CS 353 – Database Systems

Project Final Report

Online Professional Hiring System

Group 39

Bora Ecer

Alp Ege Baştürk

Deniz Alkışlar

Buğra Aydın

14.5.2018

1. Project Description and System Overview

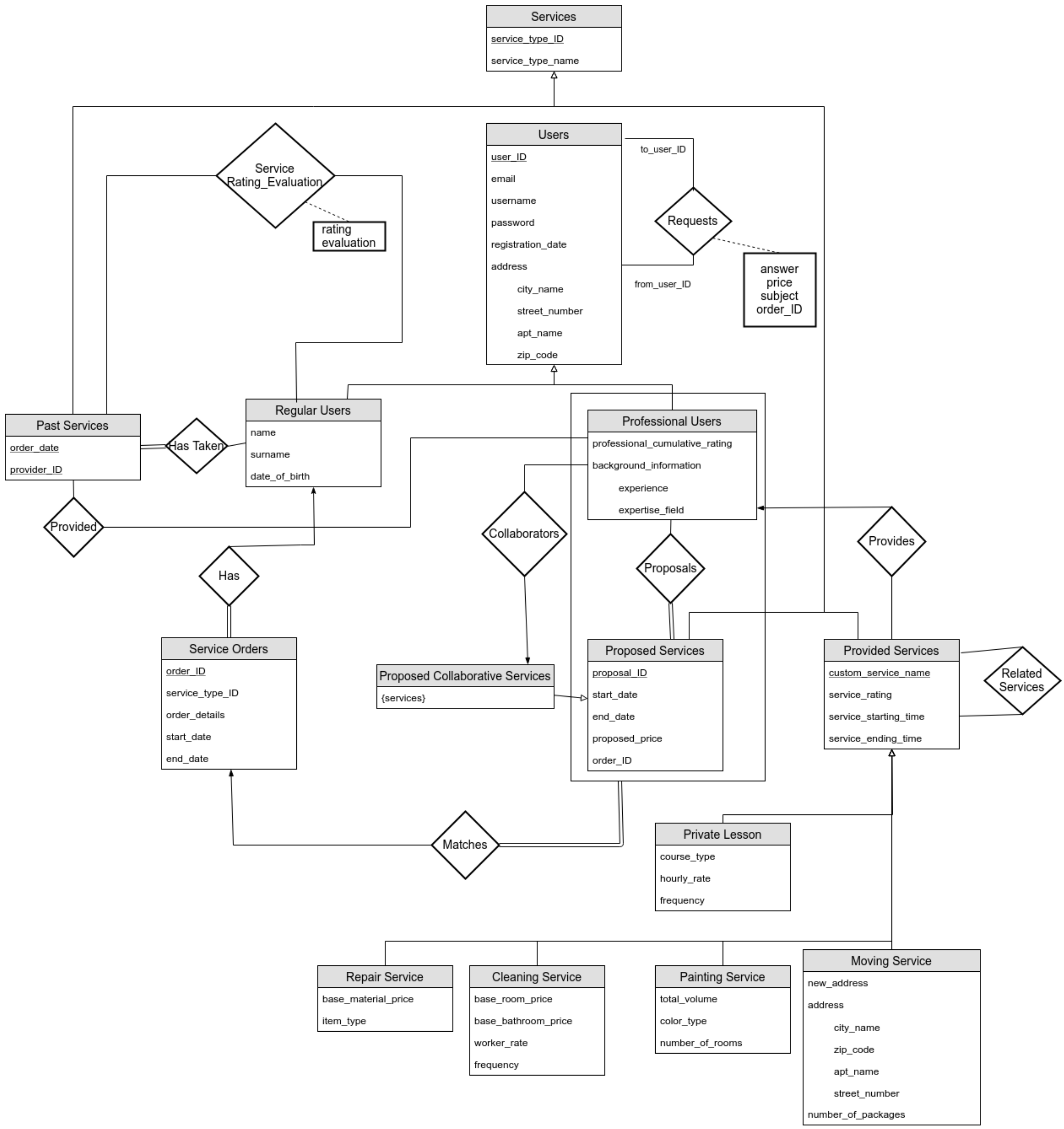
Portakal is a web-based online professional hiring system application that brings professionals and customers together. The system consists of two main user types; regular users (RU) and professional users (PU). System stores information about orders, order types, dates of proposals from professionals, duration of provided services and relations of proposals and service requests. Users are shown respective homepages after they logged in to the system. In this window, they can use the manage account button to modify their account details. Users can navigate, see notifications by using buttons on the navigation bar. In the create service request window RUs can create new requests with the specified type and time period they want the service or modify previous requests. They can see the proposals related to that request by selecting it. Also, they can click on the view proposals button on the main page to see all requests and proposals in a single window. RUs can click matches button on both of these pages to see best matching proposal in time period. Also, RU can click add button on the view proposals page to add them to comparison list. This will show them the details of the proposal and professional who is providing that service. This feature can show two details of two proposals side-by-side. User can use accept buttons on the respective windows in order to accept the proposal. This sends a notification message to professional.

PUs also have view requests and view proposals pages. PUs see requests which are related to services provided by them. For example, a professional who is providing painting services cannot see a repair request. PUs can propose to requests which they want by giving time period and price information. If there is an existing proposal, they can modify it instead of proposing. Also they can press view proposals button to see their proposals. In this page, they can modify their proposals. System allows professionals to register their services. These have type and a custom name. For example someone can provide a Repair service with Super Repair name.

Contributions

- **Alp Ege Baştürk:** Wrote the Java code, php and sql parts of the following: create_proposal_pro.php, view_service_requests_reg.php, view_service_requests_pro.php, view_proposals_pro.php, register_service.php, view_services_pro.php create_service.php, button references on logon_pro.php and logon_reg.php.
- **Bora Ecer:** Updated Java code and wrote php and sql parts of the following: config.php, create_proposal_pro.php, evaluate_service.php, notification menus in logon_pro.php and logon_reg.php, match.php, modify_proposal.php, modify_registered_service.php, modify_service_request.php, past_services.php, view_service_requests_reg.php, view_service_request_prof.php also edited view_proposals_pro.php and view_proposals_reg.php.
- **Buğra Aydın** wrote the php parts of the following: homepage.php, login.php, signup_pro.php, signup_reg.php, logon_pro.php, logon_reg.php, manage_pro.php, manage_reg.php. Implemented the logout operation. Wrote SQL queries for inserting and updating to several tables during sign up and manage account operations. Worked collaboratively on the UI with Deniz Alkışlar to implement several operations and improve user friendliness.
- **Deniz Alkışlar** built graphical user interface of each page according to mockups. Implemented the compare_list.php and contributed some other PHP codes.
- Further details can be found on the GitHub repository of the project.

2. Final E/R Model



3. Final Relation Schemas

Primary keys are underlined or stated explicitly. Foreign keys are stated explicitly.

3.1 Users

Relational Model:
Users(user_ID, password, email, username, city_name, street_number, apt_name, zip_code)

PRIMARY KEY: user_ID

3.2 Regular Users

Relational Model:
Regular Users(user_ID, name, surname, date_of_birth)

FOREIGN KEY: (user_ID) REFERENCES users(user_ID)

3.3 Professional Users

Relational Model:
Professional Users(user_ID, experience, expertise_field)

FOREIGN KEY: (user_ID) REFERENCES users(user_ID)

3.4 Services

Relational Model:
Services(service_type_ID, service_type_name)

PRIMARY KEY: service_type_ID

3.5 Past Services

Relational Model:
Past Services(service_type_ID, order_date, provider_ID)

FOREIGN KEY: (service_type_ID) REFERENCES services(service_type_ID)
FOREIGN KEY: (provider_ID) REFERENCES professional_users(user_ID)

3.6 Service Orders

Relational Model:
Service Orders(order_ID, requester_ID, service_type_ID, order_details, start_date, end_date)

FOREIGN KEY: (service_type_ID) REFERENCES services(service_type_ID)

3.7 Requests

Relational Model:
Requests(to_user_ID, from_user_ID, order_ID**,** subject, price, answer)

FOREIGN KEY: (to_user_ID) REFERENCES users(user_ID)
FOREIGN KEY: (to_user_ID) REFERENCES users(user_ID)
FOREIGN KEY: (order_ID) REFERENCES service_orders(order_ID)

3.8 Proposed Services

Relational Model:
Proposed Services(proposal_ID, service_type_ID, start_date, end_date, proposed_price, order_ID)
PRIMARY KEY: proposal_ID
FOREIGN KEY: (service_type_ID) REFERENCES services(service_type_ID)
FOREIGN KEY: (order_ID) REFERENCES service_orders(order_ID)

3.9 Proposed Collaborative Services

Relational Model:
Proposed Collaborative Services(proposal_ID, services)

FOREIGN KEY: (proposal_ID) REFERENCES proposed_services(proposal_ID)

