**RestaurantFinder**

**Makes Easy Restaurant Reservations**

System Design Document

16.07.2016

H. Selen Kozanoğlu – 210CS2022

Aslı Tuba Metin – 210CS2023

Arda Yazkan – 210SE2213

Emine Yıldız- 212CS2176

Prepared for

SE301 Software Engineering



Table of Contents

[1. Introduction 1](#_Toc433996772)

[1.1. Purpose of the System 1](#_Toc433996773)

[1.2. Design Goals 1](#_Toc433996774)

[1.3. Definitions, Acronyms, and Abbreviations 1](#_Toc433996775)

[1.4. References 1](#_Toc433996776)

[2. Current Software Architecture 1](#_Toc433996777)

[3. Proposed Software Architecture 1](#_Toc433996778)

[3.1. Overview 1](#_Toc433996779)

[3.2. System Decomposition 1](#_Toc433996780)

[3.3. Hardware Software Mapping 2](#_Toc433996781)

[3.4. Persistent Data Management 2](#_Toc433996782)

[3.5. Access Control and Security 2](#_Toc433996783)

[3.6. Global Software Control 2](#_Toc433996784)

[3.7. Boundary Conditions 2](#_Toc433996785)

[4. Subsystem Services 2](#_Toc433996786)

[5. References 2](#_Toc433996787)

SYSTEM DESIGN DOCUMENT[1]

# Introduction

RestaurantFinder is a web site for users who wants to make a easily reservation for the restaurants which they want. This project allows users easily find a restaurant and make reservation for it. In this project, users can access restaurants’ information and they can look theirs address, telephone, pictures and descriptions. Also, users can keep their past reservations and can follow them.

## Purpose of the System

The purpose of this project is to create a web site for making reservation for restaurants. But, users must register and login the system to make reservation. After login the system, they can make reservation, also they can cancel or edit their reservation. Further, this project have users profile page and users can enter their profile page and look past reservations. Also, they can discover new restaurants by category or city without logging the system and access their telephones, photos, addresses and descriptions. In addition, restaurant owners can arrange their restaurant page.

## Design Goals

The goal of the project is to satisfy the functional and non functional requirements as specified in the requirements analysis document. This project provide manage restaurants by categories and by cities. Users can search on the web site by a category name or a city name. After they login the system, they can make reservation to restaurants effectively and can look their past reservations on their profile page.

## Definitions, Acronyms, and Abbreviations

UI: User Interface

PHP: Hypertext Preprocessor

SDD: System Design Document

SDD is used to define interfaces between teams of developers and serve as a reference when architecture-level decisions need to be revisited.

Sections of the SDD are described:

* Current Software Architecture

The architecture of the system is described.

* Overview

The assignment of functionality to each subsystem is described.

* System Decomposition

The part is about the decomposition into subsystems and the responsibilities of each.

## Hardware Software Mapping

The part describe how subsystems are assigned to hardware and off-the-shelf components

* Persistent Data Management

The part is about the persistent data stored by the system and the data management infrastructure required for it.

## Access Control and Security

The user model of the system in terms of an access matrix is described.

## Global Software Control

The part describe how the global software control is implemented.

## Boundary Conditions

The start-up, shutdown, and error behavior of the system is described.

# Subsystem Services

The part is about the services provided by each subsystem.

* 1. **References**

-Requirements Analysis Document (26.06.2016)

# Current Software Architecture

Currently there are some software architecture for the Restaurant Finder. When the customer use this program, they can find every restaurant where he/she wants. Our program seems to be useful and friendly for our customers. Also we want this program helps the customers to find new restaurants.

Nowadays, there are lots of restaurant finder systems, but our program is not the same others. We developed the software for the using RestaurantFinder easily. On the other hand there 2 type search button. One of them is, entering city where the customers want to find in restaurant and other one is searching restaurant with using restaurant's name. Also we used database system for customers can reach their favorite restaurants.

