1.
2.
3.
4.
5.
6.

1.
2. find findOne
3.

Mongoose

```
const mongoose = require('mongoose')

var eventSchema = new mongoose.Schema({
    eventName: {type: 'string', required: true},
    dateTime: {type: 'Date', required: true},
})
var Event = mongoose.model('Event', eventSchema)
```

```
const myEvent = new Event({
    eventName: 'Make School Demo Night',
    dateTime: new Date(2020, 2, 6, 18, 30, 0) // March 6, 2020 at 6:30PM
})

myEvent.save() // Save to our Mongoose database
```

Promise

```
Event.findOne({ eventName: 'Make School Demo Night' })
.then(demoNightEvent => {
    console.log(demoNightEvent)
```

```
})
```

```
Event.findOneAndUpdate(
    { eventName: 'Make School Demo Night' }, // How to find the event
    {dateTime: new Date(2020, 6, 6, 18, 30, 0)} // What to change
)
.then(demoNightEvent => {
    console.log(demoNightEvent)
}
```

```
Event.findOneAndDelete({ eventName: 'Make School Demo Night' })
.then(result => {
    console.log('Successfully deleted.')
})
```

starter code `npm install`

`User`

`src/models/user.js`

```
const UserSchema = new Schema({
  username: { type: String, required: true },
  password: { type: String, select: false }
})

const User = mongoose.model('User', UserSchema)

module.exports = User
```

`src/routes/user.js`

```
const User = require('../models/user')

router.get('/', (req, res) => {
    User.find().then((users) => {
        return res.json({users})
    })
    .catch((err) => {
```

```
        throw err.message
    });
})

router.get('/:userId', (req, res) => {
    console.log(`User id: ${req.params.userId}`)
    User.findById(req.params.userId).then((user) => {
        return res.json({user})
    })
    .catch((err) => {
        throw err.message
    });
})

router.post('/', (req, res) => {
    let user = new User(req.body)
    user.save().then(userResult => {
        return res.json({user: userResult})
    }).catch((err) => {
        throw err.message
    })
})

router.put('/:userId', (req, res) => {
    User.findByIdAndUpdate(req.params.userId, req.body).then((user) => {
        return res.json({user})
    }).catch((err) => {
        throw err.message
    })
})

router.delete('/:userId', (req, res) => {
    User.findByIdAndDelete(req.params.userId).then(() => {
        return res.json({
            'message': 'Successfully deleted.',
            '_id': req.params.userId
        })
    })
    .catch((err) => {
        throw err.message
    })
})
```

**Message**

**title** **body** **author**

**Message** **User**

**models/message.js**

```
const MessageSchema = new Schema({
  // ...
  author : { type: Schema.Types.ObjectId, ref: "User", required: true },
```

```
})
```

`models/user.js`

```
const UserSchema = new Schema({
  messages : [{ type: Schema.Types.ObjectId, ref: "Message" }]
})
```

`routes/message.js` `User`

```
/** Route to add a new message. */
router.post('/', (req, res) => {
    let message = new Message(req.body)
    message.save()
    .then(message => {
        return User.findById(message.author)
    })
    .then(user => {
        console.log(user)
        user.messages.unshift(message)
        return user.save()
    })
    .then(_ => {
        return res.send(message)
    }).catch(err => {
        throw err.message
    })
})
```

`m`

```
UserSchema.pre('findOne', function (next) {
    this.populate('messages')
    next()
})

UserSchema.pre('find', function (next) {
    this.populate('messages')
    next()
})
```

**Vibe Check form**

1. **Mongoose Models**
2. **Mongoose Queries**
3. **Mongoose: Working with Dates**
4. **JavaScript Date**
5. **Moment.js**