3.10 Provided Services

Relational Model:
Provided Services(service_type_ID, custom_service_name, service_rating, service_starting_date, service_ending_date)

PRIMARY KEY: service_type_ID, custom_service_name
FOREIGN KEY: (service_type_ID) REFERENCES services(service_type_ID)

3.11 Service Ratings Evaluations

Relational Model:

Service Ratings Evaluations(user_ID, service_type_ID, order_date, provider_ID, rating, evaluation)

FOREIGN KEY: (user_ID) REFERENCES regular_users(user_ID)
FOREIGN KEY: (service_type_ID) REFERENCES services(service_type_ID)
FOREIGN KEY: (provider_ID) REFERENCES professional_users(user_ID)

3.12 Has Taken

Relational Model:
Has Taken(user_ID, service_type_ID, order_date, provider_ID)

FOREIGN KEY: (user_ID) REFERENCES regular_users(user_ID)
FOREIGN KEY: (service_type_ID) REFERENCES services(service_type_ID)
FOREIGN KEY: (provider_ID) REFERENCES professional_users(user_ID)

3.13 Collaborators

Relational Model:
Collaborators(proposal_ID, user_ID)

FOREIGN KEY: (proposal_ID) REFERENCES proposed_services(proposal_ID)
FOREIGN KEY: (user_ID) REFERENCES professional_users(user_ID)

3.14 Proposals

Relational Model:
Proposals(professional_ID, proposal_ID)

FOREIGN KEY: (professional_ID) REFERENCES professional_users(user_ID)
FOREIGN KEY: (proposal_ID) REFERENCES proposed_services(proposal_ID)

3.15 Provides

Relational Model:
Provides(user_ID, service_type_ID, custom_service_name)

FOREIGN KEY: (user_ID) REFERENCES professional_users(user_ID)
FOREIGN KEY: (service_type_ID,custom_service_name) REFERENCES provided_services (service_type_ID,custom_service_name)

3.16 Matches

Relational Model:
Matches(oder_ID, porposal_ID)

FOREIGN KEY: (order_ID) REFERENCES service_orders (order_ID)
FOREIGN KEY: (proposal_ID) REFERENCES proposals (proposal_ID)

3.17 Provided

Relational Model:
Provided(service_type_ID, order_date, provider_ID, served)

FOREIGN KEY: (service_type_ID) REFERENCES past_services(service_type_ID)
FOREIGN KEY: (provider_ID) REFERENCES professional_users(user_ID)

3.18 Private Lesson

Relational Model:
Private Lesson(service_type_ID, custom_service_name, course_type, hourly_rate, frequency)

FOREIGN KEY: (service_type_ID, custom_service_name) REFERENCES provided_services(service_type_ID, custom_service_name)

3.19 Repair Service

Relational Model:
Repair Service(service_type_ID, custom_service_name, base_material_price, item_type)

FOREIGN KEY: (service_type_ID, custom_service_name) REFERENCES provided_services(service_type_ID, custom_service_name)

3.20 Cleaning Service

Relational Model:
Cleaning Service(service_type_ID, custom_service_name, base_room_price, base_bathroom_price, worker_rate, frequency)

FOREIGN KEY: (service_type_ID, custom_service_name) REFERENCES provided_services(service_type_ID, custom_service_name)

3.21 Painting Service

Relational Model:
Painting Service(service_type_ID, custom_service_name, total_volume, color,type, number_of_rooms)

FOREIGN KEY: (service_type_ID, custom_service_name) REFERENCES provided_services(service_type_ID, custom_service_name)

3.22 Moving Service

Relational Model:
Moving Service(service_type_ID, custom_service_name, new_city_name, new_zip_code, new_apt_name, new_street_number, city_name, zip_code, apt_name, street_number)

FOREIGN KEY: (service_type_ID, custom_service_name) REFERENCES provided_services(service_type_ID, custom_service_name)

4. Implementation Details

The project consists two parts; database and website. MySQL is used for the database implementation and queries. We have written Java code to automate database creation and population. We have used HTML, JavaScript and CSS for user interface. Bootstrap Framework version 3 was used in order to provide a better user interface and make development easier. PHP is used to implement operations. PHP scripts were used in the pages for programming functionality. SQL queries and operations were implemented in PHP. Also, HTML code was created programmatically by PHP scripts in HTML codes when dynamic pages are needed.

We have faced problems with inserting to and deleting from multiple tables. We were able to solve some of them with triggers such as inserting to past services. However, we have explicitly ordered queries in a transaction and stated when to commit in some of them. Explicit transaction and commit queries also helped with possible problems with non atomic operations which could cause problems. Create_proposals_pro.php has such queries. Also queries must be separated to different strings because of limitations of mysqli.

Another problem was implementing connections between professionals and regulars and professionals and other professionals. Initially we have used *mail()* function provided by PHP to send emails to respective targets. We have implemented queries for this solution. However we could not set the server and also decided to implement a notification system inside our app, since it would be more convenient for users. This decision caused requests table to be added to the database to store information about messages.This notification system is capable of sending basic notification messages as well as sending invitation messages. Invitation messages are used by the professionals to propose a collaborative service, basic notification messages are used when a regular user accepts a proposal.

5. Advanced Database Components

5.1 Triggers

```
CREATE TRIGGER past_insert " +
            "AFTER INSERT ON past_services " +
            "FOR EACH ROW " +
            "INSERT INTO provided (service_type_ID, order_date, provider_ID, served) " +
            "VALUES (NEW.service_type_ID, NEW.order_date , NEW.provider_ID, '0')
```

5.2 Reports

5.2.1 Matching Proposals

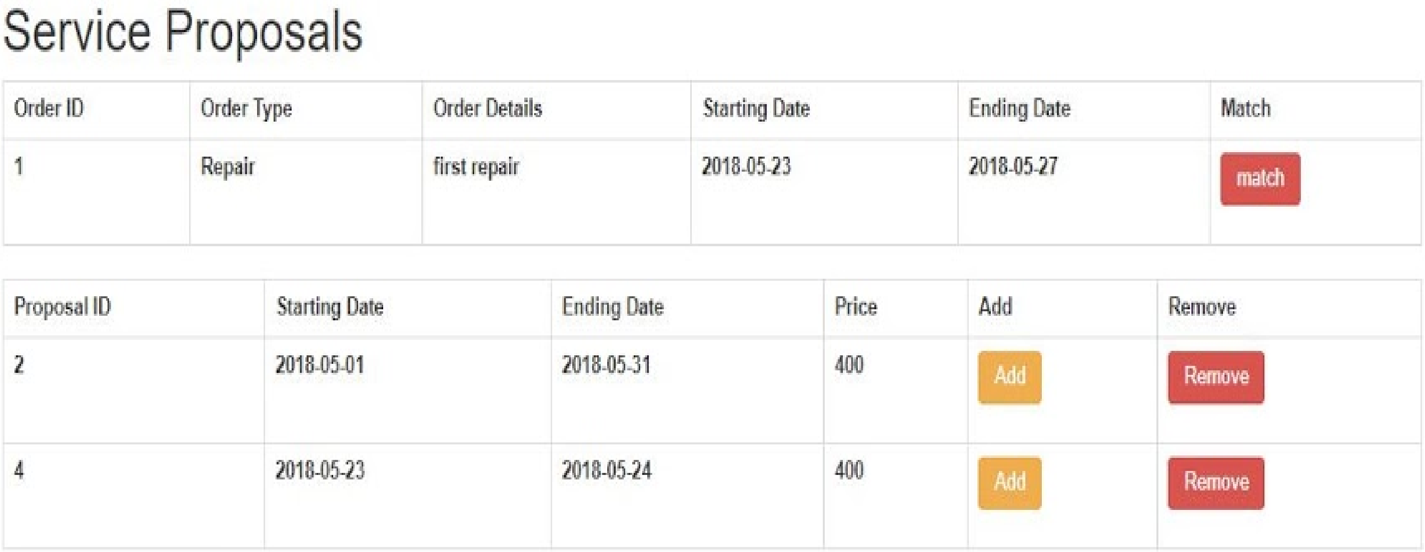
Our system provides regular users to see all of the proposals done to their service orders, regardless of whether the time intervals match, or not. We also provide proposal matching, with allows regular users to eliminate the proposals that do not match the service order dates. This matching also includes proposals with shorter date interval than the specified date interval in the service order.

```
CREATE TEMPORARY TABLE temp_proposal (
              proposal_ID INT PRIMARY KEY,
              p_start DATE,
              p_end DATE,
              proposed_price INT
         );
CREATE TEMPORARY TABLE temp_date (
                 proposal_ID INT PRIMARY KEY,
                 proposed_price INT,
                 p_start DATE,
                 p_end DATE,
                 start_date DATE,
                 end_date DATE
         );

INSERT INTO temp_proposal(proposal_ID, p_start, p_end, proposed_price)
                   SELECT proposal_ID, start_date AS p_start, end_date AS p_end,
proposed_price
                   FROM proposed_services
                   WHERE order_ID = $order_id;
INSERT INTO temp_date(proposal_ID, proposed_price, p_start, p_end, start_date,
end_date)
          SELECT proposal_ID, proposed_price, p_start, p_end, start_date, end_date
```

```
              FROM temp_proposal NATURAL JOIN service_orders so
              WHERE so.order_ID = $order_id;
SELECT proposal_ID, proposed_price, p_start, p_end, DATEDIFF(p_start, p_end) AS
prop_time, DATEDIFF(start_date, end_date) AS serv_time,
             DATEDIFF(p_start, start_date) AS starts, DATEDIFF(p_end, end_date) AS ends
             FROM temp_date
             HAVING ABS(prop_time) < ABS(serv_time) OR (ABS(ends) < 3 AND ABS(starts) <
3 );
```

In the following screenshot, the unmatched proposals can be seen, proposal with ID 2 requires 26 days more than what was determined by the regular user. So, the regular user might not want to see such proposals, in order to make his/her comparison easier. So, when the match for the service order is called, it eliminates such redundant proposals, as it can be seen in the second screenshot.

