



16^{-ТО} издание

FMI Parking System : Паркосистема на ФМИ



Ръководители: доц. д-р Милен Петров, ас. Александър Стефанов			
Начална година: 2021	Програма: бакалавър, (СИ)	Курс: 3	
Предмет: WEB Технологии	Издание: 16 ^{-ТО}	Код: w16prj_SI_final	
Име	Фамилия	Факултетен номер	e-mail
Андрей	Стоев	62369	akstoev@uni-sofia.bg
Светлин	Попиванов	62275	svetlinip@gmail.com
Калоян	Николов	62252	kaloyan8nikolov@gmail.com



Факултет по математика и информатика
СУ „Св. Климент Охридски“

2021-06-10, гр. София

СЪДЪРЖАНИЕ

1. Условие	3
2. Въведение	3
3. Използвани технологии	3
4. Инсталация и настройки	3
5. Ръководство на потребителя	4
5.1. Регистрация	4
5.2. Вход в системата	4
5.3. Начална страница „Моят акаунт“	4
5.4. Секция „Паркинг зони“	5
5.5. Секция „Запази паркомясто“	6
5.6. Секция „Моят график“	7
6. Примерни данни	8
6.1. Тестов план	10
7. Описание на програмния код	11
8. База данни	15
9. Описание на функционалностите	16
10. Описание на алгоритмите използвани във функционалностите	23
11. Принос на студента	26
12. Ограничения и възможности за бъдещо развитие	26
13. Какво научихме	27
14. Използвани източници	27

1. Условие

Да се създаде паркосистема на ФМИ, която включва:

- ◆ Вход в системата и управление на видовете регистрирани потребители: „Щатен преподавател“, „Хоноруван преподавател“, „Студент“;
- ◆ Начална страница за всеки потребител, включваща информация за неговите данни и отключени функционалности спрямо вида потребител;
- ◆ Прочитане на CSV файл, съдържащ графика на занятията на щатен или хоноруван преподавател и запаметяване на информацията от него в базата данни на системата;
- ◆ Глобална карта на паркозоните, където може да се паркира;
- ◆ Възможност за търсене на свободни паркоместа по зададена дата и времеви интервал;
- ◆ Възможност за резервиране на паркомясто по зададена дата и времеви интервал;
- ◆ Възможност за преглед на всички запазени паркоместа направени от съответния потребител;
- ◆ Възможност за преглед на всички предстоящи занятия на „Щатен преподавател“ и „Хоноруван преподавател“.

2. Въведение

FMI Parking System има за цел да въведе ред в използването на паркоместата на територията на ФМИ и близките до него факултети. Като собственост на СУ, за тях трябва да бъде предоставена „справедлива“ логика за използване от оторизираните за това потребители. До този момент такава система няма и всички потребители, които използват паркоместата, го правят на принципа „ако се случи да е свободно ще паркирам, ако не – ще импровизирам“. FMI Parking System ще промени тази практика, като отново запази борбата за ресурси по време, но обосobi допълнителна логика, която ще внесе ред и последователност.

3. Използвани технологии

Системата за паркиране е със стандартна трислойна архитектура състояща се от презентационен слой (Presentation Tier), логически слой (Logic Tier) и слой „Данни“ (Data Tier).

За реализацията на презентационния слой сме използвали маркиращ език HTML, стилизиращ език CSS и програмен език JavaScript. При разработката са следвани добрите практики за семантичен HTML5 синтаксис, както и ECMAScript 6 синтаксис за JavaScript.

Логическия слой е базиран на програмния език PHP и чрез него е реализирана сървърната логика.

За слоя „Данни“ сме използвали релационна база данни – MySQL.

При разработването на системата се използва мултиплатформения софтуер  XAMPP за поддръжка на базата данни и изпълнението на PHP скриптовете.

4. Инсталация и настройки

За създаването на базата данни е необходимо да се импортира SQL скрипта "backend/db/create_db/create_db.sql".

За добавяне на тестово съдържание в базата данни е необходимо да се импортира SQL скрипта "backend/db/create_db/test_data_db.sql".

5. Ръководство на потребителя

5.1. „Регистрация“

Потребител се регистрира в системата като попълни форма със следните полета:

Система за паркиране във ФМИ

Име: Въведете име

Фамилия: Въведете фамилия

Имейл: Въведете имейл

Парола: Въведете парола

Повторете паролата: Повторете паролата

Тип: Щатен преподавател

Регистриране

Фиг. 01

Система за паркиране във ФМИ

Имейл: въведете имейл

Парола: въведете парола

Вход

Регистрация

Фиг. 02

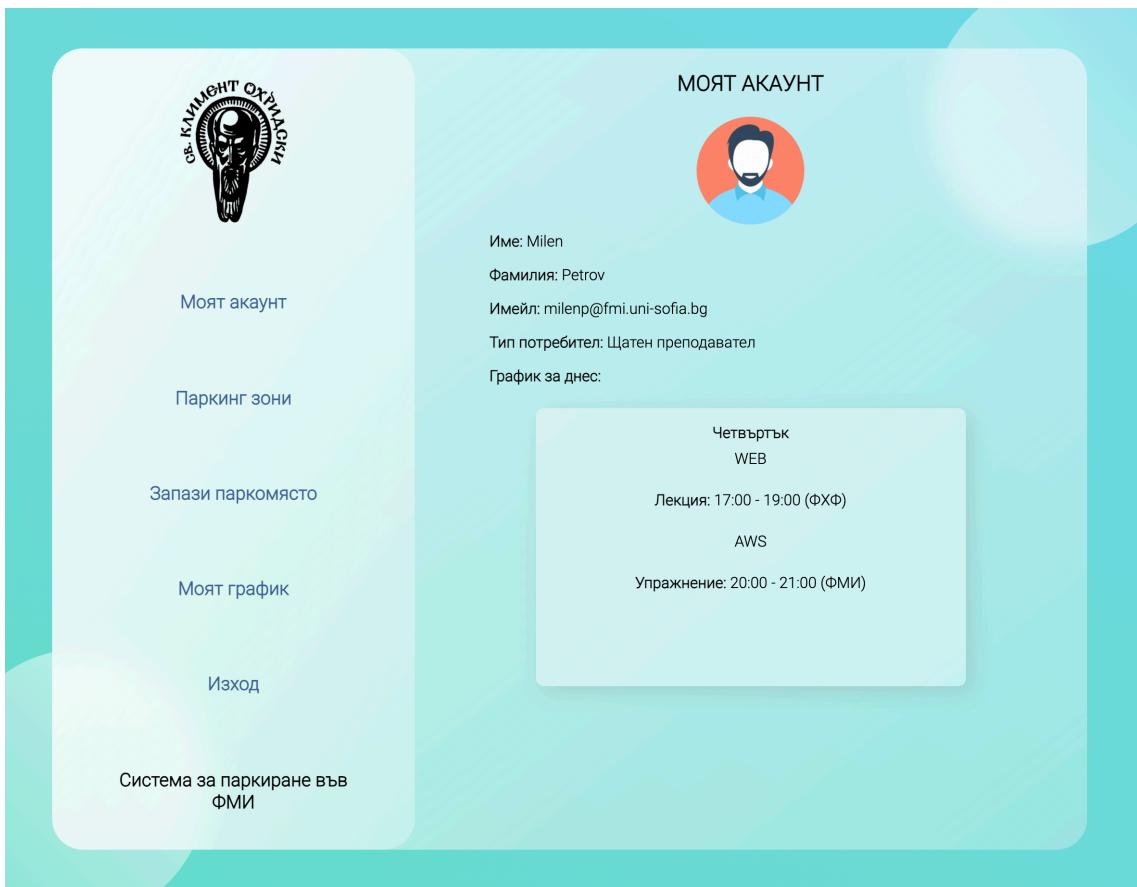
Типовете потребители, които се поддържат от системата са: „Щатен преподавател“, „Хоноруван преподавател“ и „Студент“ (Фиг. 01).

