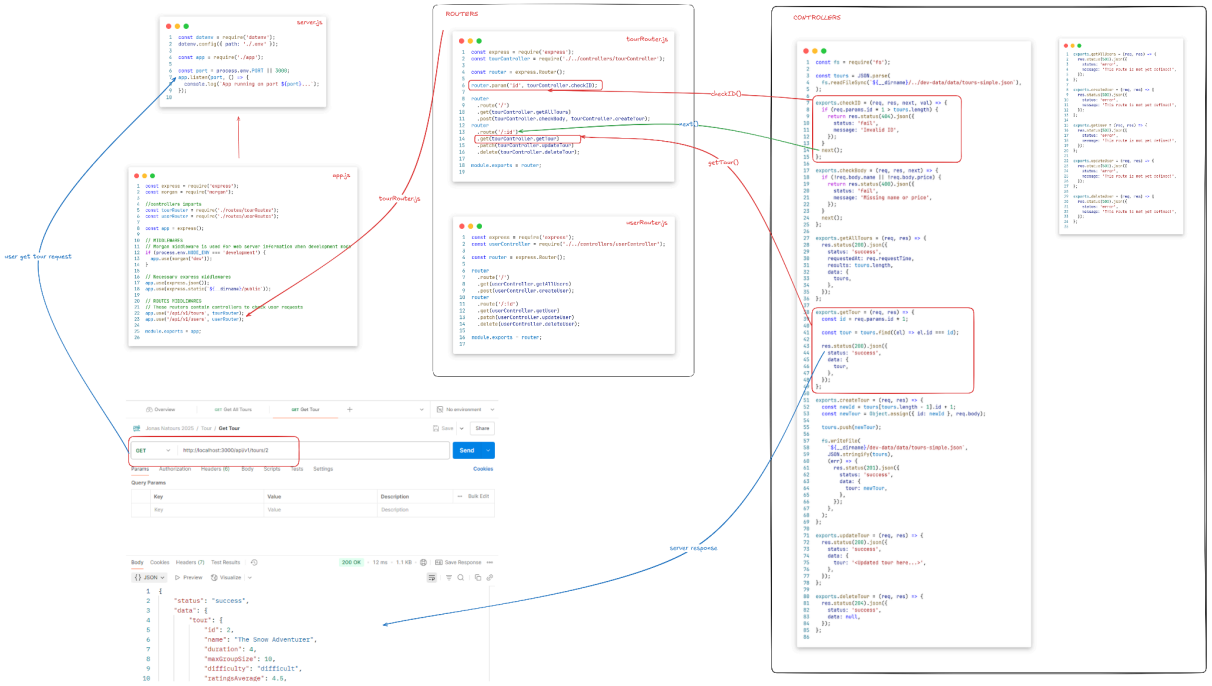
Simple Web Api



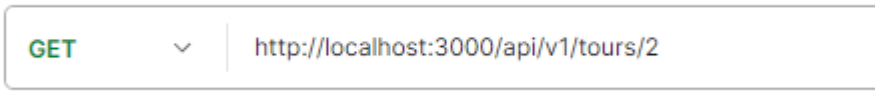
server.js

Web server'ın başlatıldığı dosya.

```
app.listen(port, () => {
  console.log(`App running on port ${port}...`);
});
```

server.js içindeki **app.listen()** metoduyla web server çalışmaya başlar ve kullanıcı taleplerini(request) yanıtlayarak kullanıcıya cevap(response) döner.

Request Response Adımları



Kullanıcı tours kayıtlarından (collection veya tablo) 2 numaralı kaydı talep ediyor.

/api/v1/tours/2

Kullanıcı url olarak parametreler girer. Bu parametreler v1 versiyonlu api istemcisinin tours kayıtlarından 2 numaralı(id) kaydı ister.

```
app.listen(port, () => {
  console.log(`App running on port ${port}...`);
});
```

server.js port değişkeninde gelen (3000 portu) isteği **app.js**'e yönlendirir.

```
app.use('/api/v1/tours', tourRouter);
```

app.js üzerinde **app.use /api/v1/tours** isteklerini yakalar ve bu istekleri **tourRouter.js**'e yönlendirir.

```
const express = require('express');
const tourController = require('./../controllers/tourController');

const router = express.Router();

router.param('id', tourController.checkID);

router
  .route('/')
  .get(tourController.getAllTours)
  .post(tourController.checkBody, tourController.createTour);
router
  .route('/:id')
  .get(tourController.getTour)
  .patch(tourController.updateTour)
  .delete(tourController.deleteTour);

module.exports = router;
```

kullanıcının girdiği /api/v1/tours/2 talebinde 2 numaralı id isteği bulunduğundan kullanıcı isteği önce router.param("id", tourController.checkID) kontrolüne yönlendiriliyor.

```
router.param('id', tourController.checkID);
```

bu router tanımı sayesinde kullanıcıların girdiği her id tabebi öncelikle kayıtlarda var mı diye kontrol edilir. Böylelikle her kontrolde id kontrolüne gerek kalmaz ve hatalı id girişinde veri üzerinde işlem yapılmasını engeller.

```
exports.checkID = (req, res, next, val) => {
  console.log(`Tour id is: ${val}`);

  if (req.params.id * 1 > tours.length) {
    return res.status(404).json({
      status: 'fail',
      message: 'Invalid ID',
    });
  }
  next();
};
```

kullanıcının talep ettiği id'nin verilerde olup olmadığına bakan checkID() fonksiyonu.Bu fonksinda dikkat edilmesi gereken en önemli nokta next() fonksiyonu. Bu durumu şöyle açıklayalım;

Eğer id verileri içinde varsa next() fonksiyonu işletilir ve bir sonraki middleware olan getTour() fonksiyonu işletilir. Eğer id veriler içinde yoksa kullanıcıya kullanıcıya ID bulunamadı mesajı kullanıcıya döner ve bir sonraki kullanıcı istemi beklenmeye başlar.