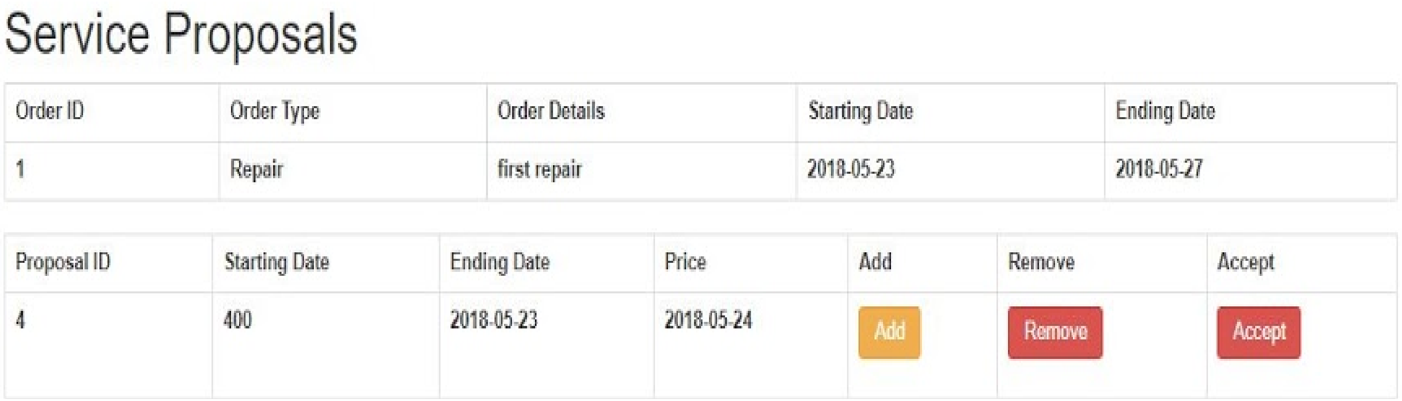


Service Proposals

Order ID	Order Type	Order Details	Starting Date	Ending Date	Match
1	Repair	first repair	2018-05-23	2018-05-27	match

Proposal ID	Starting Date	Ending Date	Price	Add	Remove
2	2018-05-01	2018-05-31	400	Add	Remove
4	2018-05-23	2018-05-24	400	Add	Remove

6. Sample Outputs



Service Proposals

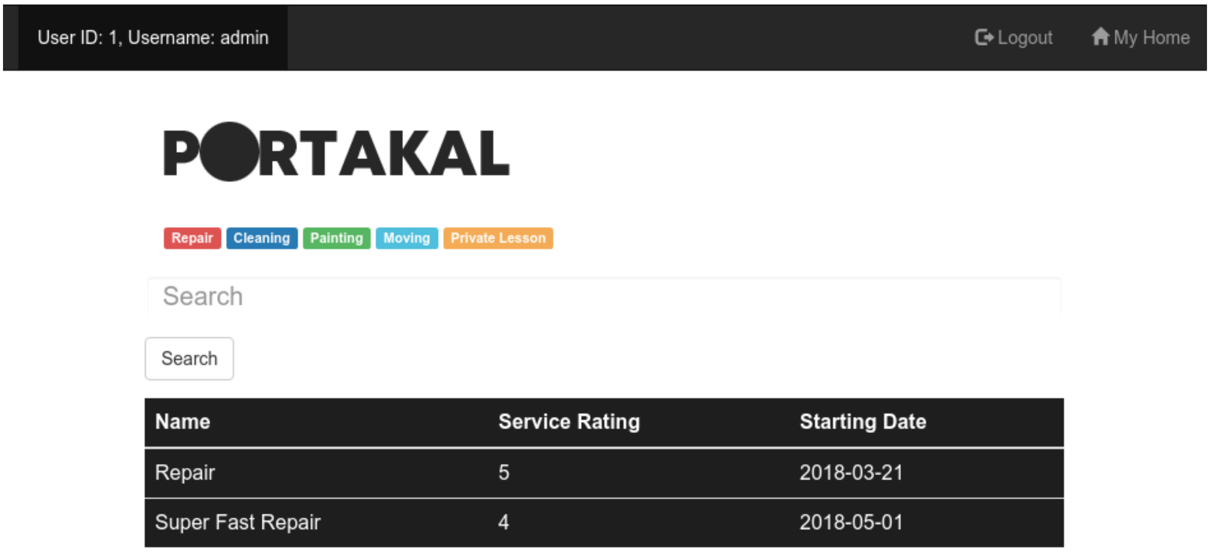
Order ID	Order Type	Order Details	Starting Date	Ending Date
1	Repair	first repair	2018-05-23	2018-05-27

Proposal ID	Starting Date	Ending Date	Price	Add	Remove	Accept
4	400	2018-05-23	2018-05-24	Add	Remove	Accept

This is the output of the match query. The query considers extra details and produces new output for the request.

7. User’s Manual

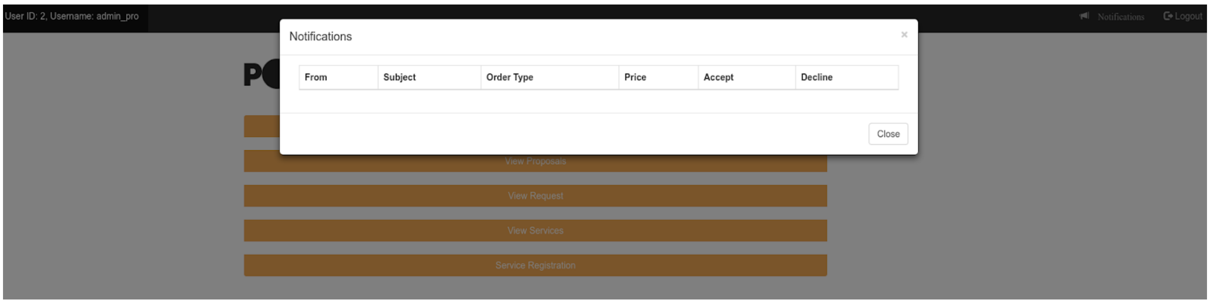
Home Page



Name	Service Rating	Starting Date
Repair	5	2018-03-21
Super Fast Repair	4	2018-05-01

In this page users can search for provided services. Also they can use Logout and My Home buttons on top. Clicking the ‘Portakal’ logo redirects users to the homepage in every screen. My Home button directs to respective home page mentioned in 7.1.1