5.2. „Вход в системата“

Потребител влиза във вече съществуваща си акаунт в системата чрез въвеждане на имейл и парола (Фиг. 02).

5.3. Начална страница „Моят акаунт“.

От лявата страна на началната страница е разположено навигационно меню с връзка към следните функционалности „Моят акаунт“, „Паркинг зони“, „Запази паркомясто“, „Моят график“, „Изход“. От дясната страна на страницата е обособена секция с информация относно акаунта на потребителя.



Фиг. 03

5.4. „Паркинг зони“

Представена е общата карта с всички паркинг зони, които се администрират от системата.



Фиг. 04
Назад към НАЧАЛО

5.5. „Запази паркомясто“

Включена е галерия, която предоставя визуална информация на потребителя за разпределението на местата по зони.

Потребител избира дата и времеви интервал посредством специализирана за това форма. След изпращане на формата, до базата данни се представя информация за статуса на всяко паркомясто, в избраната от потребителя зона.

Легенда:

A3

НЕДОПУСТИМО. Паркомясто е маркирано в сив цвят, ако попада в един от трите случая:

1. Потребителят все още не е направил заявка;
2. Мястото не е достъпно за дадения вид потребител. За студент не са достъпни всички места предназначени за преподавател (в случая местата с номера < 7). За хоноруван преподавател не са достъпни всички места, предназначени за преподавател от зона А (в случая местата от зона А с номера < 7);
3. Мястото не е разрешено. Преподавателят няма лекция или упражнение в търсения от него интервал за резервация на паркомясто (въпреки добавения допълнителен аванс от по един час преди и след края на всяко занятие).

A3

СВОБОДНО. Паркомястото е маркирано в зелен цвят, ако може да бъде запазено от потребителя.

A3

РЕЗЕРВИРАНО. Паркомястото е маркирано в червен цвят, ако вече е запазено от друг потребител.

A3

ТОКУ-ЩО РЕЗЕРВИРАНО. Паркомястото е маркирано в оранжев цвят, ако потребителят тъкмо го е запазил.

Потребител може да избере само паркоместата маркирани със зелен цвят. При натискане на дадено място изскуча меню за потвърждение на избора на паркомястото.

Фиг. 05

Изберете времеви интервал

Дата: dd.mm.yyyy Начало: 7 Край: 8 ТЪРСИ

Избран времеви интервал
Дата: 2021-06-18; Интервал: 17-20

ЗОНА A ЗОНА B ЗОНА C ЗОНА D

B0	B1	B2	B3	B4
B5	B6	B7	B8	B9

Фиг. 06

Сигурни ли сте, че искате да резервирате B6?

ДА НЕ

Избран времеви интервал
Дата: 2021-06-18; Интервал: 17-20

ЗОНА A ЗОНА B ЗОНА C ЗОНА D

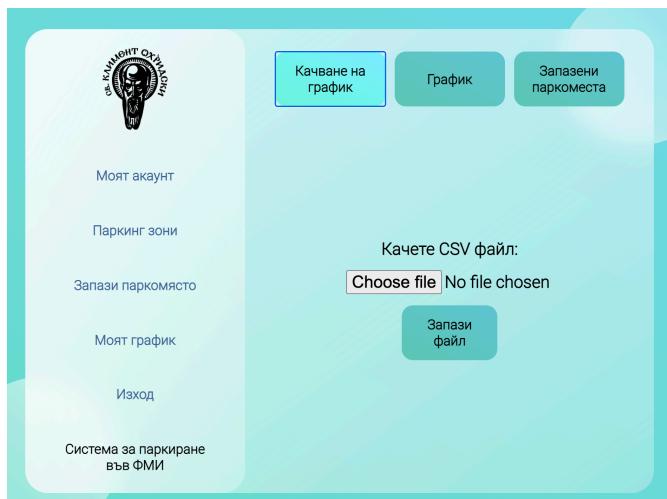
B0	B1	B2	B3	B4
B5	B6	B7	B8	B9



Фиг. 07

5.6. „Моят график“

Потребители от тип „Щатен преподавател“ или „Хоноруван преподавател“ могат да качат своя график чрез бутона „Качване на график“ и избиране на съответния CSV файл. Предаването става с бутона „Запази файл“ (Фиг. 8 и Фиг. 9).

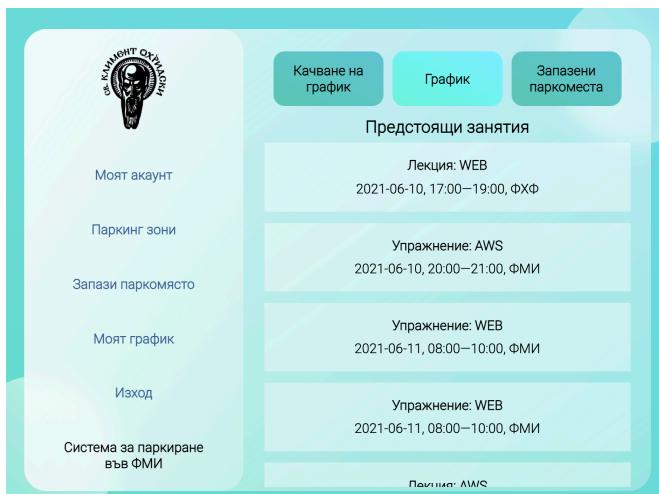


Фиг. 08

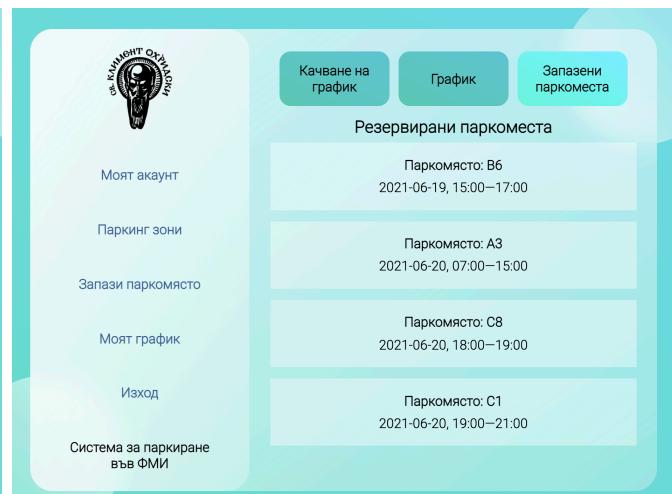


Фиг. 09

В секцията „График“, „Щатен преподавател“ и „Хоноруван преподавател“ проверяват предстоящите си занятия сортирани по време (Фиг. 10).



Фиг. 10



Фиг. 11

В секция „Запазени паркоместа“, всички потребители на системата могат да проверят предстоящите си резервации на паркоместа сортирани по време (Фиг. 11).

6. Примерни данни

Импортирането на SQL скрипта с тестови данни описан в точка 4 ще създаде следните три акауна на потребители, техните графици и направени резервации за паркоместа:

	„Щатен преподавател“	„Хоноруван преподавател“	„Студент“
Име:	Milen	Alexander	Pesho
Фамилия:	Petrov	Stefanov	Peshov
Имейл:	milenp@fmi.uni-sofia.bg	astef@fmi.uni-sofia.bg	pesho@gmail.com
Парола	milenp	astef	pesho

petrov_shedule.csv

Дисциплина	Тип занятие	Дата	Начален час	Краен час	Сграда
AWS	Лекция	2021-06-23	10:00	14:00	ФМИ
WEB	Лекция	2021-06-24	09:00	11:00	ФМИ
AWS	Упражнение	2021-06-24	16:00	19:00	ФзФ
Network programming	Упражнение	2021-06-25	11:00	13:00	ФХФ
JST	Лекция	2021-06-25	15:00	18:00	ФМИ
OOP	Упражнение	2021-06-26	08:00	10:00	ФМИ
OOP	Упражнение	2021-06-26	12:00	14:00	ФМИ
OOP	Лекция	2021-06-26	18:00	21:00	ФХФ

stefanov_schedule.csv

Дисциплина	Тип занятие	Дата	Начален час	Краен час	Сграда
WEB	Упражнение	2021-06-24	10:00	12:00	ФМИ
Statistics	Упражнение	2021-06-24	18:00	20:00	ФМИ
OOP	Упражнение	2021-06-25	16:00	18:00	ФзФ

