Univerzitet u Beogradu

Fakultet organizacionih nauka

Katedra za elektronsko poslovanje

ReactJS

Domaći zadatak 3

|  |  |  |
| --- | --- | --- |
| Ime | Prezime | Broj indeksa |
|  |  |  |
| Link ka Github-u |  | |

Sadržaj

[1 Korisnički zahtev 3](#_Toc117932679)

[2 Implementacija 3](#_Toc117932680)

[3 Korisničko uputstvo 3](#_Toc117932681)

# Korisnički zahtev

Potrebno je napraviti veb aplikaciju za rad sa aukcijama i proizvodima. Aplikacija treba da se osloni na api iz prvog domaceg cije putanje ce pozivati i predstavlja admin panel.

Potrebno je imati stranice za login i registraciju korisnika. Ukoliko je korisnik ulogovan, potrebno mu je prikazati stranice za pretragu i kreiranje aukcija kao i stranicu za pretragu, kreiranje i izmenu proizvoda.

# Implementacija

Za cuvanje podataka o trenutno ulogovanom korisniku, kreirali smo kontekst na vrhu aplikacije. Pored toga, u kontekstu smo dodali i metode za login, registraciju i logout.

export function AuthContextProvider(props: React.PropsWithChildren) {

    const [user, setUser] = useState<User | undefined>(undefined);

    const [loading, setLoading] = useState(true);

    const [error, setError] = useState('');

    const navigate = useNavigate();

    useEffect(() => {

        if (!loading) {

            return;

        }

        const token = localStorage.getItem('token');

        if (!token) {

            setLoading(false);

            return;

        }

        axios.get('/api/user', { headers: { Authorization: 'Bearer ' + token } })

            .then(res => {

                setUser(res.data);

                axios.defaults.headers.common.Authorization = 'Bearer ' + token;

            })

            .catch(err => {

            })

            .finally(() => {

                setLoading(false)

            })

    }, [loading])

    const logout = async () => {

        await axios.post('/api/logout')

        setUser(undefined);

        axios.defaults.headers.common.Authorization = undefined;

        localStorage.removeItem('token');

    }

    const login = async (email: string, password: string) => {

        try {

            const res = await axios.post('/api/login', { email, password });

            setUser(res.data.user);

            localStorage.setItem('token', res.data.token);

            axios.defaults.headers.common.Authorization = 'Bearer ' + res.data.token;

        } catch (error) {

            if (axios.isAxiosError(error)) {

                setError(error.response?.data.message);

            } else {

                setError('Something went wrong');

            }

        }

    }

    const register = async (u: RegisterUser) => {

        try {

            const res = await axios.post('/api/register', u);

            setUser(res.data.user);

            localStorage.setItem('token', res.data.token);

            axios.defaults.headers.common.Authorization = 'Bearer ' + res.data.token;

            navigate('/')

        } catch (error) {

            if (axios.isAxiosError(error)) {

                setError(error.response?.data.message);

            } else {

                setError('Something went wrong');

            }

        }

    }

    if (loading) {

        return null;

    }

    return (

        <AuthContext.Provider

            value={{

                user,

                login,

                setError,

                logout,

                register,

                error

            }}

        >

            {props.children}

        </AuthContext.Provider>

    )

}

Na stranici aukcija ima 2 stanja:

* Spisak aukcija koje zadovoljavaju pretragu
* Objekat za pretragu aukcijia
* const [auctions, setAuctions] = useState<Pagination<Auction> | undefined>(undefined);
* const [params, setParams] = useState({
* from: '',
* to: '',
* userId: '',
* page: 0,
* size: 20
* })

Pored toga, imamo i useEffect koji ce na svaku promenu stanja params poslati zahtev ka serveru da dohvati aukcije.

const fetchAuctions = async () => {

        const res = await axios.get('/api/auctions', {

            params: {

                from: params.from || undefined,

                to: params.to || undefined,

                userId: params.userId || undefined,

                page: params.page,

                size: params.size

            }

        });

        setAuctions(res.data);

    }

    useEffect(() => {

        fetchAuctions();

    }, [params])

Aukcije su prikazane u tabeli a proizvode smo iscrtali u vidu kartice.

export default function ProductCard(props: Props) {

    return (

        <div className="card" >

            <div className='card-header d-flex justify-content-between'>

                <strong>{`ID: ${props.product.id}`}</strong>

                {

                    props.product.sold && (

                        <strong >

                            Sold

                        </strong>

                    )

