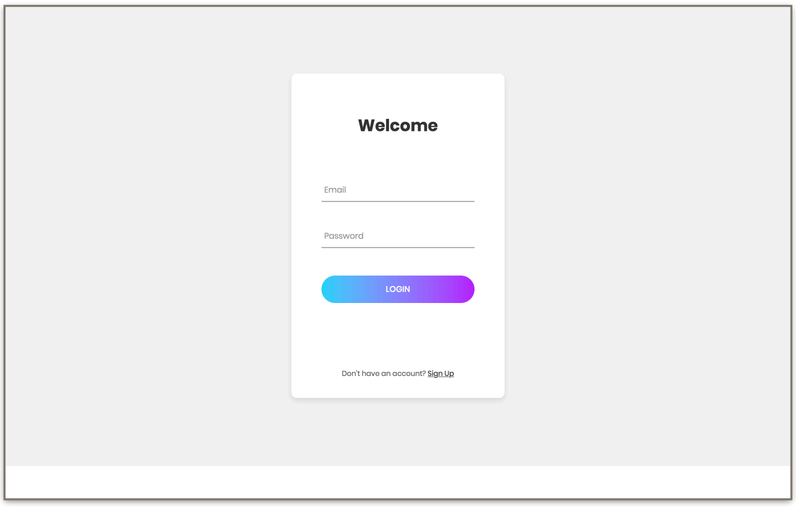
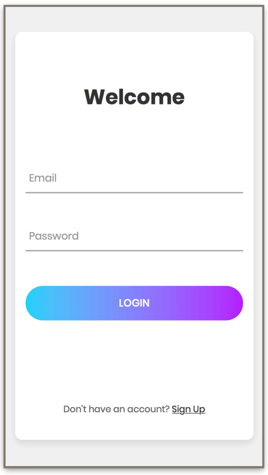
Kings’ Landing

A Technical Overview of the new Check-in System developed for Kings’ Landing

The new Check-in and Seat Selection System which was developed on top of the client’s website was made to provide ease of travel and flexibility to the customers flying KLAir. This systems makes it easier for a passenger to manage their upcoming journeys and select their preferred seats at their fingertips.

Responsive Design





On a desktop and a mobile device

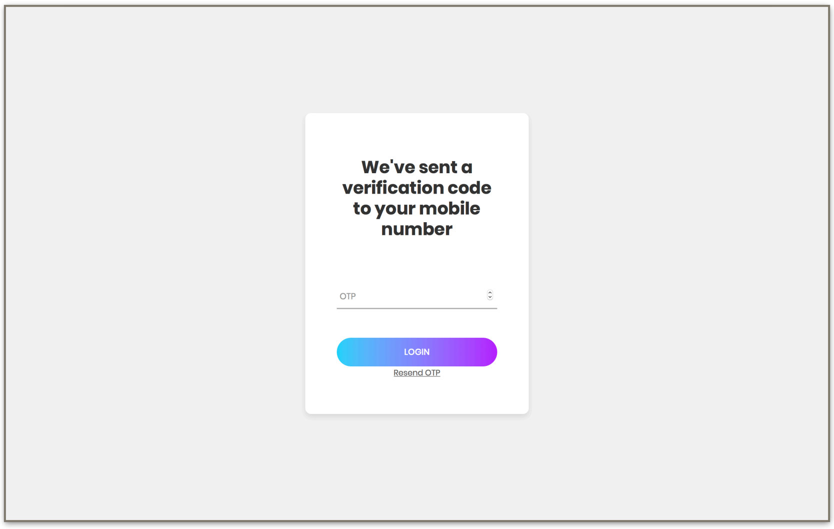
This system has been developed in such a way that it is cross-platform and does not impose any new dependencies as the operating system and operating device change. The website will support any desktop/laptop browser, and will also be automatically scaled and adjusted to support mobile devices like smart phones and tablets.

User friendly

Keeping in mind the vast types of customers KLAir has, the system has been designed in a way that is understandable and operable by all customers. Only minimal user input has to be given and everything else happens automatically under the hood. No confusing menus or misleading content, everything is as simple as ABC.