RestaurantFinder program have restaurants information in its database. When the customers want to reach restaurant information such as telephone number, location etc. they can be reach. Because of this, customers can trust our program. It has not got any unreal informations about restaurants that are including our database system.

In our software about the RestaurantFinder, we used admin page to select restaurants that can have best quailities. We cannot select restaurants which have got any worse comments.

To conclude, our program, which is name RestaurantFinder, is very useful and understandable software.

# Proposed Software Architecture

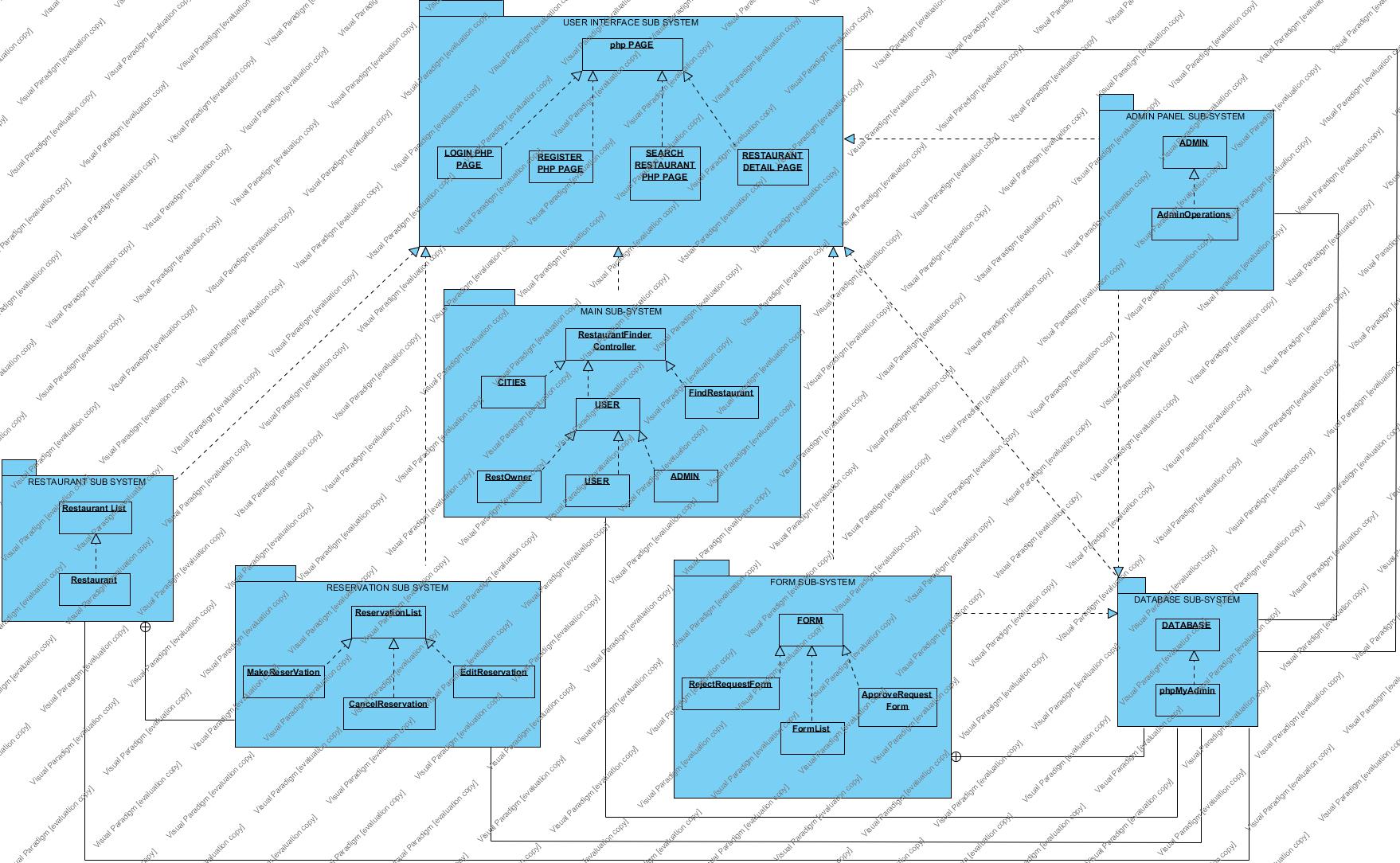
## Overview

Restaurant Finder is a restaurant reservation system which is based on satisfying customers and helping restaurant owners handle management. Restaurant Finder system is a web-based project. There are some people that are involved in this system these people called actor that are user, admin and restaurant owner. The user can make reservation and search restaurants. The restaurant owner can edit restaurant profile by changing information , restaurant photos ,color etc. The admin can add restaurants which comes from restaurant owner , and his/her edit profile. If the user wants to become a restaurant owner he/she needs to fill restaurant register form from main page. After he/she sends the form to the admin ,he becomes a part of Restaurant Finder, unless the admin rejects him/her form.



**-Birdseye View-**

## 3.2 System Decomposition



**-USER INTERFACE SUBSYSTEM**

User Interface Subsystem is responsible for all interaction between the users.  The UI is decomposed into a set of PHP pages loaded and displayed in a web-browser.  All entry of data, modification of data, and deletion of data will be handled directly through the UI.

**-MAIN SUB-SYSTEM**

Main Subsystem is responsible for interactions between the Restaurant Finder Controller class and the Cities, User, and Find Restaurant classes. The Main Subsystem is also responsible for the interactions between the Cities, User, Find Restaurant, and Restaurant Finder Controller classes to the Database Subsystem.