таблица с направени резервации на паркоместа

Потребител	Номер	Дата	Начален час	Краен час
Milen Petrov	B6	2021-06-25	15:00	17:00
Milen Petrov	A3	2021-06-26	07:00	15:00
Milen Petrov	C8	2021-06-26	18:00	19:00
Milen Petrov	C1	2021-06-26	19:00	21:00
Alexander Stefanov	B2	2021-06-24	09:00	12:00
Alexander Stefanov	B8	2021-06-24	14:00	15:00
Alexander Stefanov	B4	2021-06-24	18:00	21:00
Pesho Peshov	B7	2021-06-24	16:00	17:00
Pesho Peshov	B8	2021-06-24	17:00	18:00
Pesho Peshov	B9	2021-06-24	18:00	19:00
Pesho Peshov	D0	2021-06-25	09:00	15:00
Pesho Peshov	D7	2021-06-25	16:00	20:00

Примерен CSV файл, съдържащ допълнителни занятия за преподавател, който да се въведе в базата данни на системата чрез функционалността „Качване на файл“.

test_schedule.csv

Дисциплина	Тип занятие	Дата	Начален час	Краен час	Сграда
OS	Лекция	2021-06-27	09:00	11:00	ФХФ
Swift	Упражнение	2021-06-28	16:00	19:00	ФМИ

6.1. Тестов план

1. „Щатният преподавател“ доц. д-р Петров влиза в системата като въвежда:

Имейл: milenp@fmi.uni-sofia.bg
Парола: milenp

2. Избира секция „Моят график“ и вижда в подсекция „График“, че на 2021-06-24 има две занятия:

Лекция: WEB, 2021-06-24, 09:00 – 11:00, ФМИ
Упражнение: AWS, 2021-06-24, 16:00 – 19:00, ФЗФ

3. Избира подсекция „Запазени паркоместа“ и вижда, че няма резервирани паркоместа за 2021-06-24.
4. Избира секция „Паркинг зони“, за да провери коя зона за паркиране е най-близо до съответните сгради. Вижда, че зона А е най-близо до ФМИ, а зона В е най-близо до ФЗФ.
5. Избира секцията „Запази паркомясто“ и вижда разпределението на паркоместата в зона А.
6. Попълва, формата за търсене на свободни паркоместа, избирайки

Дата: 24.06.2021
Начало: 8
Край: 12

и изпраща заявката с бутона „ТЪРСИ“.

7. Вижда, че няма заети места от зона А и избира да резервира паркомясто А1 като натиска бутона „А1“.
8. Потвърждава, че иска да резервира паркомясто А1 чрез натискане на бутона „ДА“ от появилия се прозорец за потвърждение.
9. Натиска бутона „зона В“ и вижда, че паркомясто В2 е заето по време на първото му занятие, но той иска да намери свободни паркоместа за второто си занятие.
10. Попълва, формата за търсене на свободни паркоместа, избирайки

Дата: 24.06.2021
Начало: 16
Край: 19

и изпраща заявката с бутона „ТЪРСИ“.

11. Вижда, че паркомясто В2 вече е свободно за съответния времеви интервал, но паркоместата В4, В7, В8 и В9 са заети. Избира да резервира паркомясто В1 като натиска бутона „В1“.

12. Потвърждава, че иска да резервира паркомясто B1 чрез натискане на бутона „ДА“ от появилия се прозорец за потвърждение.
13. Доц. Петров вече е резервиран паркоместа за двете си занятия и натиска бутона „НАЗАД“.
14. Избира секция „Моят график“ и избира подсекция „Запазени паркоместа“.
15. Вижда, че в резервираните му паркоместа вече са добавени новите две резервации за занятията от 2021-06-24.

7. Описание на програмния код

FMI-Parking-System:     

0. index.html

Стартира заглавната страница на приложението.

1. frontend

1.1. registration

1.1.1. registration_form.html

HTML код за страницата на регистрацията.

1.1.2. css

1.1.2.1. form_style.css

Стилизация на страницата на регистрацията.

1.1.3. javascript

1.1.3.1. send_form.js

Изпраща въведените данни от формата за регистрация към endpoint 2.1.8.1. на сървъра.

1.1.4. images

1.1.4.1. error_response.png

1.2. login

1.2.1. login_form.html

HTML код за вход в системата.

1.2.2. css

1.2.2.1. login_form_styles.css

Стилизация на страницата за вход в системата.

1.2.3. javascript

1.2.3.1. send_user_data.js

Изпраща въведените данни от формата за вход в системата към endpoint 2.1.6.1. на сървъра.

1.2.4. images

1.2.4.1. error_response.png

1.3. account

1.3.1. account_view.html

HTML код за главната страница на потребителя.

1.3.2. css

1.3.2.1. account_view_styles.css

Стилизация на главната страница на потребителя.

1.3.3. javascript

1.3.3.1. button_reaction.js

Дефинира реакциите на бутоните от навигационното меню от главната страница на потребителя.

1.3.3.2. get_schedule.js

Дефинира реакциите на бутоните от секция „Моят график“, проверявайки типа на потребителя.

При натискане на бутон „График“ извлича графика за съответния преподавател с предстоящите му занятия (от endpoint 2.1.5.1) и ги визуализира в сортиран по време ред.

При натискане на бутон „Запазени паркоместа“ извлича предстоящите резервации на паркоместа за съответния потребител (от endpoint 2.1.5.2.) и ги визуализира в сортиран по време ред.

При натискане на бутон „Качване на график“ се открива подсекцията за качване на CSV файл.

1.3.3.3. load_today_schedule.js

Зарежда графика със занятия за съответния ден на преподавателя (чрез endpoint 2.1.4.1.), сортирани по два критерия: първо по дисциплина, след което по време.

1.3.3.4. load_user_data.js

Зарежда данните от регистрацията на потребителя (чрез endpoint 2.1.4.2.) и ги визуализира в заглавната му страница „Моят акаунт“.

1.3.3.5. log_out_user.js

Изпраща заявка към endpoint 2.1.7.1. на сървъра, който да изтрие сесията и бисквитките на съответния потребител.

1.3.3.6. map_click_reaction.js

Добавя динамично класове към елемента със снимка на картата, които чрез CSS файла 1.3.2.1. отговарят за увеличаването и намаляването на изображението.

1.3.3.7. submit_csv_file.js

Изпраща се въведения от преподавателя CSV файл към endpoint 2.1.10.1 на сървъра.

1.3.4. images

1.3.4.1. map.png

1.3.4.2. sample_user1.png

1.3.4.3. Sofia_University_Logo.png

1.4. map

1.4.1. zones.html

HTML код за секция „Запази паркомясто“.

1.4.2. css

1.4.2.1. zones.css

Стилизация на страницата за секция „Запази паркомясто“.

1.4.3. javascript

1.4.3.1. buttons_click.js

Създават се слушател за събитие 'click' върху всеки от бутоните на паркоместата. При настъпване на такова събитие се проверява дали съответния бутон е на свободно паркомясто и се изпраща заявка към endpoint 2.1.9.1. на сървъра. В зависимост от отговора на сървъра – ще се промени цветът на бутона, за да отговаря на новия му статус (току-що резервиран) или ще визуализира самозатварящ се прозорец, индикиращ за грешка.

1.4.3.2. handle_select_option.js

Задава логика за рестрикция на опциите за „Край“ в зависимост от избора на „Начало“ на времевия интервал от формата за търсене на свободно паркомясто.

1.4.3.3. load_map.js

Създава динамично бутона за всички паркоместа. Създава динамично опциите за „Начало“ и „Край“ от формата за търсене. Дефинира поведението на слайдшоуто със стрелките в ляво и дясно на снимките, както и на бутоните под него.