                }

            </div>

            <img className="card-img-top" src={props.product.image} alt="Product image" />

            <div className="card-body">

                <h5 className="card-title">{props.product.name}</h5>

                <p className="card-text">{props.product.description}</p>

            </div>

        </div>

    )

}

U formi za rad sa proizvodom imamo stanje koje predstavlja sta pise u kom inputu. Pored toga, imamo i props product na ciju promenu reagujemo tako sto popunjavamo stanje sa tim podacima.

interface Props {

    product?: Product,

    onSubmit: (p: CreateProduct) => void

}

const emptyProduct = {

    description: '',

    image: '',

    name: '',

    sold: false

}

export default function ProductForm(props: Props) {

    const [productData, setProductData] = useState<CreateProduct>(emptyProduct)

    useEffect(() => {

        if (props.product) {

            setProductData({

                description: props.product.description,

                name: props.product.name,

                image: props.product.image,

                sold: props.product.sold,

            })

        } else {

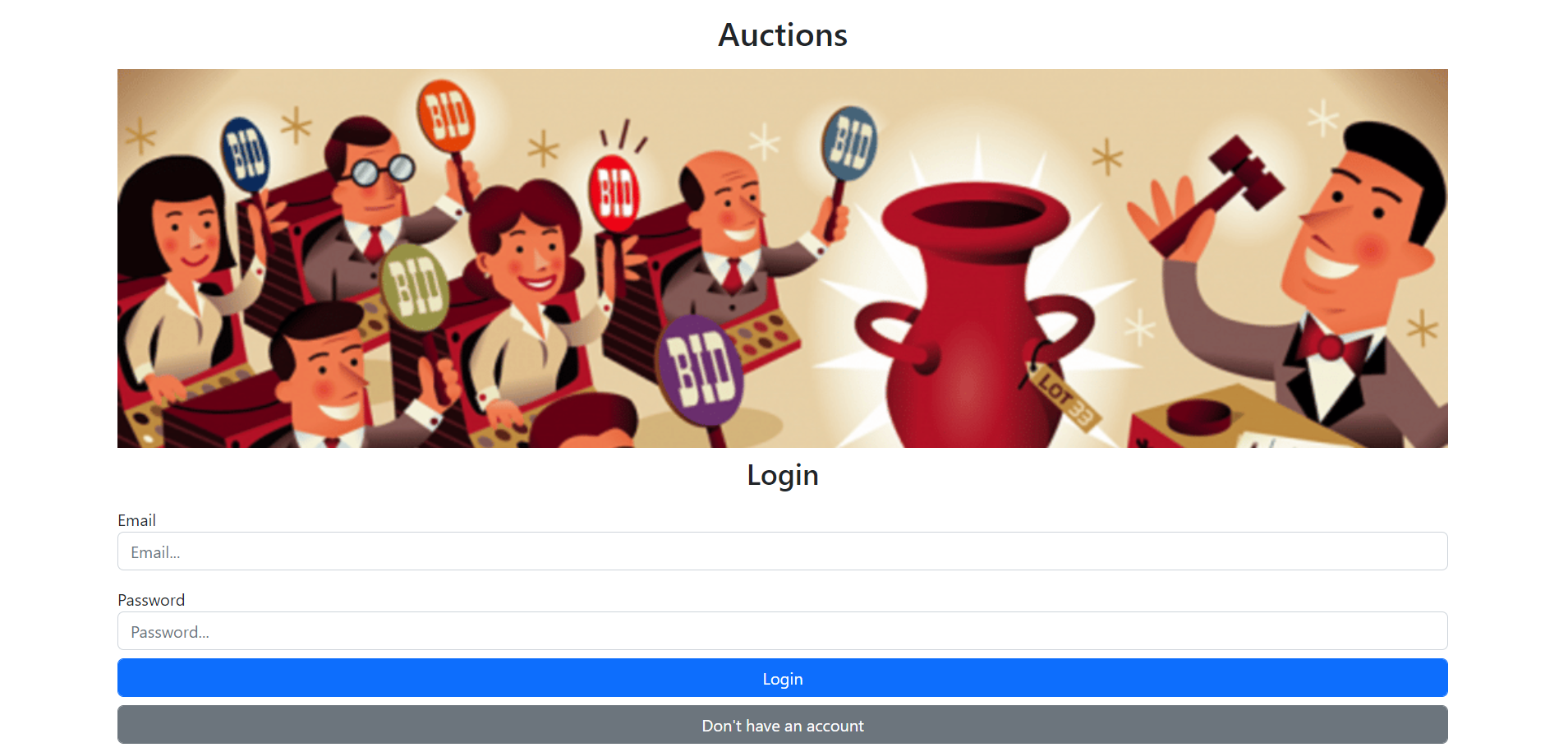
            setProductData(emptyProduct)