**-DATABASE SUB-SYSTEM**

Database Subsystem is responsible for all insertion, reading, updating and deletion of all data by all classes within the Main, Restaurant, Reservation, Form, Admin Panel Subsystems.  No data will be accessed in any way except through and by the Database Subsystem.  Each class from within the Subsystems described above are responsible for inserting, updating and modifying their own data by utilizing the corresponding methods belonging to the Database Class within the Database Subsystem.

**-RESTAURANT SUB-SYSTEM**

Restaurant Subsystem is responsible for all interactions involving restaurants.  All interactions with any restaurant will be handled through and by the restaurant list class. The Restaurant Subsystem is also responsible for the interactions between the Restaurant List and Restaurants classes to the Database Subsystem.  Both the Restaurant List and Restaurants class will each be responsible for utilizing the appropriate and necessary methods within the Database class to insert, read, update and delete all corresponding data from within each of the classes themselves.

**-RESERVASYON SUB-SYSTEM**

Reservation Subsystem is responsible for all interactions involving reservation.  All interactions with any reservation will be handled through and by the Reservation List class. The Reservation Subsystem is also responsible for the interactions between the Reservation List and Make Reservation, Cancel Reservation, Edit Reservation classes to the Database Subsystem.

**-FORM SUB-SYSTEM**

Form Subsystem is responsible for all interactions involving reservation.  All interactions with any form will be handled through and by the Form List class. The Form Subsystem is also responsible for the interactions between the Form List and Approve Request Form, Reject Request Form classes to the Database Subsystem.

**-ADMIN PANEL SUB-SYSTEM**

Admin Panel Subsystem is responsible for all interactions involving admin.  All interactions with any form will be handled through and by the Admin Operations class. The Admin Panel Subsystem is also responsible for the interactions between the Admin and Admin Operations classes to the Database Subsystem.

## 3.3 Hardware Software Mapping

The Hardware Software diagram describes the outline of development mapping schema about the restaurantFinder.com system. The component layout the restaurantFinder system.