1.4.3.4. send_form_search.js

Изпраща въведените от потребителя данни във формата за търсене на свободно паркомясто до endpoint 2.1.1.1. на сървъра. В зависимост от отговора на сървъра или всеки бутон ще се оцвети в цвета, отговарящ на статуса му, или самозатварящ се прозорец ще индицира за грешка.

1.4.4. images

- 1.4.4.1. A.png
- 1.4.4.2. azone.png
- 1.4.4.3. B.png
- 1.4.4.4. back.png
- 1.4.4.5. bzone.png
- 1.4.4.6. C.png
- 1.4.4.7. czone.png
- 1.4.4.8. D.png
- 1.4.4.9. dzone.png
- 1.4.4.10. map.png
- 1.4.4.11. search.png
- 1.4.4.12. zA.png
- 1.4.4.12. zB.png
- 1.4.4.13. zC.png
- 1.4.4.13. zD.png

2. backend

2.1. api

2.1.1. check_slots

2.1.1.1. check_slots.php

Проверява данните за въведения времеви интервал от потребителя и го валидира. Ако е некоректен връща грешка, а ако е коректен прилага алгоритмите A1 и A2 и връща два масива от асоциативни масиви, които съдържат информация за резервираните и недостъпните за потребителя бутони.

2.1.2. check_user_type

2.1.2.2. check_user_type.php

Проверява и връща типа на потребителя.

2.1.3. functions

2.1.3.1. check_JSON_validity.php

Валидира подаден обект с JSON формат.

2.1.3.2. login.php

При успешно автентифициране на потребител създава негова сесия и бисквитки.

2.1.4. load_account_page

2.1.4.1. get_today_schedule.php

Връща графика със занятия на съответния преподавател за деня, извлечени от базата данни.

2.1.4.2. get_user_data.php

Връща име, фамилия, имейл и типа на потребителя, извлечени от базата данни.

2.1.5. load_user_schedule_data

2.1.5.1. load_user_schedule.php

Връща графика с предстоящите занятия на съответния преподавател, извлечени от базата данни.

2.1.5.2. load_user_taken_slots.php

Връща графика с предстоящите резервации на паркоместа за съответния потребител, извлечени от базата данни.

2.1.6. login

2.1.6.1. login_user.php

Получава и проверява формата на данните от 1.2.3.1. чрез endpoint 2.1.3.1. и ги препраща на endpoint 2.1.3.2. на сървъра. Върнатият отговор се предава към 1.2.3.1.

2.1.7. logout_user

2.1.7.1. delete_user_session.php

Изтрива сесията и бисквитките на съответния потребител.

2.1.8. registration

2.1.8.1. register_user.php

Проверява дали въведените данни от 1.1.3.1. за регистрация са валидни и ако да – ги запаметява в базата данни и създава сесия и бисквитки на съответния потребител.

2.1.9. reserve_slots

2.1.9.1. reserve_slot.php

Изпраща заявка до базата данни, за да провери дали потребителят вече има съществуваща резервация за (част от) избрания времеви интервал.

Проверява дали избраното от потребителя място е "lecturer_only" и на база на това определя дали паркомястото е достъпно за съответния потребител.

Ако потребителят е преподавател и мястото е "lecturer_only" се проверява дали преподавателят има занятие в рамките на часовия интервал. Ако сме достигнали до тук след всички проверки и не сме върнали съобщение за грешка, то единственото нещо което остава да проверим е дали заявленото паркомясто е свободно или заето. Това става чрез заявка до базата данни. Ако е свободно (няма сечение с други резервации) се изпраща нова заявка до базата данни, с която паркомястото се резервира.

2.1.10. upload_csv

2.1.10.1. send_csv_to_db.php

Проверява дали в асоциативния масив от файлове, които се получават чрез POST заявка, има качен файл в CSV формат. След проверката, съдържанието на файла се разделя по нов ред ('\n') и се записва като масив от редове. Проверява за наличието на заглавен ред (header) и в случай, че има го пропуска. Всеки ред се разделя по запетая (','), преформатират се данните като базата данни очаква и всеки ред се записва в масив от асоциативни масиви. Всеки ред от новия масив се подава на заявка, която го записва в базата данни.

2.2. db

2.2.1. create_db

2.2.1.1. create_db.sql

Създава празни таблици от базата данни и релациите между тях. Този файл служи за конфигуриране на базата данни на системата. След неговото импортиране, системата ще бъде напълно функционална.

2.2.1.2. test_data_db.sql

Добавя примерни записи в таблиците на базата данни, които служат за тестването на системата.

2.2.2. db_connection

2.2.2.1. connect_to_db.php

Създава клас, който има метод за свързване с базата данни.

2.2.2.2. db_info.ini

Конфигурационен файл за осъществяване на връзка с базата данни.

8. База данни

8.1. Структурата на базата от данни се състои от следните таблици:

1. **users**: съдържа информация за потребителите на системата. Когато потребител се регистрира в системата се добавя запис. Когато потребител влиза в системата се автентицира с помощта на съдържанието в тази таблица.
2. **schedules**: съдържа информация за занятията от графиците на всички преподаватели. Когато преподавател качва в системата CSV файл, съдържанието му се добавя под формата на записи в таблицата. Служи за валидация на времеви интервали при правенето на резервации на паркоместа.
3. **user_schedules**: съдържа идентификационния номер на преподавателя за всеки запис от schedules. Служи като свързваща таблица между users и schedules.
4. **slots**: съдържа информацията за всяко паркомясто и се попълва от конфигурационния файл "create_db.sql". Използва се за валидация при резервация на паркомясто.
5. **reservations**: съдържа информация за всички резервации на паркоместа. При създаване на нова резервация се добавя нов запис в таблицата. Използва се за валидация при резервация на паркомясто.

8.2. Връзки в базата от данни:

- ◆ users – user_schedules: връзка от тип „1 : много“
- ◆ user_schedules – schedules: връзка от тип „1 : 1“
- ◆ users – reservations: връзка от тип „1 : много“
- ◆ reservations – slots: връзка от тип „много : 1“