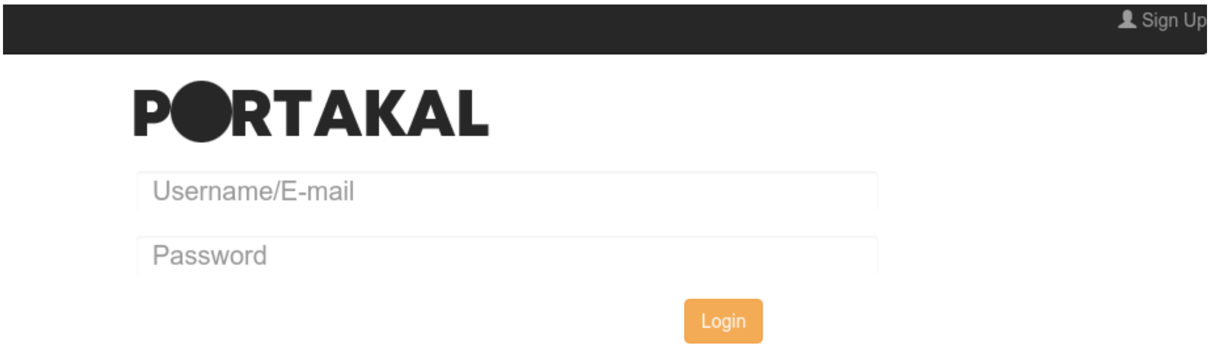
View Notifications



From	Subject	Order Type	Price	Accept	Decline

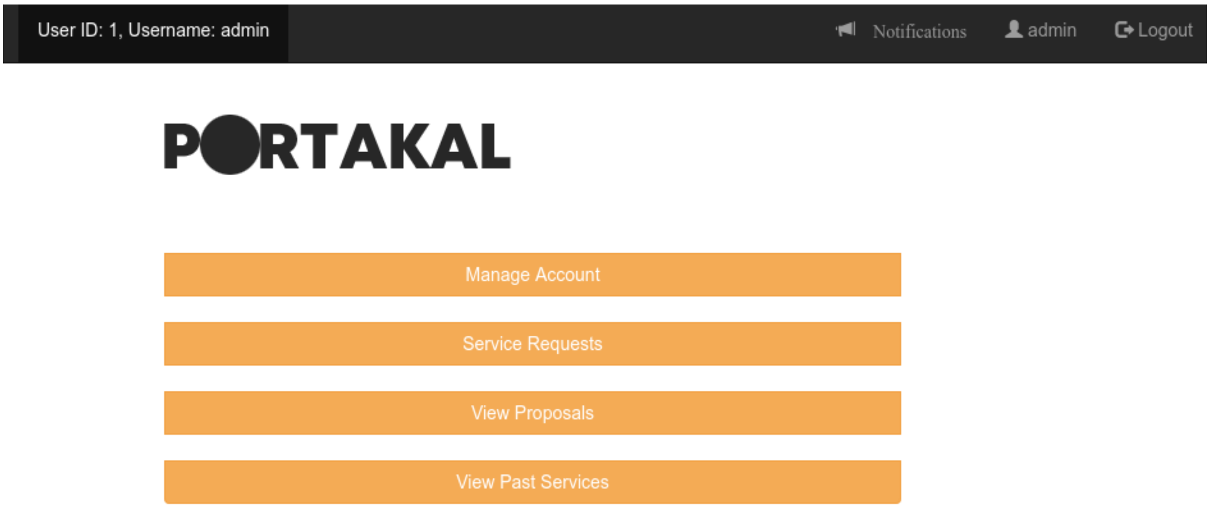
Users can see notifications in such a window.

Login Page



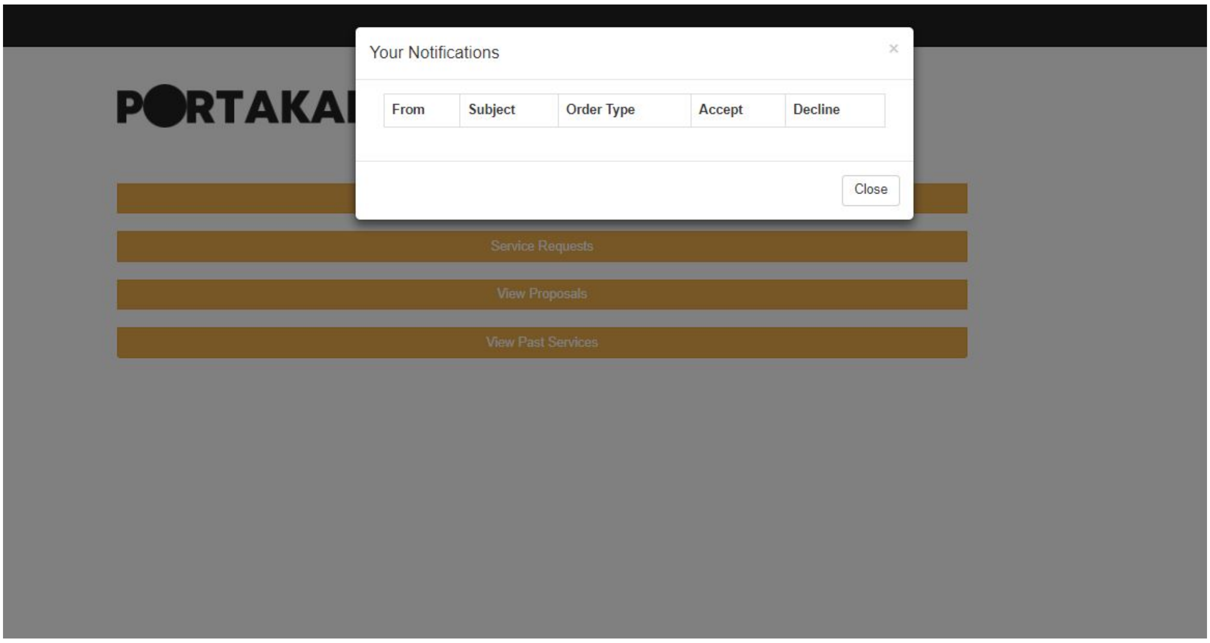
7.1 Regular Users

Log on Page for Regular User



In this page, regular user are provided option to manage their account, go to service requests page, view proposals page and view past services page. They can also see their User ID and Username. In the top right, navigation bar provides them the options to view notifications and logout.

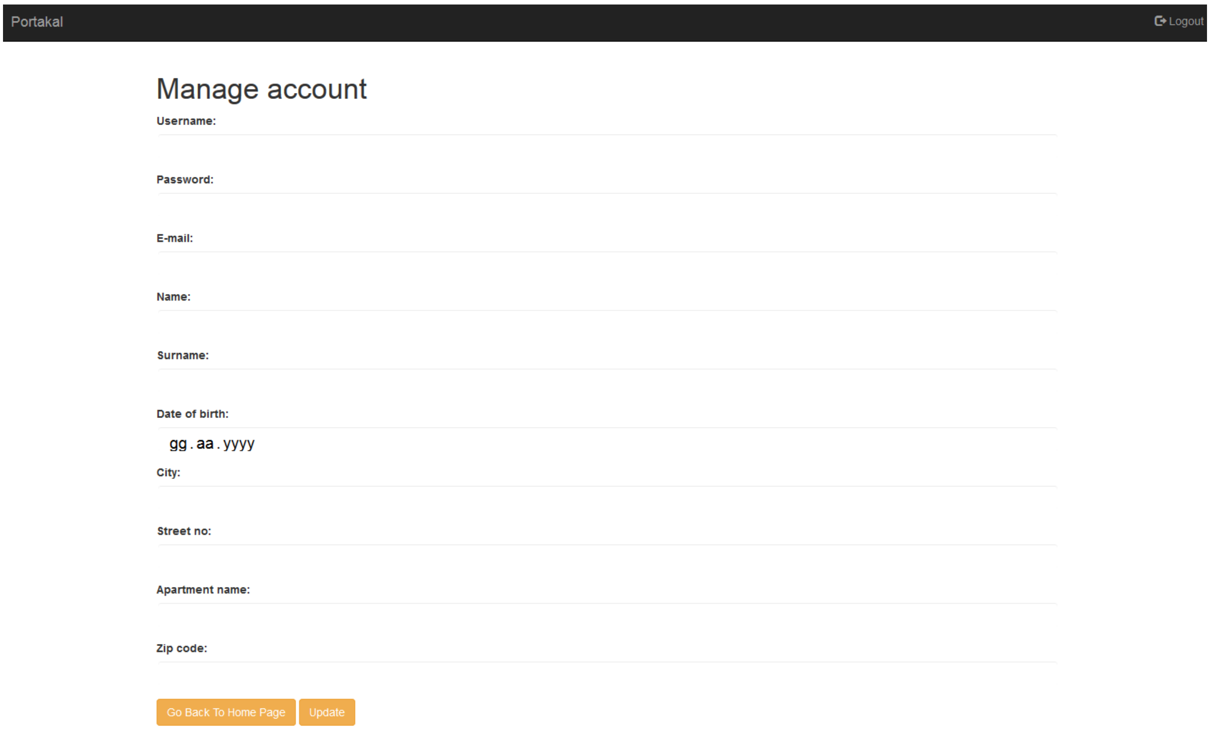
Notifications for Regular Users



Your Notifications

From	Subject	Order Type	Accept	Decline

Manage Account for Regular Users



Manage account

Username:

Password:

E-mail:

Name:

Surname:

Date of birth:

gg . aa . yyyy

City:

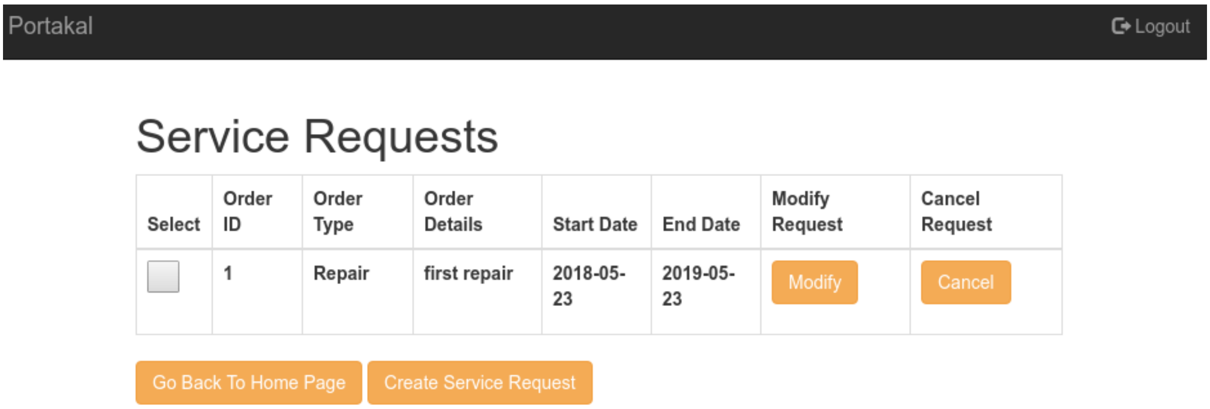
Street no:

Apartment name:

Zip code:

Regular users can update their account information from this screen by providing valid information. Back button is also provided.