        }

    }, [props.product])

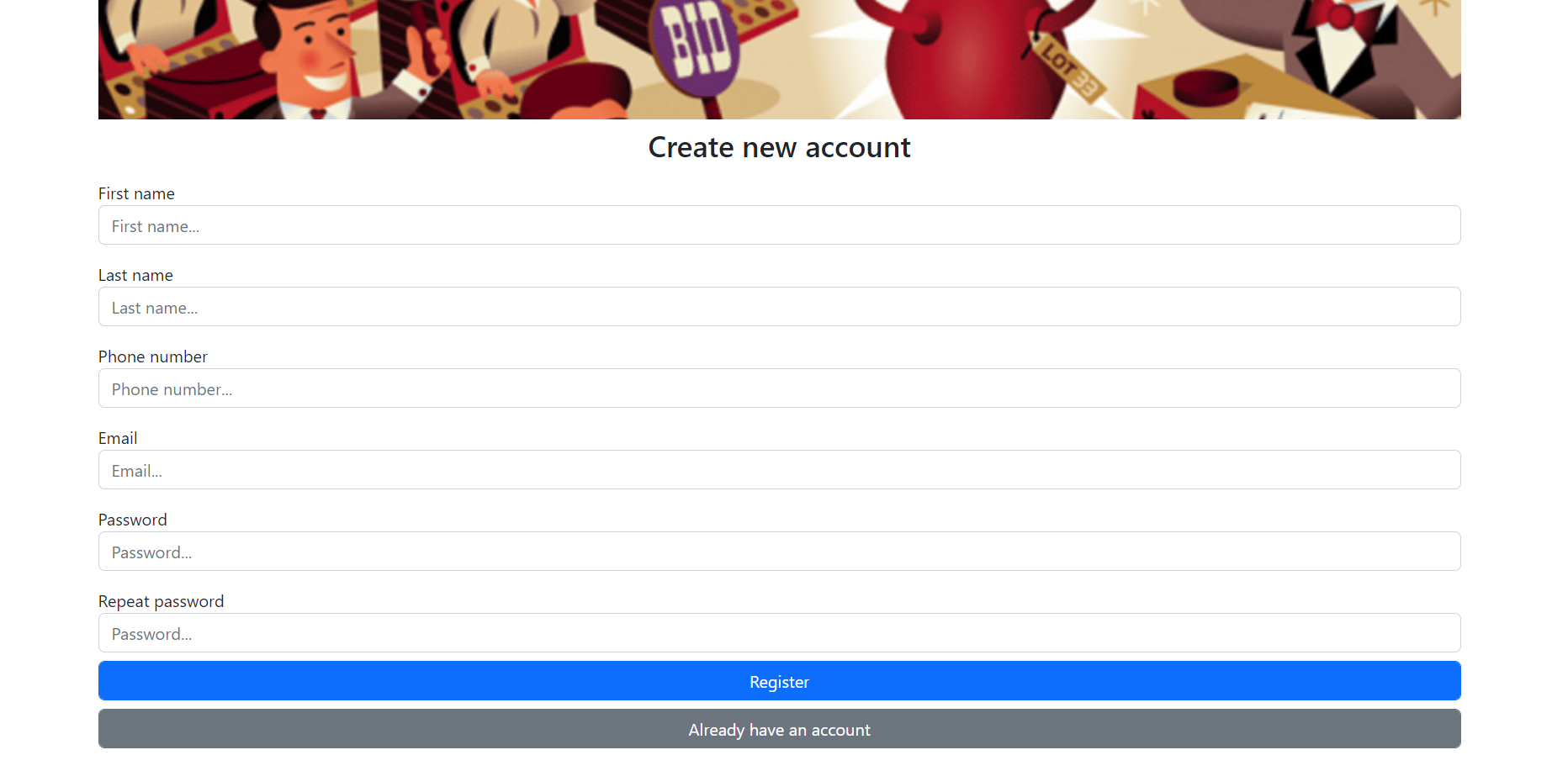
# Korisničko uputstvo

Ukoliko korisinik nije ulogovan, prikazace mu se stranica za login (slika 1).