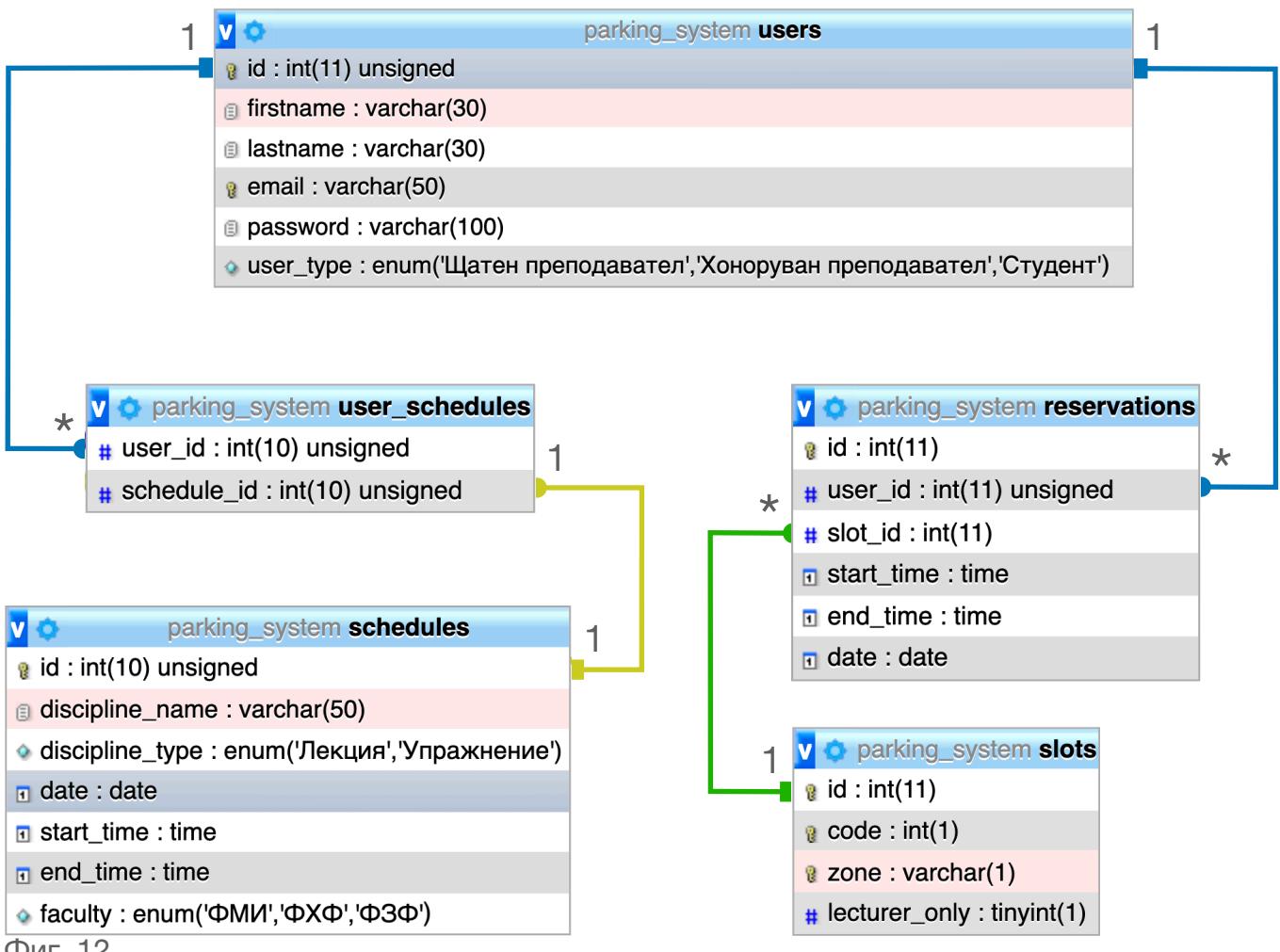
8.3. Ключове

Ключове:

- 0→ users: id;
- 0→ schedules: id;
- 0→ reservations: id;
- 0→ slots: id;

Чужди ключове:

- 0→ user_schedules:
user_id, schedule_id
- 0→ reservations:
user_id, slot_id



Фиг. 12

9. Описание на функционалностите

Ф1. Преглед на резервириани паркоместа

Функционално изискване: Регистриран потребител прави справка за предстоящите си резервации на паркоместа (Фиг. 11).

Резултат: Системата визуализира списък на предстоящите резервации на паркоместа на потребителя.

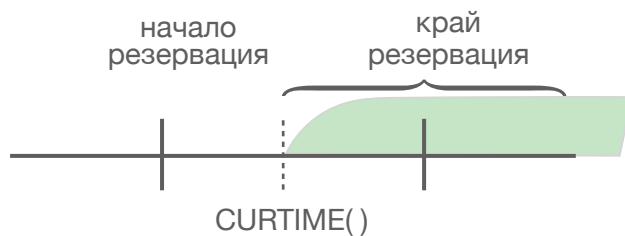
Начин на реализация:

1. Потребител избира подсекция „Запазени паркоместа“ от „Моят график“;
2. JavaScript функция в "get_schedule.js" слуша за това събитие и след като го улови, изпраща GET заявка до endpoint "load_user_taken_slots.php" на сървъра;
3. Прави се заявка до базата данни, която обединява таблиците за резервации и паркоместа и извлича всички предстоящи резервации (с кода и зоната) на паркоместа на съответния потребител, сортирани по време;
4. Форматира часовете, получени от базата данни (hh:mm:ss → hh:mm);
5. След всяко форматиране се записва реда в масив "taken_slots";
6. Масивът "taken_slots" се връща в JSON формат към "get_schedule.js";

- За всяка резервация от масива се създават динамично HTML елементи, които я описват и след това се стилизират чрез CSS;

(Ф1.1.) SQL заявка за извлечане на всички предстоящи резервации на паркоместа на потребител:

```
$sql = "SELECT r.start_time, r.end_time, r.date, s.code, s.zone
        FROM reservation r JOIN slots s ON s.id = r.slot_id
        WHERE r.user_id LIKE :user_id AND (r.date > CURDATE()
        OR (r.date LIKE CURDATE() AND r.end_time >= CURDATE()))
        ORDER BY r.date, r.start_time, r.end_time";
```



Ф2. Качване на график във CSV формат

Функционално изискане: Регистриран потребител от тип „Хоноруван преподавател“ или „Щатен преподавател“ въвежда график на занятията си в CSV формат (Фиг. 8 и Фиг. 9).

Резултат: Системата записва информацията от CSV файла в базата данни.

Начин на реализация:

- Преподавател избира подсекция „Качване на график“ от „Моят график“;
- Преподавател избира CSV файл, който да бъде качен чрез бутона „Choose file“ (Фиг. 08);
- Преподавател финализира изпращането на файла към системата посредством бутона „Запази файл“;
- JavaScript функция от файла "submit_csv_file.js" улавя за това събитие и създава асоциативен масив, в който добавя формата на файла като ключ и файла като стойност;
- Изпраща POST заявка към endpoint "send_csv_to_db.php" на сървъра;
- Проверява дали в асоциативния масив от файлове, които се получават чрез POST заявка, има качен файл в CSV формат;
- Съдържанието на файла се разделя по нов ред ('\n') и се записва като масив от редове;
- Проверява за наличието на заглавен ред (header) и в случай, че има го пропуска;
- Всеки ред се разделя по запетая (,), преформатират се данните както базата данни очаква и всеки ред се записва в масив от асоциативни масиви;
- Всеки ред от новия масив се подава на заявка, която го записва в базата данни;
- Връща се статуса на заявката за добавяне на съдържанието на файла в базата данни;

12. JavaScript функция от файла "submit_csv_file.js" визуализира самозатварящ се прозорец със съобщение за статуса на изпълнената операция.

Структура на CSV файла:

```
$> cat test_schedule.csv
Дисциплина, Тип занятие, Дата, Начален час, Краен час, Сграда
OS, Лекция, 2021-06-27, 09:00, 11:00, ФХИ
Swift, Упражнение, 2021-06-28, 16:00, 19:00, ФМФ
```

CSV файлът може да съдържа като първи ред опционален хедър с информацията за съдържанието във всяка от колоните. Колоните използват за разделител ",". Информацията от CSV файла се запазва в базата данни.

(Ф2.1.) SQL заявка за въвеждане на данни от график, чрез CSV файл:

```
$sql = "INSERT INTO schedules (discipline_name, discipline_type, date, start_time,
                                end_time, faculty)
VALUES (:discipline_name, :discipline_type, :date, :start_time,
        :end_time, :faculty);"
```

(Ф2.2.) SQL заявка за въвеждане релации, на вече въведените данни по-горе:

```
$sql = "INSERT INTO user_schedules (user_id, schedule_id)
VALUES (:user_id, :schedule_id);"
```

Ф3. График на предстоящите занятия

Функционално изискване: Регистриран потребител от тип „Хоноруван преподавател“ или „Щатен преподавател“ прави справка за графика си със занятия (Фиг. 10).

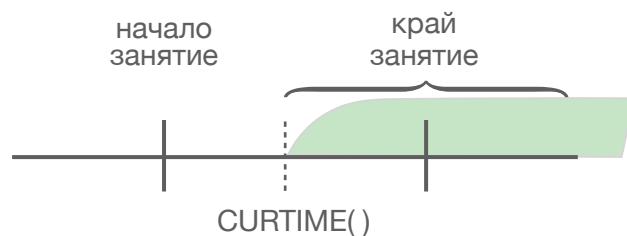
Резултат: Системата визуализира списък на предстоящите занятия на преподавателя.