View Service Requests for Regular Users

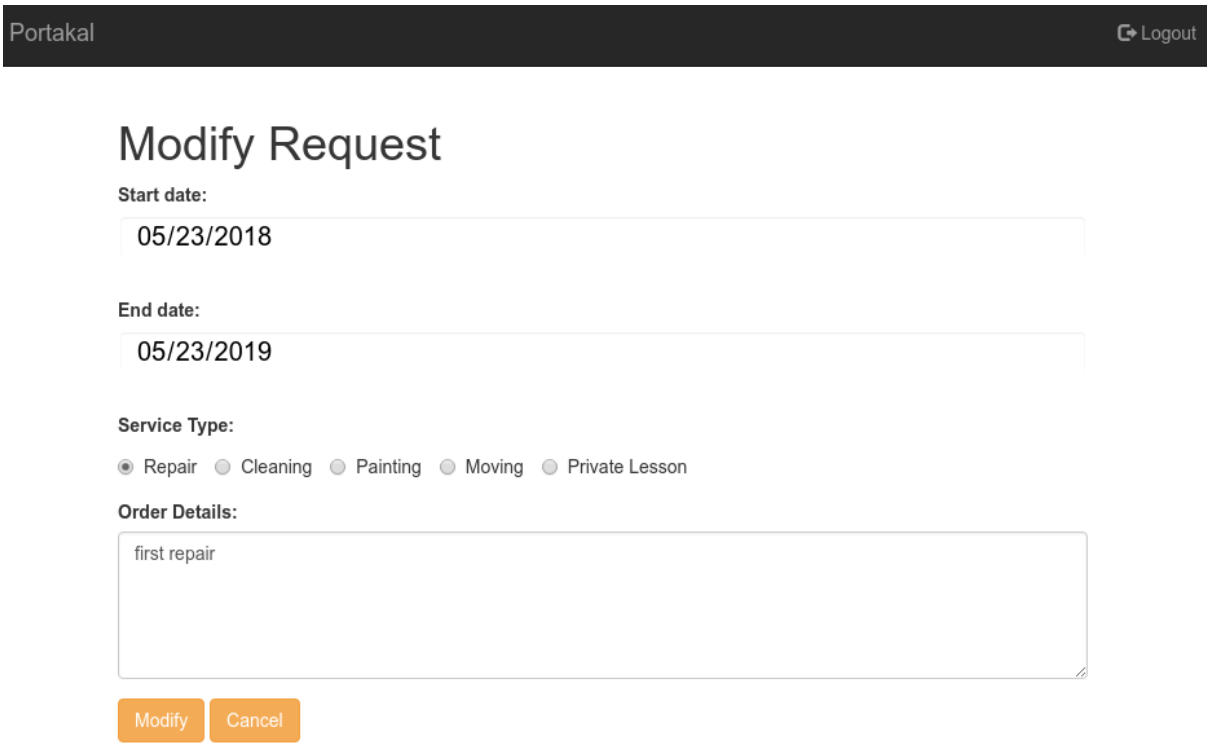


Service Requests

Select	Order ID	Order Type	Order Details	Start Date	End Date	Modify Request	Cancel Request
□	1	Repair	first repair	2018-05-23	2019-05-23	Modify	Cancel

RUs can see their requests, modify and cancel them in this page. Also, they can click in the button on the Select column to see proposals for the selected request.

Modify Request for Regular Users



Modify Request

Start date: 05/23/2018

End date: 05/23/2019

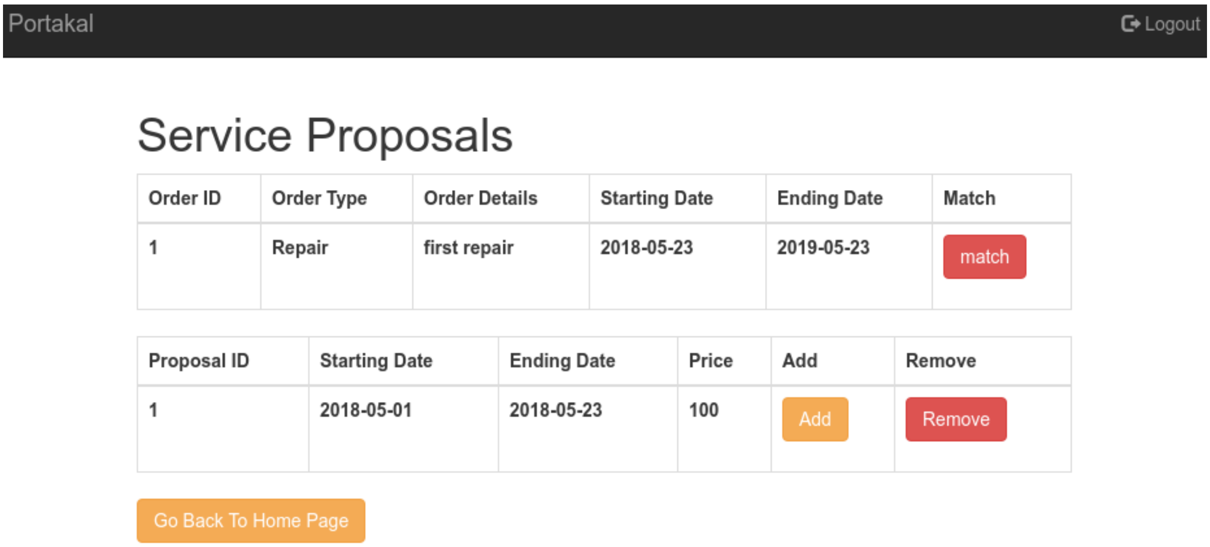
Service Type:

● Repair ○ Cleaning ○ Painting ○ Moving ○ Private Lesson

Order Details:

first repair

View Proposal for Regular Users



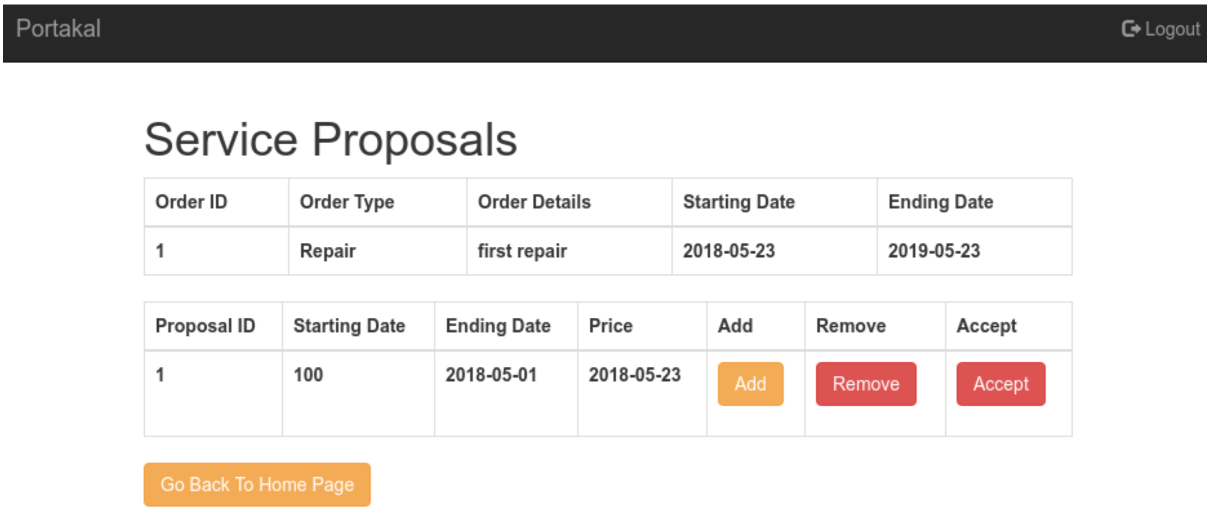
Service Proposals

Order ID	Order Type	Order Details	Starting Date	Ending Date	Match
1	Repair	first repair	2018-05-23	2019-05-23	match

Proposal ID	Starting Date	Ending Date	Price	Add	Remove
1	2018-05-01	2018-05-23	100	Add	Remove

RUs can see request and related proposals. Match button shows best matching proposals. Also add and remove are used to add proposals to the comparison list. Similar page can be accessed by the View Proposals button on the home page however, it shows all requests and proposals. Match functionality again works similarly in that case.

