**//KORISNICI:**

**//registracija korisnika**

app.post('/registracija', async (req,res)=>{

    const user = new User(req.body)

    try{

        await user.save()

        const token = await user.getToken()

        res.status(201).send({user, token})

    }catch(error){

        res.status(400).send(error)

    }

})

**//prijava korisnika**

app.post('/prijava', async (req,res)=>{

    try{

        const user = await User.findByEmailAndPw(req.body.email, req.body.password)

        const token = await user.getToken()

        res.send({user, token})

    }catch(error){

        res.status(400).send()

    }

})

**//odjava**

app.post('/odjava', auth, async(req,res)=>{

    try{

        req.user.tokens = req.user.tokens.filter((token)=>{

            return token.token !== req.token

        })

        await req.user.save()

        res.send()

    }catch(error){

        res.status(500).send()

    }

})

**//profil korisnika**

app.get('/profil', auth, async (req,res)=>{

    res.send(req.user)

})

**//promjena lozinke**

app.patch('/promjena-lozinke', auth, async (req,res)=>{

    const updates = Object.keys(req.body)

    const canUpdate = ['password']

    const isValid = updates.every((update) => canUpdate.includes(update))

    if(!isValid){

        return res.status(400).send({error: 'Nije moguće promijeniti lozinku'})

    }

    try{

        updates.forEach((update) => req.user[update] = req.body[update])

        await req.user.save()

        res.send(req.user)

    }catch(error){

        res.status(400).send(error)

    }

})

**//brisanje korisnika**

app.delete('/obrisi-profil', auth,  async (req,res)=>{

    try{console.log('mmm')

        await req.user.remove()

        console.log('aaa')

        res.send(req.user)

    }catch(error){

        res.status(500).send(error)

        console.log(error)

    }

})

**//OBJAVE:**

**//sve objave (home page)**

app.get('/', async (req,res)=>{

    try{

        const posts = await Post.find({})

        res.send(posts)

    }catch(error){

        res.status(500).send(error)

    }

})

//nova objava

app.post('/nova-objava', auth, async (req,res)=>{

    const post = new Post({...req.body, postedBy: req.user.\_id})

    try{

        await post.save()

        res.status(201).send(post)

    }catch(error){

        res.status(400).send(error)

    }

})

**//pretraga objave po gradu**

app.get('/objava/filter', auth, async (req,res)=>{

    try{

        const filter={}

    if(req.query.filter != 'undefined' && req.query.filter != null){

        let sea = new RegExp(`^.\*${req.query.filter}.\*$`, "img")

        filter['grad'] = sea

    }

        const onePost = await Post.find(filter)

        res.send(onePost)

    }catch(error){

        res.status(500).send()

    }

})

**//pronadi objavu po id**

app.get('/objava/:id', auth, async (req,res)=>{

    const \_id = req.params.id

    try{

        const post = await Post.findOne({\_id, postedBy: req.user.\_id})

        if(!post){

            return res.status(404).send()

        }

        res.send(post)

    }catch(error){

        res.status(500).send()

    }

})

**//moje objave**

app.get('/moje-objave', auth, async (req,res)=>{

    try{

        await req.user.populate('posts').execPopulate()

        res.send(req.user.posts)

    }catch(error){

        res.status(500).send()

    }

})

**//uredi objavu po id**

app.patch('/objava/uredi/:id', auth, async (req,res)=>{

    const updates = Object.keys(req.body)

    const canUpdate = ['cijena','zupanija','grad','naselje','ljubimci','dostupnoGod','terasa','ukljuceneRezije','odvojenaSoba']

    const isValid = updates.every((update) => canUpdate.includes(update))

    if(!isValid){

        return res.status(400).send({error: 'Nije moguće urediti objavu'})

    }

    try{

        const post = await Post.findOne({ \_id: req.params.id, postedBy: req.user.\_id})

        if(!post){

            return res.status(404).send()

        }

        updates.forEach((update) => post[update] = req.body[update])

        await post.save()

        res.send(post)

    }catch(error){

        res.status(400).send(error)

    }

})

**//brisanje objave po id**

app.delete('/objava/obrisi/:id', auth,  async (req,res)=>{

    try{

        const post = await Post.findOneAndDelete({ \_id: req.params.id, postedBy: req.user.\_id})

        if(!post){

            res.status(404).send()

        }

        res.send(post)

    }catch(error){

        res.status(500).send(error)

    }

})