Torta Webshop

Készítette: Veres Szabolcs, Nenad Tasic
és Suti Zsolt

Az oldalról

- A csapatunk egy cukrászdának a webshopját készítette el, ahol regisztráció, majd belejentkezés után lehetőséget kap a felhasználó a vásárlás, és megrendelés lebonyolítására.
- Az webshop elkészítéséhez a spring boot és az angular állt rendelkezésünkre.

Torta választék

Csoki torta

Fő hozzávaló: Csoki

Kiszerelés: 12 Szelet

Ára: 12.000 Ft

Erdei torta

Fő hozzávaló: Erdei gyümölcsök

Kiszerelés: 12 Szelet

Ára: 16.000 Ft

Fehér csokoládé

Fő hozzávaló: Fehér csoki

Kiszerelés: 12 Szelet

Ára: 21.000 Ft

Idővonal

2022 Július

Megterveztük a webshopot

2022 Szeptember

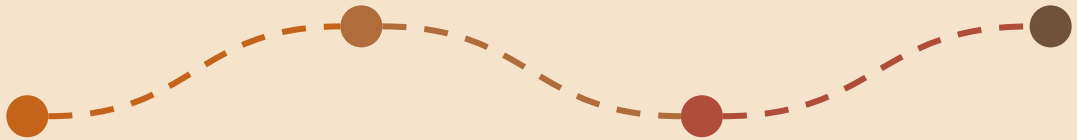
Elkészült a backend

2022 November

Végeztünk a frontend nagy részével és tökéletesítettük az adatbázist

2023 Január

Befejeztük a designt is, ezzel végeztünk a feladattal



Backend

- A backendhez spring boot-t használtunk, ami egy JAVA alapú szerveroldali framework. Általában bankok rendszereiben használják kiemelkedő biztonsági funkciói miatt.
- A munka nagy részét automatikusan kezeli, így nekünk nagyrészt kizárólag a pontot kellett felrakni az “i”-re.

```
3 usages
@Autowired
private OrderDetailsRepo orderDetailsRepo;

1 usage
@Autowired
private CakeBasketRepo cakeBasketRepo;
1 usage  Sabolc120
@Override
public OrderDetails completeOrderPost(
                                    Long userId, String firstName, String lastName, String customerPhoneNumber, String customerEma
                                    String address, String userName) {
    OrderDetails orderDetails = new OrderDetails();
    UserModel userModel = userRepo.findById(userId).orElse( other: null);
    orderDetails.setCustomerPhoneNumber(customerPhoneNumber);
    orderDetails.setCustomerEmail(customerEmail);
    orderDetails.setFirstName(firstName);
    orderDetails.setLastName(lastName);
    orderDetails.setUserModel(userModel);
    orderDetails.setCity("Szeged");
    orderDetails.setAddress(address);
    orderDetails.setUserName(userName);
    return orderDetailsRepo.save(orderDetails);

}

1 usage  Sabolc120
@Override
public List<OrderDetails> getOrders() { return orderDetailsRepo.findAll(); }
```

```
@Autowired
private com.example.cakeExamBackend.Repositories.Dao.basketDao basketDao;

1 usage
@Autowired
private CakeRepo cakeRepo;
1 usage  Sabolc120
@Override
public CakeBasketModel addCakeIntoBasket(String cake_img, String cake_main_ingredient, String cake_name,
                                         Integer cake_price, Integer cake_quantity, Long cake_id, Long user_id,
                                         Integer candle, Boolean placedOrder, String userName) {
    CakeBasketModel cakeBasketModel = new CakeBasketModel();
    CakeModel cakeModel = cakeRepo.findById(cake_id).orElse( other: null);
    UserModel userModel = userRepo.findById(user_id).orElse( other: null);
    cakeBasketModel.setCakeImg(cake_img);
    cakeBasketModel.setCakeQuantity(1);
    cakeBasketModel.setCakePrice(cake_price);
    cakeBasketModel.setCakeMainIngredient(cake_main_ingredient);
    cakeBasketModel.setUserModel(userModel);
    cakeBasketModel.setCakeModel(cakeModel);
    cakeBasketModel.setCakeName(cake_name);
    cakeBasketModel.setPlacedOrder(false);
    cakeBasketModel.setCandle(candle);
    cakeBasketModel.setUserName(userName);
    return basketDao.save(cakeBasketModel);
}

1 usage  Sabolc120
@Override
public List<CakeBasketModel> getCakesInBasket(Long user_cakes_in_basket) {
```

Frontend

- Frontendhez angulart használtunk, amire azért esett a választás, mert könnyedén modulokra lehet bontani a weboldal részeit.
- Az elkészítés folyamata hasonlított egy kirakóhoz, ahol a darabok voltak a modulok, amelyek miután a helyükre kerültek, megmutatták az összképet.
- Az angular célja általában az SPA (Single Page Application) webalkalmazások létrehozása.

```
templateUrl: './register-page.component.html',
styleUrls: ['./register-page.component.css']
)
xport class RegisterPageComponent implements OnInit {

constructor(private userRegService:UserServiceService,
  private router:Router) { }

registerUser(registerForm:NgForm){
  console.log(registerForm.value);

  this.userRegService.registerUser(registerForm.value).subscribe(
    (resp) =>{
      console.log(resp);
      this.router.navigate(['loginPage'])
    },
    (err) =>{
      console.log(err);
    }
  );
}

ngOnInit(): void {
}
```

```
  styleUrls: ['./login-page.component.css']
})
export class LoginPageComponent implements OnInit {

  constructor(private userService: UserServiceService,
    private userAuthService:UserAuthService,
    private router:Router) { }

  ngOnInit(): void {
  }
  login(loginForm:NgForm){
    this.userService.login(loginForm.value).subscribe(
      (resp:any) =>{
        this.userAuthService.setRoles(resp.userModel.authorities);
        this.userAuthService.setToken(resp.jwtToken)
        this.userAuthService.setIdUser(resp.userModel.id)
        this.userAuthService.setUserName(resp.userModel.userName)
        const role = resp.userModel.authorities;
        console.log(resp);
        if(role === 'USER'){
          this.router.navigate(['mainPage'])
        }
        else if(role === 'ADMIN'){
          this.router.navigate(['mainPage']);
        }
      },
      (err) =>{
        console.log(err);
      }
    );
  }
```

Design

- A design kiválasztása és elkészítése volt az a rész, ahol nagyon sokat kellett változtatni.
- Törekedtünk arra hogy szép, látványos és könnyen kezelhető legyen mindenki számára.
- A képeket AI segítségével generáltuk.

A rendelés menete

Köszönjük!