Matched Proposals for Regular Users



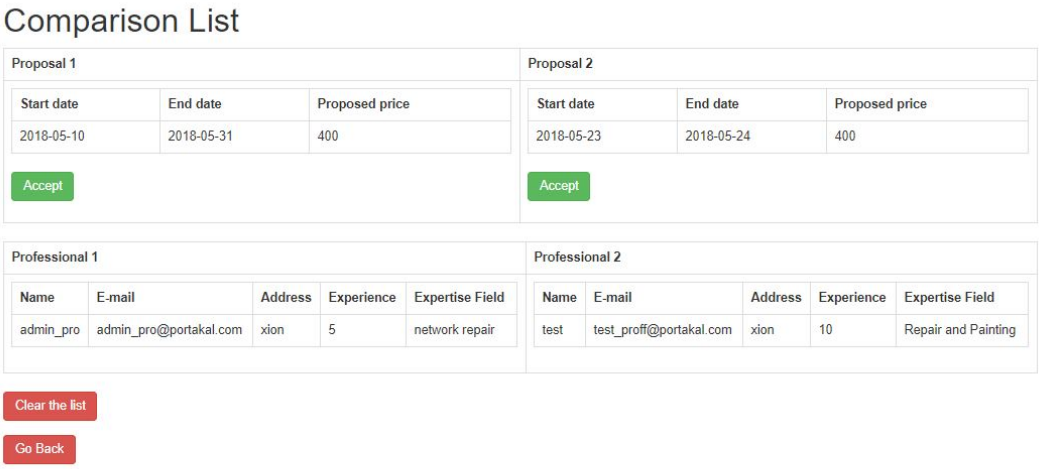
Service Proposals

Order ID	Order Type	Order Details	Starting Date	Ending Date
1	Repair	first repair	2018-05-23	2019-05-23

Proposal ID	Starting Date	Ending Date	Price	Add	Remove	Accept
1	100	2018-05-01	2018-05-23	Add	Remove	Accept

This page shows matched proposals. User can add them to comparison list or accept one of them.

Comparison for Regular Users



Comparison List

Proposal 1

Start date	End date	Proposed price
2018-05-10	2018-05-31	400

Proposal 2

Start date	End date	Proposed price
2018-05-23	2018-05-24	400

Professional 1

Name	E-mail	Address	Experience	Expertise Field
admin_pro	admin_pro@portakal.com	xion	5	network repair

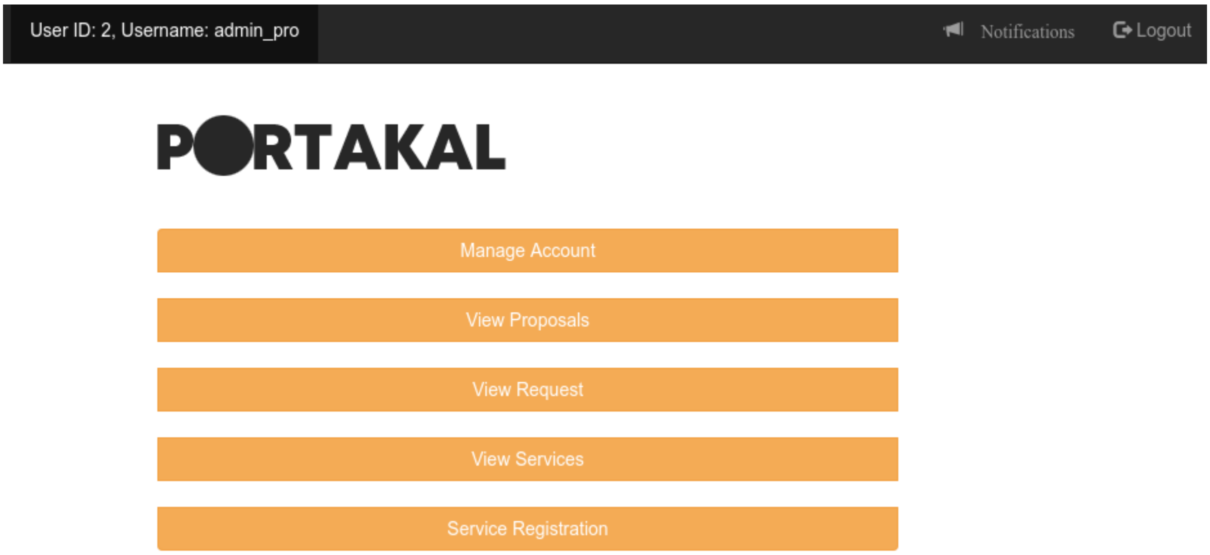
Professional 2

Name	E-mail	Address	Experience	Expertise Field
test	test_proff@portakal.com	xion	10	Repair and Painting

In this page details of two proposals and their providers are shown side-by-side.

7.2 Professional Users

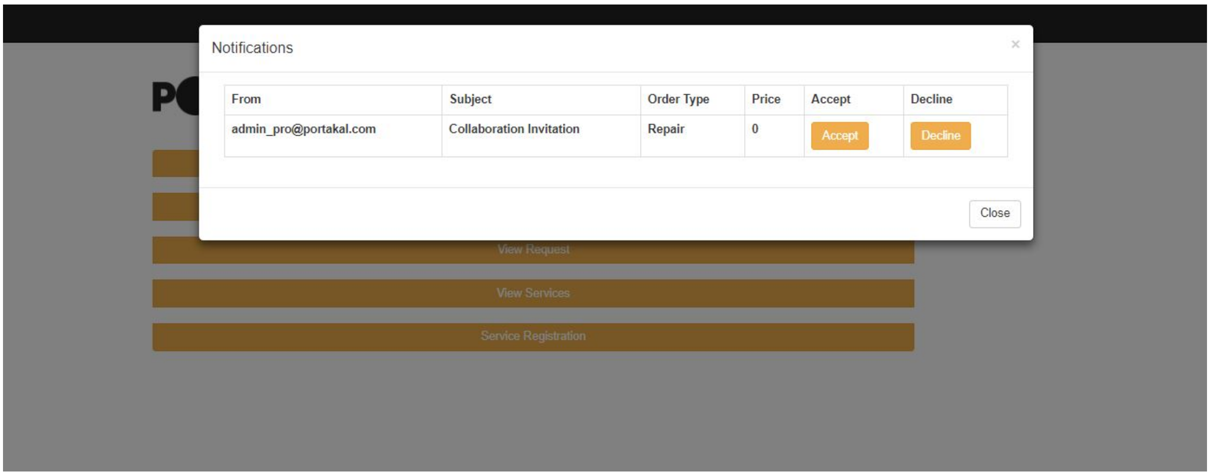
Logon Page for Professional Users



User ID: 2, Username: admin_pro

This is the home page for professional users. They can see notifications and use the buttons for respective functionality.