Начин на реализация:

1. Потребител избира подсекция „График“ от „Моят график“;
2. JavaScript функция в "get_schedule.js" слуша за това събитие и след като го улови, изпраща GET заявка до endpoint "load_user_schedule.php" на сървъра;
3. Прави се заявка до базата данни, която обединява таблиците за графици и свързващата таблица между графици и преподаватели и извлича всички предстоящи занятия на съответния преподавател, сортирани по време;
4. Форматира часовете, получени от базата данни (hh:mm:ss → hh:mm);
5. След всяко форматиране се записва реда в масив "schedules";
6. Масивът "schedules" се връща в JSON формат към "get_schedule.js";
7. За всяко занятие от масива се създават динамично HTML елементи, които го описват и след това се стилизират чрез CSS;

(Ф3.1.) SQL заявка за извлечане на график на съответния преподавател:

```
$sql = "SELECT s.discipline_name, s.discipline_type, s.date, s.start_time,
           s.end_time, s.faculty
      FROM schedules s JOIN user_schedules us ON us.schedule_id LIKE s.id
     WHERE us.user_id LIKE :user_id AND
           (s.date > CURDATE() OR
            (s.date LIKE CURDATE() AND s.end_time >= CURTIME() ))
    ORDER BY s.date, s.start_time, s.end_time";
```



Ф4. Справка за свободни паркоместа в определен интервал от време

Всеки регистриран потребител има достъп от бутона „Запази паркомясто“ до слайдшоу с различните зони за паркиране (от А до D). Всяка една зона разполага с 10 места за паркиране, номерирани по следния шаблон <[име на зона: А-D]><[номер на парко място: 0-9]>. Например, паркомясто B9 е последното паркомясто от зона B, а паркомясто A0 е първото място от зона A. По подразбиране всяко едно от паркоместата е маркирано в сив цвят, който индицира, че потребителя не е осъществил никаква заявка до този момент и все още само разглежда.

На страницата за запазване на паркомясто има форма за попълване, която се състои от три полета за въвеждане на данни от потребителя. Данните са съответно: дата, начало и край.

Изберете времеви интервал

Дата: Начало: Край:

Фиг. 13

С посочените по-горе полета, потребителят избира конкретната дата и задава интервал от време, в който проверява за свободни паркоместа. При направена справка, той получава информация за това, кои точно са свободните и заетите места, за всяка от зоните.

Функционално изискване: Регистриран в системата потребител прави справка за свободни паркоместа за дадена дата и интервал от време (Фиг. 05 и Фиг. 14).

Резултат: Система маркира всички паркоместа в зависимост от техния статус (виж легенда в точка 6.5.).

Начин на реализация:

1. Потребителят въвежда дата, час за начало и час за край на времевия интервал. Пояснение: при избиране на начален час, чрез JavaScript функция се определят допустимите часове съответно за край или начало на времевия интервал. По този начин се ограничава възможността от потребителска грешка, при която се въвежда невалиден интервал от време;
2. След натискане на бутона „ТЪРСИ“, JavaScript функция съхранява въведените от потребителя данни в обект D и връща формата в първоначалното ѝ състояние;
3. Системата изпраща данните от обект D в end-point "check_slots.php" на сървъра;
4. Проверява се дали в D има валидна дата (дали потребителя е въвел дата във формата);
5. Извличат се интервалите на всички занятия, които базата данни съдържа за потребителя за деня от D ;
6. Ако потребителят е щатен или хоноруван преподавател, създаваме и попълваме масива $hours[]$ (Фиг. 15);
7. Ако потребителят е щатен или хоноруван преподавател, проверяваме дали лекторът има право да запази място в посочения от него времеви интервал чрез масива $hours[]$ (Фиг. 15, алгоритъм A2);
8. Извличаме и записваме заетите паркоместа в избрания от потребителя времеви интервал в масива $slots_taken[]$. (Фиг. 14, алгоритъм A1);
9. В зависимост от типа на потребителя се извличат и съхраняват недостъпните за него места в масив $unavailable_slots[]$;
10. json форматирано съобщение със заетите и недостъпните за потребителя места (масивите $slots_taken[]$ и $unavailable_slots[]$);
11. Чрез JavaScript функция маркираме съответните паркоместа в цветовете съответстващи на тяхния статус (сив: недостъпено; зелен: свободно; червен: резервирано);
12. Създаваме секция показваща избраните от потребителя дата и времеви интервал, за който е визуализирана информацията за паркоместата.

(Ф4.1.) SQL заявка за извлечане на всички занятия за съответната дата:

```
$sql = "SELECT date, start_time, end_time  
        FROM schedules s JOIN user_schedules us ON us.schedule_id LIKE s.id  
        WHERE us.user_id LIKE :user_id AND s.date LIKE :date";
```

(Ф4.2.) SQL заявка за извлечане на всички резервирани паркоместа, които имат непразно сечение с интервала от заявката (алгоритъм A1):

```
$sql = "SELECT s.code, s.zone, s.lecturer_only  
        FROM reservations r JOIN slots s ON r.slot_id LIKE s.id  
        WHERE r.date LIKE :date AND  
              (  
                r.start_time < :start_time AND r.end_time > :start_time) OR // резервация В
```

```

(r.start_time >= :start_time AND r.end_time <= :end_time) OR // резервация C
(r.start_time < :end_time AND r.end_time > :end_time) OR // резервация D
(r.start_time < :start_time AND r.end_time > :end_time) // резервация F (Фиг. 14)
);";

```

(Ф4.3.) SQL заявка за извличане на всички паркоместа, които не са за студенти:

```

$sql = "SELECT code, zone
        FROM slots
        WHERE zone LIKE 'A' AND lecturer_only LIKE TRUE;";

```

Ф5. Резервиране на паркомясто в определен интервал от време

Функционално изискване: Регистриран в системата потребител резервира паркомясто в избрания времеви интервал от справката за търсене (Фиг. 5, Фиг. 6, Фиг. 7).

Предварителни условия: Потребителят е изпълнил функционалност Ф4.

Резултат: Система запазва в базата данни резервацията, направена от потребителя.

Начин на реализация:

1. JavaScript функция долавя събитието 'click' върху свободно паркомясто за резервация;
2. JavaScript функция от "buttons_click.js" добавя клас към HTML елемент, чрез който да се стилизира и визуализира меню за потвръждение;
3. JavaScript функция "waitForUserInput()" от "buttons_click.js" добавя отговор на потребителя;
4. JavaScript функция "sendReservationData (button)" комплектова данните за избрания бутон (код и номер на паркомясто) в асоциативна структура "reservationInfo", след което извиква функцията "makeReservation (reservationInfo)" ;
5. Изпраща POST заявка към endpoint "reserve_slot.php" на сървъра;
6. Изпраща заявка до базата данни, за да провери дали потребителят вече има съществуваща резервация за (част от) избрания времеви интервал (Ф5.1.);
7. Проверява дали избраното от потребител място е "lecturer_only" и на база на това определя дали паркомястото е достъпно за съответния потребител (Ф5.2.);
8. Ако потребителят е преподавател и мястото е "lecturer_only" се проверява дали преподавателят има занятие в рамките на часовия интервал (Ф5.3.);
9. Проверява се дали заявленото паркомясто е свободно или заето, чрез заявка до базата данни (Ф5.4.);
10. Ако е свободно (няма сечение с други резервации) се изпраща нова заявка до базата данни, с която паркомястото се резервира (Ф5.5);
11. Връща JSON код с резултата от процеса по запазване на паркомясто;
12. Визуализира се самозатварящ се прозорец със съобщение за резултата от операцията;

13. При успех се добавят класове на бутона на паркомястото, чрез които да бъде стилизиран в оранжев цвят.

