Example: Jolly Roger Airlines (JRA)

- DB to hold the data (services, availability, prices, bookings etc.)

- Scripts to handle various stages of the booking process

Typical booking session

1. Customer submits trip details using a form

2. Script "check_services" interrogates DB for suitable services, packages the results as a HTML page, and returns this to the client

3. The customer views the response and chooses desired flights

4. Script "check_availability" interrogates the DB for prices and availability of selected flights and returns the results as a HTML page

5. The customer submits payment details

6. Another script runs to finish the booking

What's important is that there's a script to advance each stage, and that they are all triggered by client action, and that they all have DB parts and non-DB parts.

We make some simplifications

- Only one-way, point-to-point, single-seat bookings are available

- Every service operates daily, every day in the year

A DB design

services (code, origin, destination, departure, duration, schedule)

flights (code, date, capacity, availability, price)

airports(code, name, timezone)

customers(customer_id, first_name, last_name, email, password)

bookings(booking_code, customer, service, date, credit_card)

A query for checking for suitable services

Imagine a customer is interested in flights from Cork (ORK) to Paris (CDG) on

01/12/2012:

```
SELECT *

FROM services

WHERE services.origin = ORK AND services.destination = CDG;
```

A script embeds the customer's choices (origin and destination here) into a template query.

The selection of dates comes at a later stage because of the way our DB is structured.

Checking availability and price

```
SELECT price, …

FROM

services JOIN flights

ON services.code = flight.code

WHERE

services.origin = ORK

AND services.destination = CDG

AND flight.date = 01/12/2012

AND availability > 0;
```

This brings in the date, and displays prices just for available flights.

Recording bookings

The script for processing bookings would be built around a number of statements:

- Record the customer's details:
  INSERT INTO customers VALUES (customer's id,…);

- Record the details of this booking:
  INSERT INTO bookings VALUES (booking code,…);

- Update the availability into for this flight:
  UPDATE flights

```
SET availability = availability - 1
WHERE flight.code = JR666
AND flight.date = 01/12/2012
```

Potential Problem

With this simplified system, if two people book the last seat on a flight simultaneously, or at least their bookings overlap slightly, then they can both succesfully book the seat.