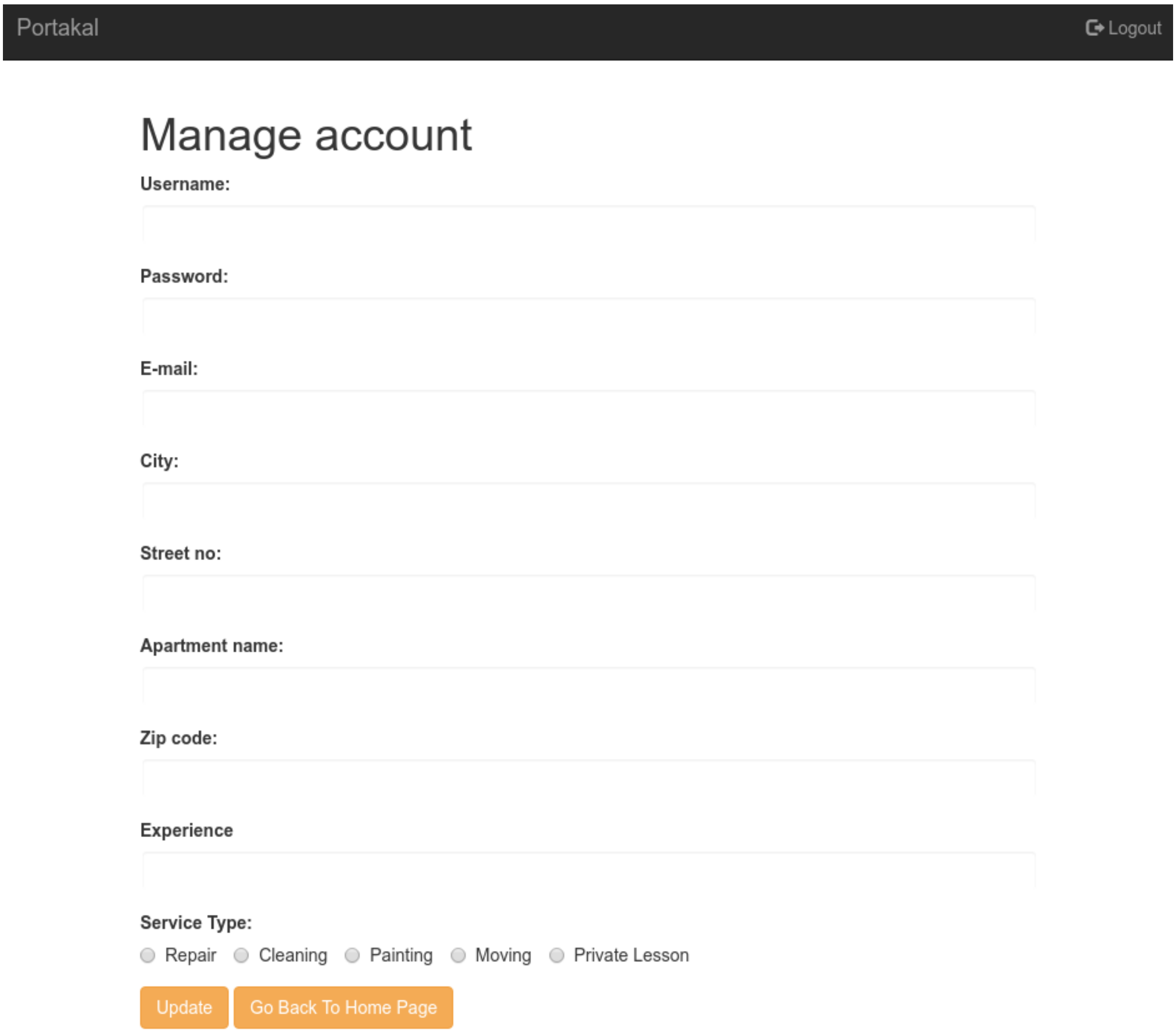
Notifications for Professional Users



Notifications

From	Subject	Order Type	Price	Accept	Decline
admin_pro@portakal.com	Collaboration Invitation	Repair	0	Accept	Decline

Manage Account for Professional Users



Portakal

Logout

Manage account

Username:

Password:

E-mail:

City:

Street no:

Apartment name:

Zip code:

Experience

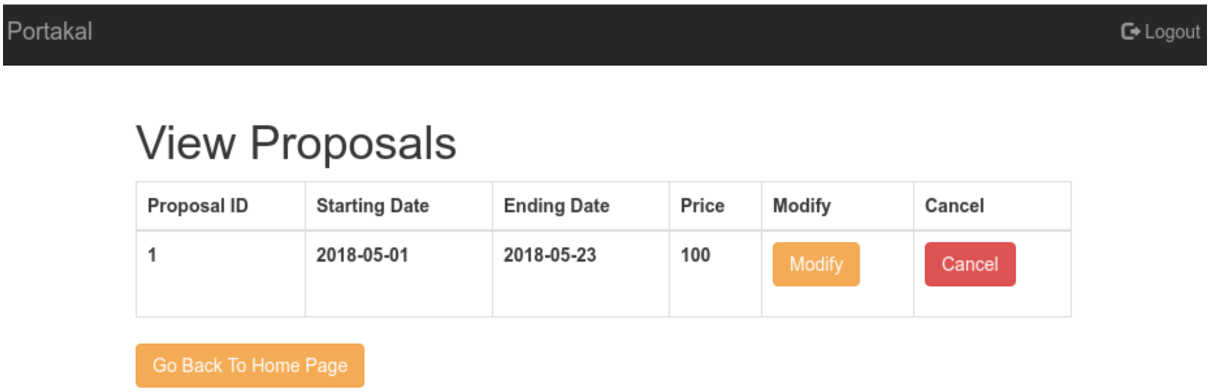
Service Type:

○ Repair ○ Cleaning ○ Painting ○ Moving ○ Private Lesson

Update | Go Back To Home Page

This page allows professional users to modify their account information.

View Proposal for Professional Users

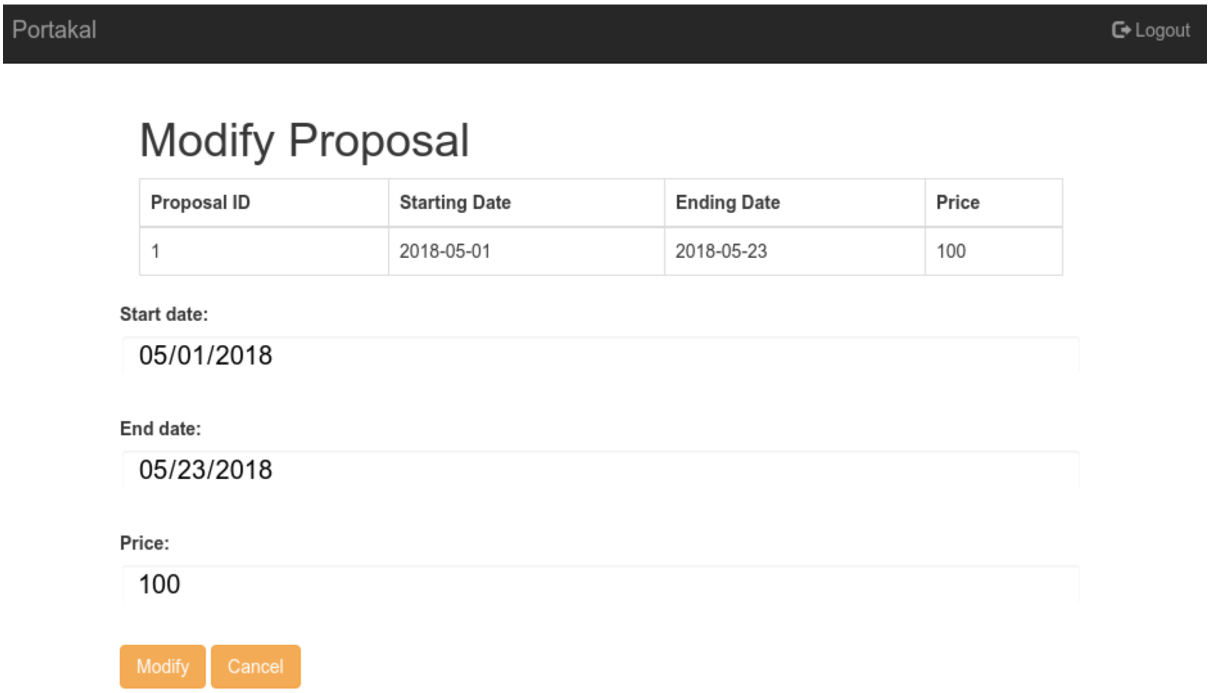


View Proposals

Proposal ID	Starting Date	Ending Date	Price	Modify	Cancel
1	2018-05-01	2018-05-23	100	Modify	Cancel

Professionals can see, modify and cancel their proposals.

Modify Proposal for Professional Users



Modify Proposal

Proposal ID	Starting Date	Ending Date	Price
1	2018-05-01	2018-05-23	100

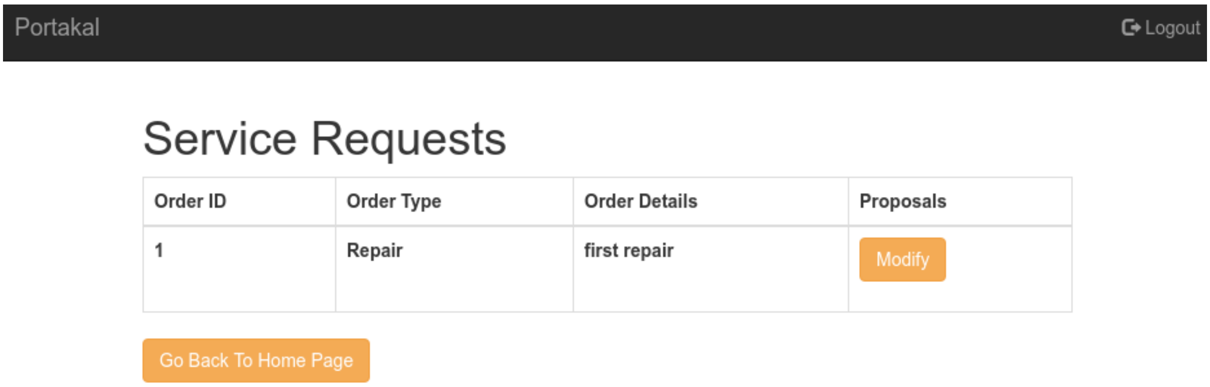
Start date:
05/01/2018

End date:
05/23/2018

Price:
100

PUs can modify their proposals from this page after pressing Modify button on the previous page.