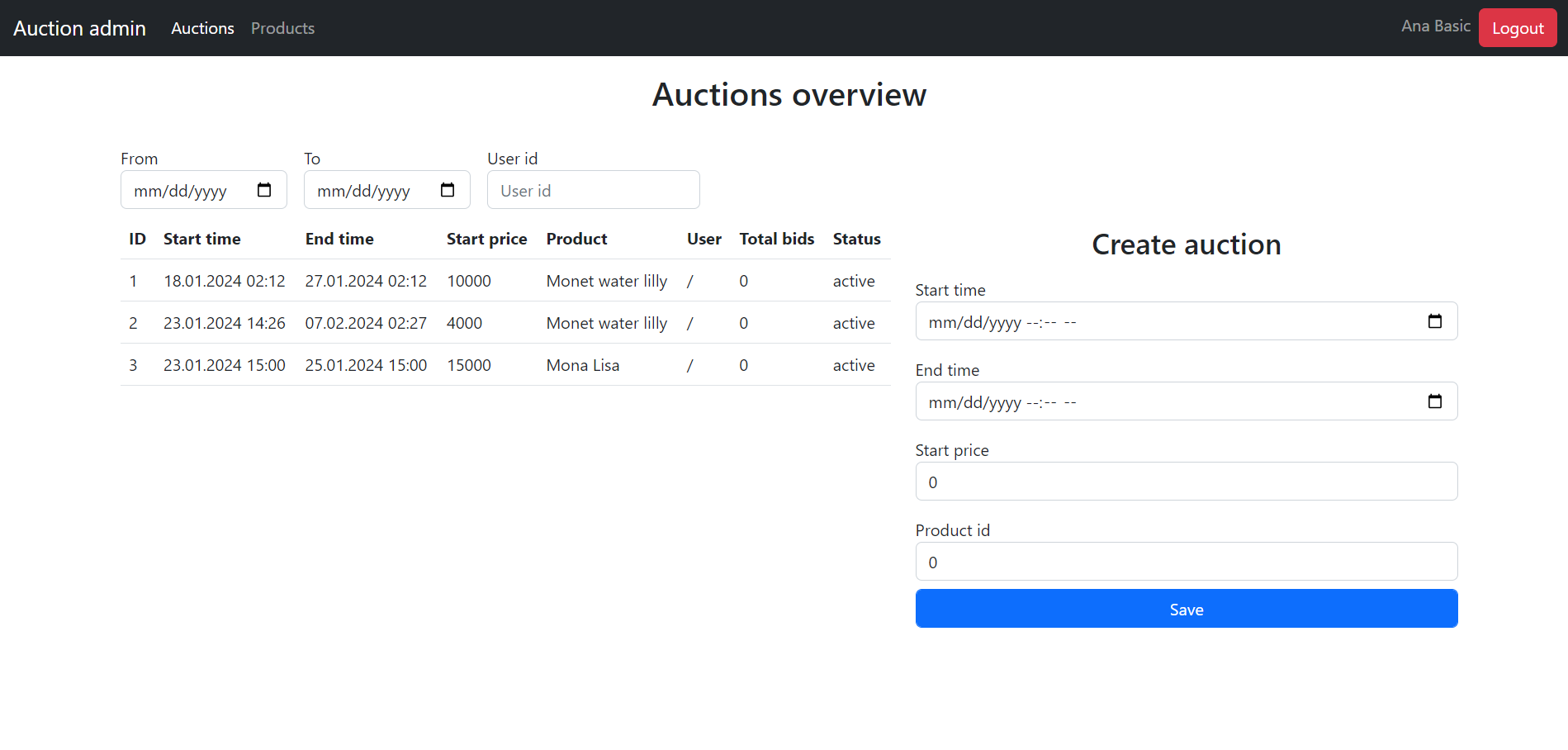
Slika 1 Login stranica

Ukoliko korisnik nema nalog, moze se odvesti na stranicu za registraciju klikom na dugme.



