletion of specific page.

**GetPageList** – get list of static pages in the system.

#### Relations

Updates Page view

# Common project

Shares common components between projects.

## Models

### FavouriteRestaurant

#### Description:

Favorite restaurants chosen by customers.

#### Relations:

ONE-TO-ONE with ‘Restaurant’

### RestaurantChain

#### Description:

Describes restaurant chains

#### Relations:

MANY-TO-ONE with ‘Client’

### RestaurantGroup

#### Description:

Describes restaurant groups

#### Relations:

MANY-TO-ONE with ‘RestaurantChain’

### RestaurantGroupRestaurant

#### Description:

Restaurants which belong to some Restaurant groupe

#### Relations:

### MenuType

#### Description:

The set of menu types (breakfast, lunch, etc.) which can be offered in restaurants during certain time period.

#### Relations:

MANY-TO-ONE with ‘Clients’

MANY-TO-MANY with ‘MenuCatagories’

### MenuCategory

#### Description:

Submenus of restaurants for each menu type.

#### Relations:

MANY-TO-ONE with ‘Clients’

### Menu

#### Description:

Information about menues.

#### Relations:

MANY-TO-MANY with ‘MenuCategory’

MANY-TO-MANY with ‘MenuType’

### MenuItem

#### Description:

Dishes offered in restaurants

#### Relations:

MANY-TO-MANY with ‘MenuItem’

MANY-TO-MANY with ‘FavouriteMenu’

### Allergy

#### Description:

Contains allergy related information

#### Relations:

MANY-TO-MANY with ‘MenuItems’

### FavouriteMenu

#### Description:

Favorite menu items in restaurants chosen by users.

#### Relations:

### MenuBundle

#### Description:

Represent the list of bundle items.

#### Relations:

MANY-TO-MANY with ‘MenuBundleItem’

### MenuBundleItem

#### Description:

Represents set of dishes (menu items) which is offered for sell together.

#### Relations:

MANY-TO-MANY with ‘MenuItem’

### MenuItemsSimilar

#### Description:

Represents similar menu items which can be replaced in one dish.

#### Relations:

ONE-TO-MANY with ‘MenuItem’

### Voucher

#### Description:

Represents a list of special promotion code which can be used for payment in special restaurants.

#### Relations:

MANY-TO-ONE with ‘Client’

### VoucherRestaurant

#### Description:

Represents a list of special restaurant with ability of voucher payment.

#### Relations:

ONE-TO-MANY with ‘Voucher’

### VoucherRestaurantGroup

#### Description:

Represents a group of voucher restaurants.

#### Relations:

ONE-TO-MANY with ‘Voucher’

ONE-TO-MANY with ‘VoucherRestaurant’

### VoucherRestaurantChain

#### Description:

Represents a group of voucher restaurant group.

#### Relations:

ONE-TO-MANY with ‘Voucher’

ONE-TO-MANY with ‘VoucherRestaurantGroup’

### VoucherUser

#### Description:

User which can pay with voucher

#### Relations:

ONE-TO-MANY with ‘Voucher’

ONE-TO-MANY with ‘Resaurant’

ONE-TO-ONE with ‘Users’

### Client

#### Description:

Represent clients of Denein. It contains restaurant owner, group restaurant owners and chain restaurant owners.

#### Relations:

ONE-TO-MANY with ‘Voucher’

ONE-TO-MANY with ‘Resaurant’

ONE-TO-MANY with ‘ResaurantGroup’

ONE-TO-MANY with ‘ResaurantChain’

### Label

#### Description:

The list of labels that will be used in web site and will be translated.

#### Relations:

MANY-TO-MANY with ‘LabelLanguage’

### LabelLanguage

#### Description:

The list of label translations.

#### Relations:

### Language

#### Description:

The list of languages

#### Relations:

ONE-TO-MANY with ‘LabelLanguage’

### Base

#### Description:

The list of all address bases

#### Relations:

MANY-TO-ONE with ‘Client’

MANY-TO-ONE with ‘Area’

### Area

#### Description:

The list of address areas.

#### Relations:

MANY-TO-ONE with ‘Country’

### Country

#### Description:

The list of countries in which DineIn is used.

#### Relations:

### Postcode

#### Description:

The list of postcodes.

#### Relations:

MANY-TO-ONE with ‘Country’

MANY-TO-ONE with ‘Area’

### City

#### Description:

The list of cities

#### Relations:

ONE-TO-MANY with ‘Postcode’

### Address

#### Description:

The list of all addresses

#### Relations:

MANY-TO-ONE with ‘City’

MANY-TO-ONE with ‘Postcode’

MANY-TO-ONE with ‘Base’

### UserAddress

#### Description:

The list of all user addresses

#### Relations:

ONE-TO-ONE with ‘Address’

ONE-TO-ONE with ‘User’

### User

#### Description:

The list of users registered in DineIn system

#### Relations:

MANY-TO-ONE with ‘UserGroup’

### UserGroup

#### Description:

The list of user group of all companies for corporate orders.

#### Relations:

MANY-TO-ONE with ‘Permissions’

### Permissions

#### Description:

Restrictions for expense amount for User group.

#### Relations:

### OrderItem

#### Description:

The list of menu items in order

#### Relations:

MANY-TO-ONE with ‘Order’

### Order

#### Description:

The list of all orders.

#### Relations:

MANY-TO-ONE with ‘promotion’

MANY-TO-ONE with ‘contact’

ONE-TO-ONE with ‘GroupOrder’

MANY-TO-ONE with ‘CorporateOrder’

### GroupOrder

#### Description:

The list of group orders.

#### Relations:

### CorporateOrder

#### Description:

The list of corporate orders.

#### Relations:

### ExpenseType

#### Description:

The list of Expense types assigned to the company

#### Relations:

MANY-TO-ONE with ‘Expense’

### Corporate

#### Description:

The list of companies which provide corporate orders.

#### Relations:

MANY-TO-MANY with ‘Clients’

### CorporateCode

#### Description:

Information about company projects for corporate orders.

#### Relations:

MANY-TO-MANY with ‘ExpenseType’

MANY-TO-ONE with ‘Corporate’

### Currency

#### Description:

The list of currencies to be used

#### Relations:

### VAT

#### Description:

The list of VATs.

#### Relations:

### Page

#### Description:

The list of all static pages.

#### Relations:

MANY-TO-ONE with ‘Language’

### Navigation

#### Description:

The list of navigation items in menus.

#### Relations:

MANY-TO-ONE with ‘Page’

## Modules

### LogModule

#### Functionality

Provides interface to log activities to the database.

### CacheModule

#### Functionality

Provides interface to manage cache data.

#### Methods

**Set** – add data to cache.

**Get** – get data from cache.

**Flush** – clear all data from cache.

# Console project

This project implements functionality executed by cron.

## Controllers

### EmailNotificationController

#### Functionality

Provides functionality to send different user notifications.

#### Methods

**SendCustomEmailTo** – send custom emails to specific system users. The HTML template and the list of users should be provided.

**SendScheduledEmailTo** – send scheduled email notifications to selected users. The HTML template, the list of users and schedule should be provided.

#### Relations

Gateway EmailService – call REST API method SendCustomEmailTo,

Gateway EmailService – call REST API method SendScheduledEmailTo

### ReportController

#### Functionality

Provides functionality to prepare data for different reports in the system.

#### Methods

**PrepareDataForReports** – analyze data in the database and prepare info for reports.

#### Relations

No relations.