View Service Requests for Professional Users

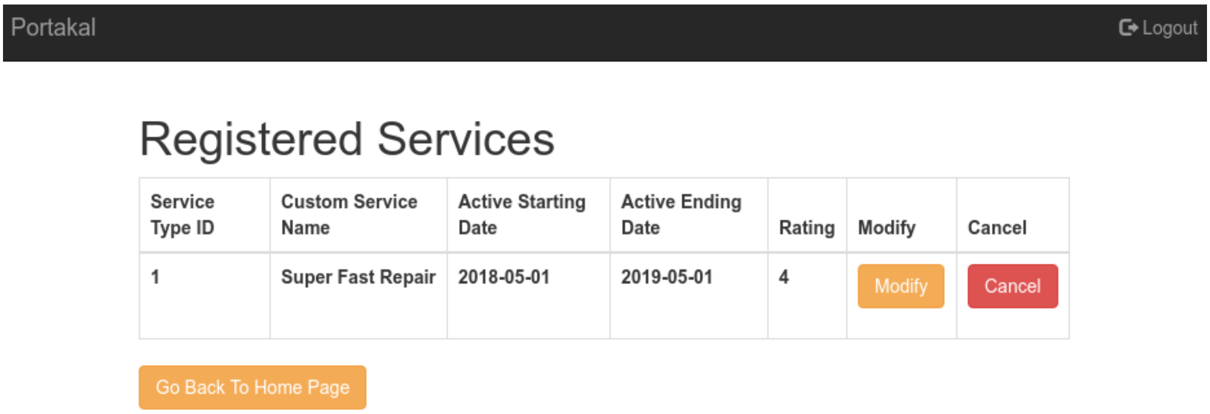


Service Requests

Order ID	Order Type	Order Details	Proposals
1	Repair	first repair	Modify

PUs can see services related to services which they provide. The button under proposals column is Modify or Propose depending on the proposal situation. If there is an existing proposal it shows Modify else it shows propose. Functionality varies according to the situation.

View Services for Professional Users

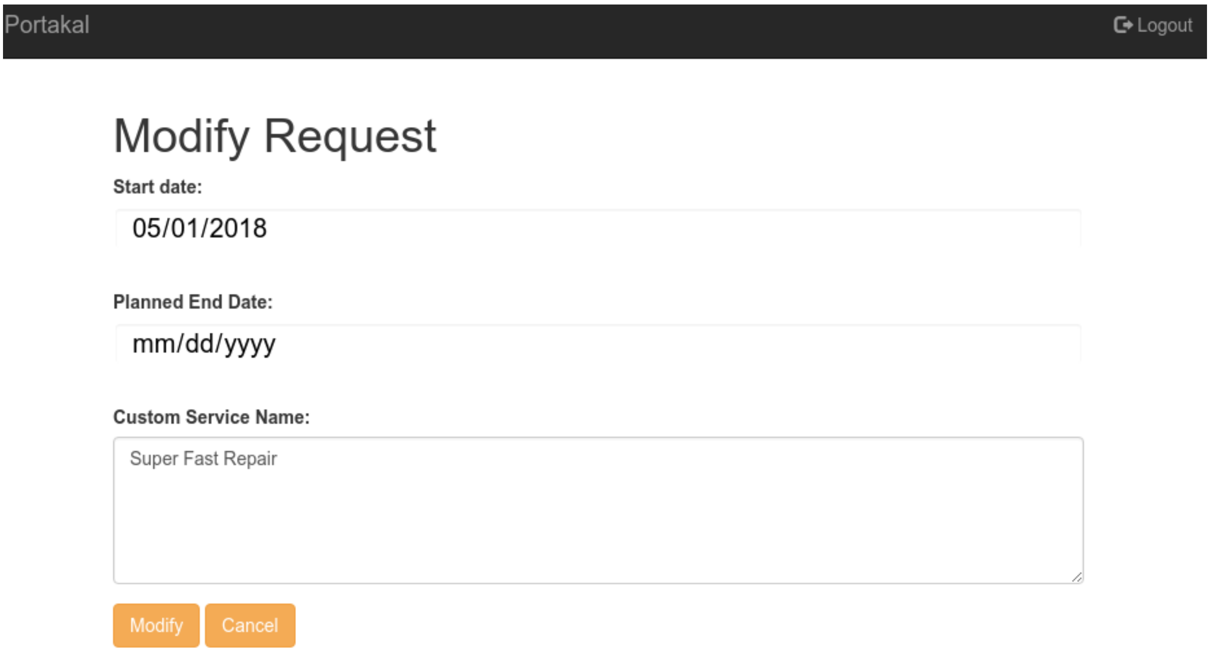


Registered Services

Service Type ID	Custom Service Name	Active Starting Date	Active Ending Date	Rating	Modify	Cancel
1	Super Fast Repair	2018-05-01	2019-05-01	4	Modify	Cancel

In this page PUs can see the services they provide. They can modify or cancel.

Modify Services for Professional Users



Modify Request

Start date:

05/01/2018

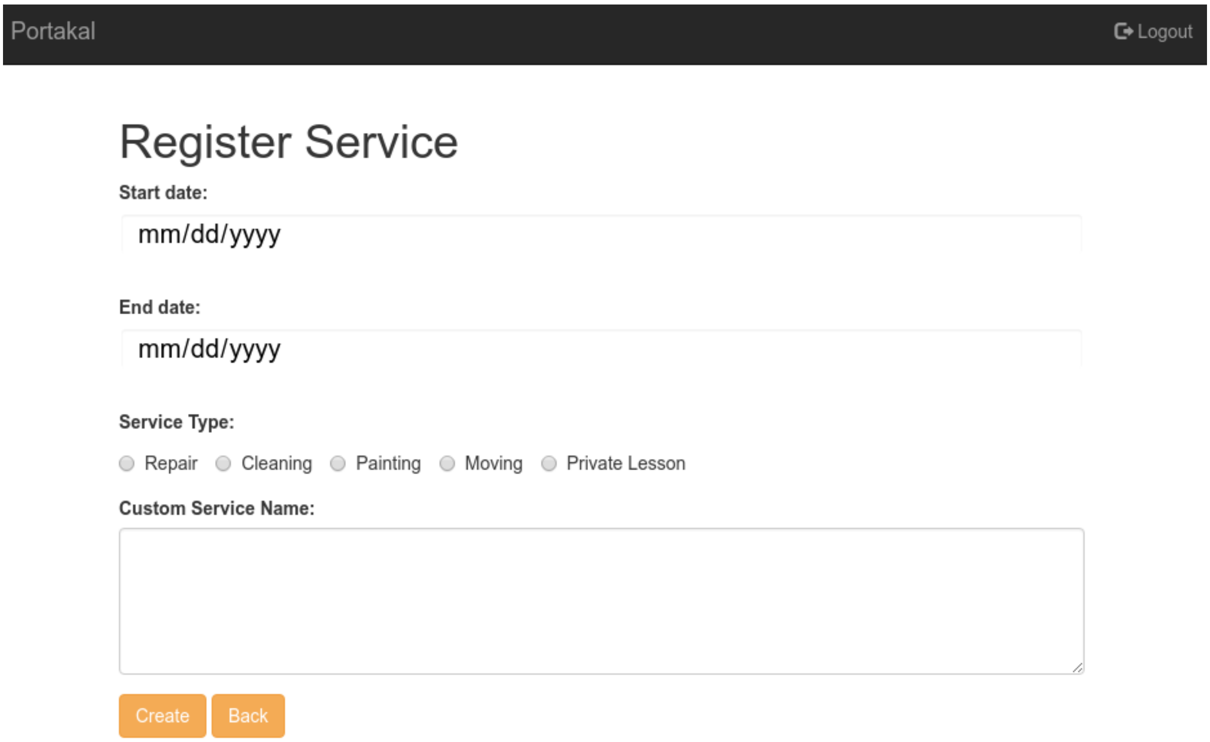
Planned End Date:

mm/dd/yyyy

Custom Service Name:

Super Fast Repair

Register Service for Professional Users



Register Service

Start date:

mm/dd/yyyy

End date:

mm/dd/yyyy

Service Type:

○ Repair ○ Cleaning ○ Painting ○ Moving ○ Private Lesson

Custom Service Name:

8. Web Site

Website information and more detailed contribution information can be found in the following github repository.

https://github.com/BecerZ/hiring_system

Demo website can be found on the following website.

http://ludeogames.com/portakal/homepage.php