(Ф5.1.) Аналогична на SQL заявката от алгоритъма А1, с разликата, че в базата данни се търсят само редове, който са за съответния потребител:

```
$sql = "SELECT id  
        FROM reservations  
        WHERE date LIKE :date AND user_id LIKE :user_id AND  
        (  
            (start_time < :start_time AND end_time > :start_time) OR  
            (start_time >= :start_time AND end_time <= :end_time) OR  
            (start_time < :end_time AND end_time > :end_time) OR  
            (start_time < :start_time AND end_time > :end_time)  
        );";
```

(Ф5.2.) SQL заявка за извлечане на информация за избраното от потребителя място, за да провери дали е lecturer_only и да запази неговото id в променлива slot_id, за да я използва в заявка (Ф5.5.).

```
$sql = "SELECT code, lecturer_only  
        FROM slots  
        WHERE code LIKE :code AND zone LIKE :zone";
```

(Ф5.3.) SQL заявка за извлечане на информация за графика на съответния преподавател за съответната дата:

```
$sql = "SELECT date, start_time, end_time  
        FROM schedules s JOIN user_schedules us ON us.schedule_id LIKE s.id  
        WHERE us.user_id LIKE :user_id AND s.date LIKE :date";
```

(Ф5.4.) Аналогична на заявката от алгоритъма А1, с разликата, че в базата данни се търси резервация за даденото място, чиито времеви интервал има сечение с желания от потребителя интервал за резервация:

```
$sql = "SELECT r.slot_id  
        FROM reservations r JOIN slots s ON r.slots_id LIKE s.id  
        WHERE r.date LIKE :date AND s.code LIKE :code AND s.zone = :zone AND  
        (  
            (r.start_time < :start_time AND r.end_time > :start_time) OR  
            (r.start_time >= :start_time AND r.end_time <= :end_time) OR  
            (r.start_time < :end_time AND r.end_time > :end_time) OR  
            (r.start_time < :start_time AND r.end_time > :end_time)  
        );";
```

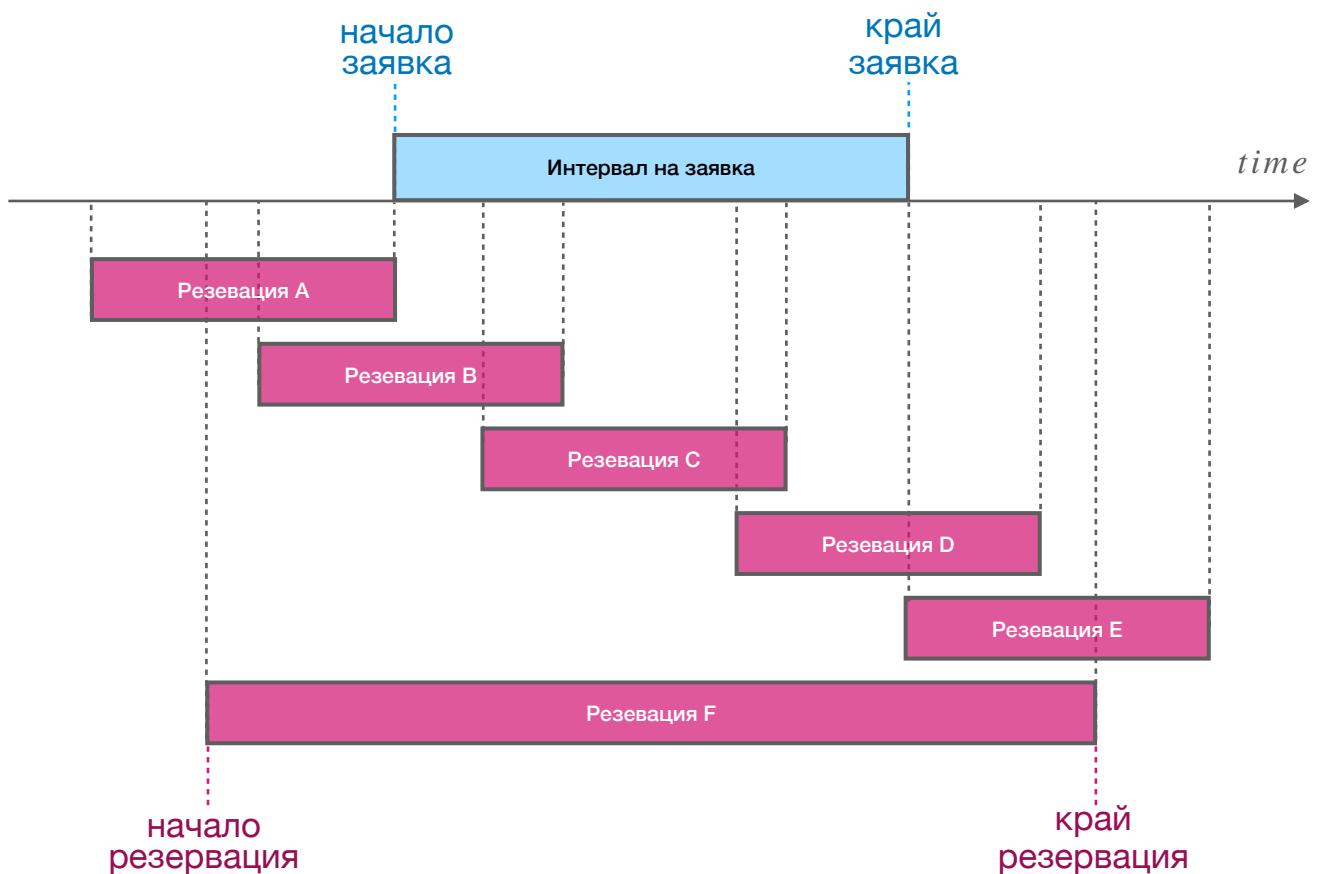
(Ф5.5.) Добавя новата резервация на паркомясто в базата данни:

```
$sql = "INSERT INTO reservations (user_id, slot_id, start_time, end_time, date)
VALUES (:user_id, :slot_id, :start_time, :end_time, :date)";
```

10. Описание на алгоритмите използвани във функционалностите

A1. Алгоритъм за определяне на свободни и заети места:

На фигурата по-долу (Фиг. 14) е изображен времевия интервал от заявката на потребителя и всички възможни сценарии за вече направени резервации. Заявката на потребителя е оцветена в син цвят, а възможните сценарии за съществуващи резервации са оцветени в розов цвят, като всеки един от тях е наименуван с конкретна буква [A-F].



Фиг. 14

Дадено място X ще бъде свободно във въведения времеви интервал I_X от потребителя $\Leftrightarrow I_X$ има празно сечение с обединението на всички направени резервации за даденото място: X е свободно $\Leftrightarrow I_X \cap \left(\bigcup_{R_X \in DB} R_X \right) = \emptyset$, където R_X

е вече направена резервация за място X .

За да бъде изпълнено условието по-горе, трябва в базата данни да не съществуват резервации от тип B, C, D и F за конкретното място X . Наличието на поне един от тези типове резервации ще определи мястото като заето и ще бъде маркирано с червен цвят. Резервациите от тип A и E, не възпрепятстват добавянето

на нова резервация в базата данни. Това означава, че конкретното паркомясто ще бъде маркирано със зелен цвят като свободно. Когато в базата данни не съществуват резервации за дадено място X , то ще бъде маркирано като свободно.

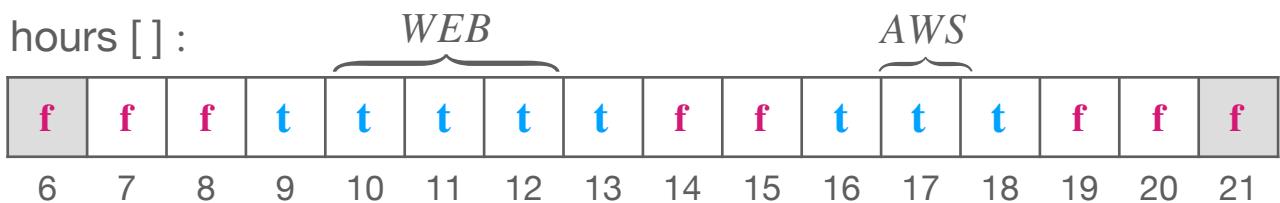
A2. Алгоритъм за обединяване на интервали от занятия:

За да може даден потребител да направи резервация за паркомясто в интервал $[a, b]$, където $a, b \in \mathbb{Z}$ са цели числа, за които е изпълнено условието $7 \leq a < b \leq 21$, е необходимо желаният интервал от резервацията да се съдържа в L , където L е сечението на всички времеви интервали, в които даден преподавател има лекция или упражнение, като всеки отделен интервал се разширява допълнително с по един час в началото и края. Например ако даден преподавател има в програмата си лекция от 10 до 12 и упражнение от 17 до 18, то възможните интервали, в който да може да направи резервация за паркомясто ще са съответно [9-13] и [16-19] (Фиг. 15).