Slika 2 Stranica za registraciju

Nakon uspesnog logovanja, prikazace mu se stranica za rad sa aukcijama (slika 3).

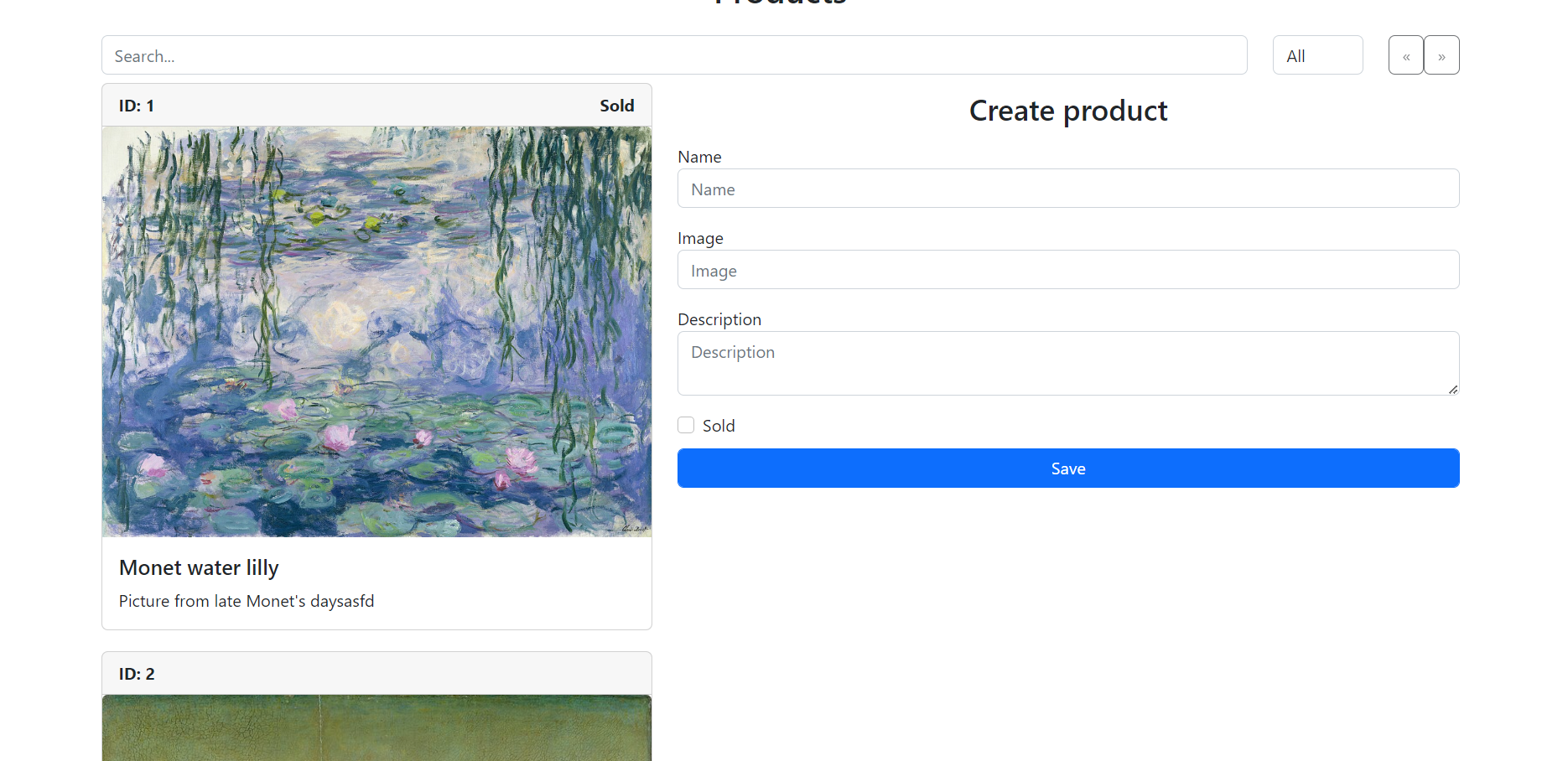


Slika 3 Stranica sa aukcijama

Sa leve strane moze videti tabelu svih aukcija. Iznad tabele nalaze se filteri.

