Experiment 8: Concurrent Ticket Booking System with Seat Locking

```
const express = require('express');
const app = express();
app.use(express.json());

let seats = {
  A1: { booked: false, lockedBy: null, lockExpiresAt: null },
  A2: { booked: false, lockedBy: null, lockExpiresAt: null },
  A3: { booked: false, lockedBy: null, lockExpiresAt: null }
};

function clearExpiredLocks() {
  const now = Date.now();
  for (const key in seats) {
    if (seats[key].lockExpiresAt && seats[key].lockExpiresAt < now) {
      seats[key].lockedBy = null;
      seats[key].lockExpiresAt = null;
    }
  }
}

app.post('/lock/:seatId', (req, res) => {
  const seatId = req.params.seatId;
  const userId = req.body.userId;
  clearExpiredLocks();

  const seat = seats[seatId];
  if (!seat) return res.status(404).send('Invalid seat ID');
  if (seat.booked) return res.status(400).send('Seat already booked');
  if (seat.lockedBy && seat.lockExpiresAt > Date.now())
    return res.status(400).send('Seat is locked by another user');

  seat.lockedBy = userId;
  seat.lockExpiresAt = Date.now() + 60000; // lock for 60 sec
  res.send(`Seat ${seatId} locked by ${userId}`);
});

app.post('/book/:seatId', (req, res) => {
  const seatId = req.params.seatId;
  const userId = req.body.userId;
  clearExpiredLocks();

  const seat = seats[seatId];
```

```
  if (!seat) return res.status(404).send('Invalid seat ID');
  if (seat.booked) return res.status(400).send('Seat already booked');
  if (seat.lockedBy !== userId) return res.status(403).send('Seat not locked by
you');

  seat.booked = true;
  seat.lockedBy = null;
  seat.lockExpiresAt = null;
  res.send(`Seat ${seatId} successfully booked by ${userId}`);
});

app.get('/seats', (req, res) => {
  clearExpiredLocks();
  res.json(seats);
});

app.listen(4000, () => console.log('Booking system running on
http://localhost:4000'));
```