Щатен преподавател: доц. д-р Милен Петров

Лекция: WEB Технологии: [10 – 13], 2021-06-10

Упражнение: Амазон Уеб Услуги (AWS): [17 – 18], 2021-06-10



Фиг. 15

Пояснения: В първата и последната позиция в масива от фиг. XX са изобразени със светло сив цвят, тъй като те ще се запълнят с **t** (true) или **f** (false), в зависимост от това дали съответно във второто или предпоследното блокче преподавателя има занятие. Системата не дава възможност за запазване на паркомясто във времеви интервали съответстващи на тези позиции ([6-X] и [X-22]), но тези позиции от масива (6 и 21) се използват за удобство при имплементацията.

Масивът `hours[]` играе ролята на операцията обединение на интервали за лекции и упражнения. Например, ако в този масив на позиция i има true, това означава, че даденият преподавател, за дадения ден може да запази паркомясто с начален час i и краен час $i + 1$ (в случай, че то вече не е резервирано). Т.е. $\text{hours}[i] == \text{true} \Leftrightarrow$ даденият преподавател може да паркира на даденото място за интервала $[i, i + 1]$, ако то не резервирано. Даденият преподавател може да паркира на дадено паркомясто в интервала $[i, j]$, където $i < j$, ако е изпълнено, че за всяко k , за което $i \leq k \leq j$: $\text{hours}[k] == \text{true}$.

В случай, че доц. Петров направи заявка за свободни места за дата 2021-06-10 за времевия интервал [15-18] системата ще провери дали в базата данни вече има направени резервации за този интервал и ще ги съхрани, за да се оцветят по-късно в зелено или червено според статуса им. След това ще провери дали преподавателят има лекция или упражнение за интервала от заявката и ако няма (както е в случая) ще маркира всички паркоместа с номера от 0 до 6 в сиво,

Назад към НАЧАЛО

като невъзможни за избор. Това бяха местата, които са позволени само за преподаватели, които имат занятия в желания интервал за паркиране. След това ще оцвети останалите места според статуса им.

Заявка: 2021-06-10 [15 – 18]

Фиг. 16

Избран времеви интервал
Дата: 2021-06-10; Интервал: 15–18

ТЪРСИ

ЗОНА A ЗОНА B ЗОНА C ЗОНА D

B0	B1	B2	B3	B4
B5	B6	B7	B8	B9

Заявката от фиг. 16 е такава, тъй като доц. Петров е искал да провери всички места, на които може да паркира за времевия интервал [15-18], но времевите интервали, в които той има занятия за този ден са [10-13] и [17-18]. Следователно толеранса от един час в началото на втория интервал не му достига, за да обхване началото на интервала от заявката ($15 < 17 - 1$). Това означава, че системата ще го тълкува като обикновен студент и той няма да има достъп до паркоместата с номера от 0 до 6, които са предназначени само за преподаватели, които имат лекции в интервалите на заявката. От фигура 16 може да заключим, че позиции от B0 до B6 не са позволени за паркиране за посочения времеви интервал, тъй като доц. Петров няма занятие тогава, B7 и B9 са свободни да бъдат запазени, а B8 вече е резервирано от друг потребител на системата (студент или преподавател).

Друг пример за заявка е този от фиг. 17 по-долу. В него интервала от заявката за справка се обхваща от интервалите на занятията.

Заявка: 2021-04-10 [9 – 13]

Избран времеви интервал
Дата: 2021-06-10; Интервал: 9–13

ТЪРСИ

ЗОНА A ЗОНА B ЗОНА C ЗОНА D

B0	B1	B2	B3	B4
B5	B6	B7	B8	B9

Фиг. 17

В тази заявка, системата е проверила, че преподавателят има занятия в интервала от заявката и е отключила първите 7 места, но три от тях са вече резервиирани от други преподаватели, които имат занятия в този интервал – B1, B3 и B4. Отново има достъп до останалите три места, но от тях отново B8 е заето. Окончателно, всички свободни места са оцветени със зелен цвят

11. Принос на студента

Усилията по разработването на FMI Parking System са разпределени равномерно между разработчиците от екипа. Стилизацията и дизайна на приложението е резултат от групово взимане и имплементиране на решения, идеи и екипна работа. По-ясно разделение на работния процес може да се долови в реализирането на отделните главни функционалности. Тъй като приложението работи синхронно, както потребителя ще очаква, то въпреки формалното разделение, и в реализацията на отделните функционалности има принос на всеки един от членовете на разработващия екип.

Име	Главни функционалности
Андрей	Ф1, Ф2, Ф3
Светлин	Ф4
Калоян	Ф5

12. Ограничения и възможности за бъдещо развитие

1. Секция „Моят акаунт“:
 - ◆ Промяна на име, фамилия, имейл и парола;
 - ◆ Добавяне и промяна на снимка на профила чрез качване на снимка от машината на потребителя или избиране на предефинирана снимка от файловата система на сървъра;
 - ◆ Избор на информацията, която да се визуализира в страница „Моят акаунт“ (Първите няколко предстоящи занятия за дадения ден / Първите няколко резервации за паркоместа за дадения ден / Хороскоп / Прогноза за времето / Интерактивна игра тип судоку);
 - ◆ Новини и събития от и за факултета.
2. Секция „Паркинг зони“:
 - ◆ Избор на паркинг зона директно от картата;
3. Секция „Запази паркомясто“:
 - ◆ Маркиране на резервациите на текущия потребител с оранжев цвят цвят;
4. Секция „Моят график“:
 - ◆ Изтриване на занятие от програмата;
 - ◆ Изтриване на резервация за паркомясто;
 - ◆ История на минали занятия от програмата;
 - ◆ История на минали резервации за паркоместа;
5. Нови типове потребители на системата – „Администратор“:
 - ◆ Ще осъществява регистрацията на нови потребители в системата (както е в Moodle);
 - ◆ Ще качва CSV файловете с програмите на различните преподаватели;
 - ◆ Ще качва разписанията за студентите и по този начин ще ограничи възможността им за запазване на паркомясто в произволен интервал;
6. Чат на живо с други активни потребители в системата.

13. Какво научихме

В рамките на FMI Parking System, екипа по разработването на проекта успя да обогати познанията си в WEB технологиите както и екипната си работа. Някои по-конкретни неща, които научихме са следните:

1. Обработка на файлове с PHP;
2. Работа с JavaScript fetch API, чрез който правим асинхронни заявки към сървъра. Когато чакаме за отговор от сървъра във JSON формат не трябва да извеждаме (echo, var_dump, print) обекти, тъй като ще се върнат заедно с отговора и ще нарушият структурата на JSON формата, което ще доведе до грешка и заявката няма да се изпълни както се очаква.
3. Научихме, че е много трудно да се направят няколко функционалности (дори и с много сложна логика) да работят в синхрон както се очаква от потребителя.

14. Използвани източници

- [1] "How TO - Slideshow, Slideshow / Carousel", последно редактиран: 2020, последно посетен: 2021-06-10,
https://www.w3schools.com/howto/howto_js_slideshow.asp
- [2] **Claudia Romano, Sebastiano Guerriero**, "Simple Confirmation Popup", създаден: 2014, последно редактиран: 2021, последно посетен: 2021-06-10,
<https://codyhouse.co/gem/simple-confirmation-popup>
- [3] **Dani Krossing**, "Upload Files and Images to Website in PHP", публикуван: 2016-12-01, последно посетен на: 2021-06-10,
<https://www.youtube.com/watch?v=JaRq73y5MJk>
- [4] The PHP Group, PHP Documentation, публикувано: 2001, последно редактирано: 2021, последно посетен: 2021-06-10,
<https://www.php.net/manual/en/index.php>