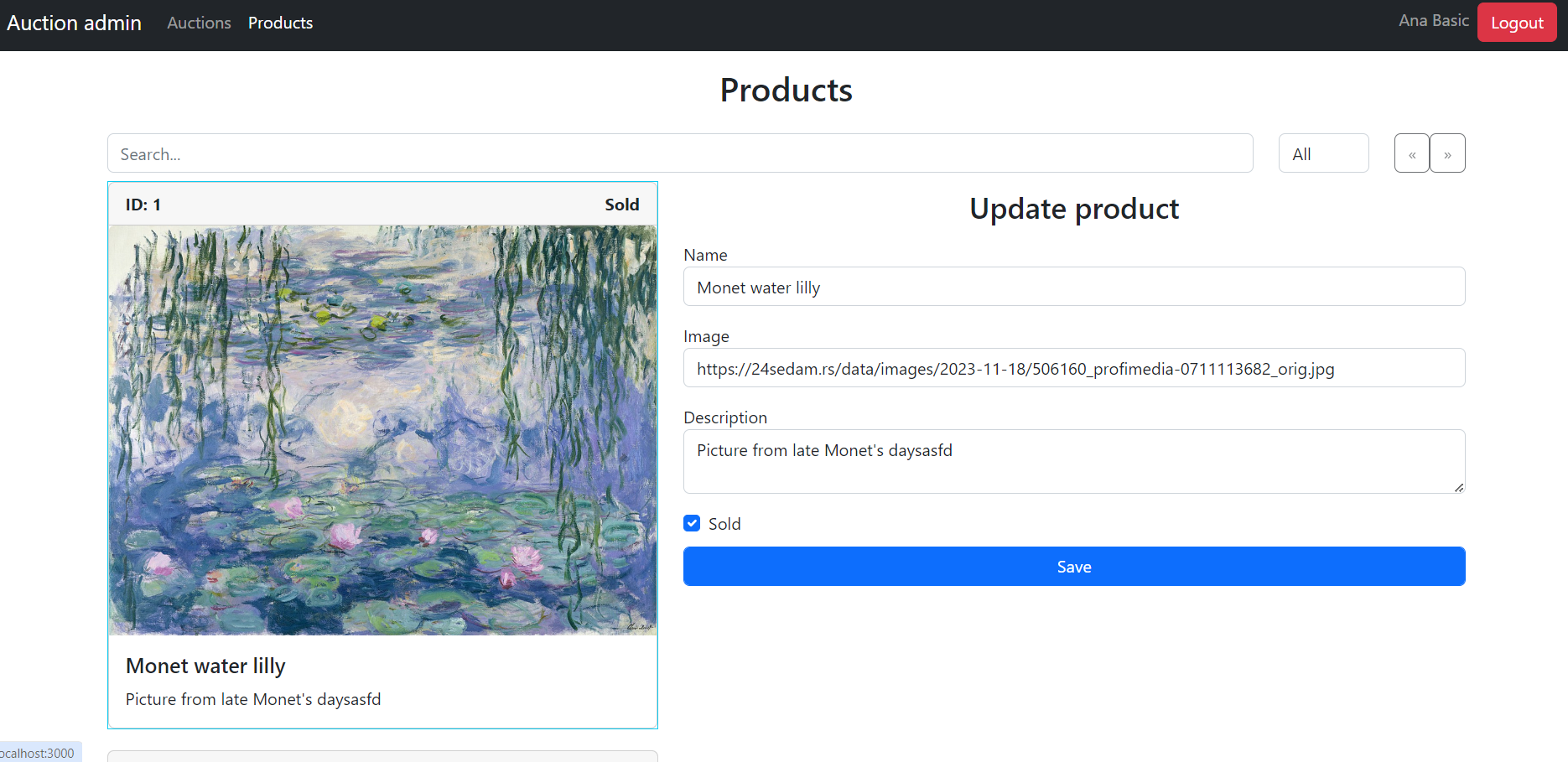
Sa desne strane nalazi se forma za kreiranje aukcije.

Na drugoj strani korisnik moze videti sve proizvode (slika 4).



Slika

Sa leve strane nalaze se svi proizvodi a sa desne forma za kreiranje proizvoda. Klikom na proizvod, forma se popunjava podacima datog proizvoda i u tom slucaju sluzi za izmenu proizvoda (slika 5)



Slika