**Database Server**

* Php MyAdmin

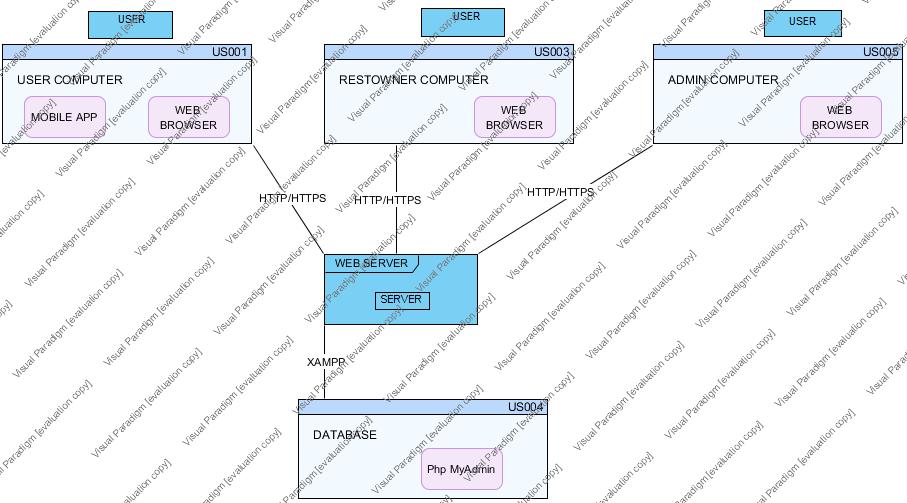
**RestaurantFinder**

* Web Server
* Apache

**User**

* Web Browser

RestaurantFinder is web-based reservation system for restaurants. RestaurantFinder is connecting to web server when user tries to visit and login in the system. Users that admin, restaurant owner and normal user which just make a reservation , use the restaurantfinder system. These actors communicate with restaurantfinder system with http-https protocol web browser in their personal computers. Furthermore restaurantFinder database server using php my admin for communicating.

****

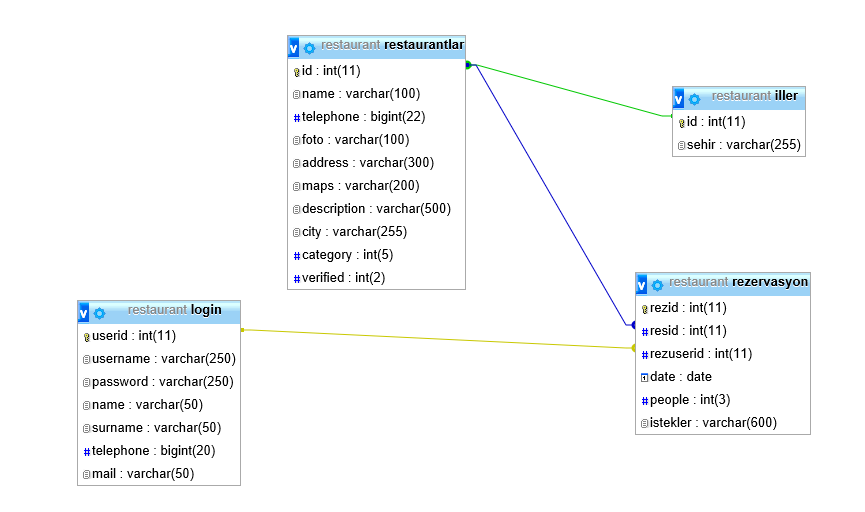
## Persistent Data Management

RestaurantFinder is using database to store all restaurants and customers informations. Also database should store cities. They must be relation in each other.

RestaurantFinder's database name is restaurant. We used 4 tables such as iller, login, restaurantlar and rezervasyon. Also maybe we can change some variables on the database.

In this program, we use phpMyAdmin. It is very useful interface to create a database.

We use insert, delete, search etc. methods.



RestaurantFinder's database keeps relations. We use id for making tables unique.

**Description of the Tables**

**Login:** Login table where we store customers' information. It has 7 fields which are userid, username, password, name, surname, telephone, mail. All these fields helps the customers to access their page.

**Restaurantlar:** Restaurantlar table where we store restaurants' information has 10 fields which are id, name, telephone, foto, address, maps, description, city, category, verified. All these fields helps the system to access restaurant's information.

**İller:** İller table where we store cities' name has 2 fields which are id and sehir. 'sehir field helps the relation in the system. Customers use the 'sehir' to find the restaurants.

**Rezervasyon:** Rezervasyon table where we use for the main topic has 6 fields which are rezid, resid, rezuserid, date, people, istekler.

## 3.5 Access Control and Security

The Access control Matrix for the Restaurant Owner System is as follows:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Login** | **Select City** | **List Restaurants** | **Edit Profile** | **Make Reservation** | **Restaurant Register Form** | **Restaurant Requests** |
| User | **yes** | **selectCity()** | **ListRestaurant()** | **yes** | **makeReservation()** | **fiilForm() sendForm()** | **-** |
| Restaurant Owner | **yes** | **-** |  | **yes** | **-** | **-** | **-** |
| Admin | **yes** | **-** |  | **yes** | **-** | **-** | **seeRequests() acceptRequest() rejectReq()** |

**Authentication and Security:**

Restaurant Finder system is designed to the user to make reservation. For this reason, systems need some personal information of the user such as the user’s name ,surname and phone number. However, this user’s personal information should not be seen by someone else.

While we were designing and programming the system, we decided to keep this saved article everyone on their phone, not the remote. This provides security to the project. Also, we used the three tier software architecture. When one tier fails there is no data loss, because you are always secure by accessing the other tier.

**Authorization:**

The system has database connection for first login. According to login user types system decides which actor will login the site by looking the actor type from database table. System has three sub database classes for actors which are database Restaurant Owner, User and Admin. Each class has specific attributes and methods for actor.. This actor login system is controlled with usertype field in database.

## Global Software Control

Restaurant Finder system is based on **object-oriented** approach. This approach makes it easy to maintain and modify existing code as new objects can be created In general, the system will be need to do simultaneous tasks at a time, so we had selected **Decentralized Control Approach**

## Boundary Conditions

There are several exceptions in our system. The most important thing is the user has to have the internet connection to reach the site and make reservation. If there is no connection, the user can’t load the main page. At this condition, system displays a message shows an exception about connectionless. Also, when the system is closed by system admin any user that currently connection in to the system will be disconnected and no user can connection the system until it is opened again.

Startup: go to system URL and login

Shut Down: click log out and close browser

Error Conditions:

* Logging in:
  + Username or password field can are blank.
  + Username is not a 5 digit decimal number.
  + Password is not 5 characters long.
  + Password and username don’t match.
  + Username is wrong or does not exist.
  + The welcome screen does not appear after logging in.
* User settings
  + User is unable to change certain settings or changes don’t reflect.
  + Between the time of editing and updating, the system crashes.

# Subsystem Services

-USER INTERFACE SUBSYSTEM

* Takes user input
* Provides access login page, register page, search restaurant page, restaurant detail page

-MAIN SUB-SYSTEM

- Receives user queries through the UI subsystem

* Sends system responses to the UI subsystem
* Handles cities.
* Provides find restaurant.

-DATABASE SUB-SYSTEM

* Receive update requests from different subsystems
* Sends object information to different requesting controllers

-RESTAURANT SUB-SYSTEM

* Provides restaurants address, telephone, description information
* Provides restaurants city and category information

-RESERVASYON SUB-SYSTEM

* Provides make reservation
* Provides cancel reservation
* Provides edit reservation

-FORM SUB-SYSTEM

* Provides approve request form
* Provides reject request form

-ADMIN PANEL SUB-SYSTEM

* Provides admin operaions

# References

1. SE301 presentation documents

# <https://www.youtube.com/watch?v=TAfccACvjXw> - Boundary, Entity & Control Elements