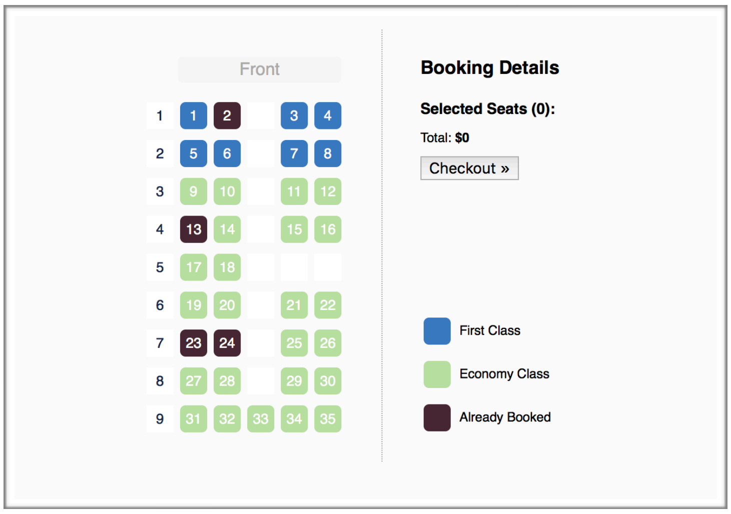
Security

The system boasts tight security on the platform. Two factor authentication is provided to ensure that all of the customer's data is protected and safe. Along with input sanitisation to prevent Injection and Phishing, 2FA ensures that hackers and other undesirable agents do not get inside the system. Apart from this payments are also handled from a secure and sandboxed gateway making sure that all payments go through without a glitch.



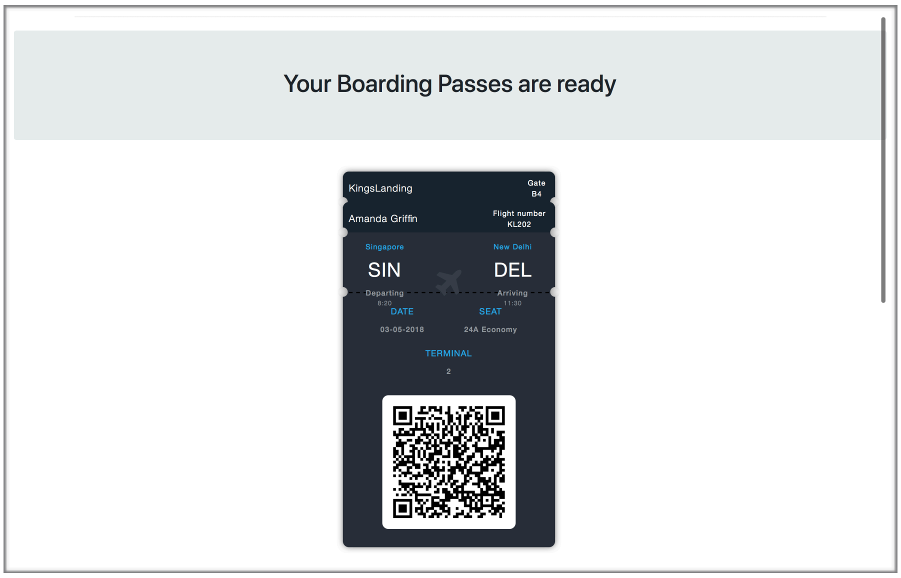
Seat Allocation Engine

The central module of this product is the Seat Allocation Engine, on top of which all other sister modules are built. The Seat Allocation Engine performs a constraint based satisfaction by allocating seats to the users with the guarantee that no two customers face a clash. It achieves this with a state of the art resource locking mechanism in accordance with time constraint which ensures atomicity and consistency. The central datastore is locked momentarily when a transaction is initiated preventing deadlocks and resource clashes.



Boarding Pass Generator

This module collects all of the user's travel details from the datastore, communicates with the airlines and issues a universal pass, along with a unique QR Code which has all the details encoded in it. This module then prepares the boarding pass and emails it to the user. It contains all information such as the Flight Number, Seat Number, ETA, Destination, Terminal Number, Gate Number and Luggage transfer information. This QR Code is generated at the server on the fly.


Your Boarding Passes are ready

KingsLanding
Gate
B4
Amanda Griffin
Flight number
KL202
Singapore
SIN
Departing
8:20
New Delhi
DEL
Arriving
11:30
DATE
03-05-2018
SEAT
24A Economy
TERMINAL
2

Flight Entry Module

This module will be handled by the employees at the airport. It scans the QR Code of a passenger's boarding pass, and decodes the information contained in it. This information is cross-checked with the central datastore and issues an alert if the passenger is about to board a wrong flight. It also gives the employees other details such as the number of passengers yet to board, the number of passengers already on board and urgent messages which needs to be conveyed to a passenger at the time of boarding.

Development

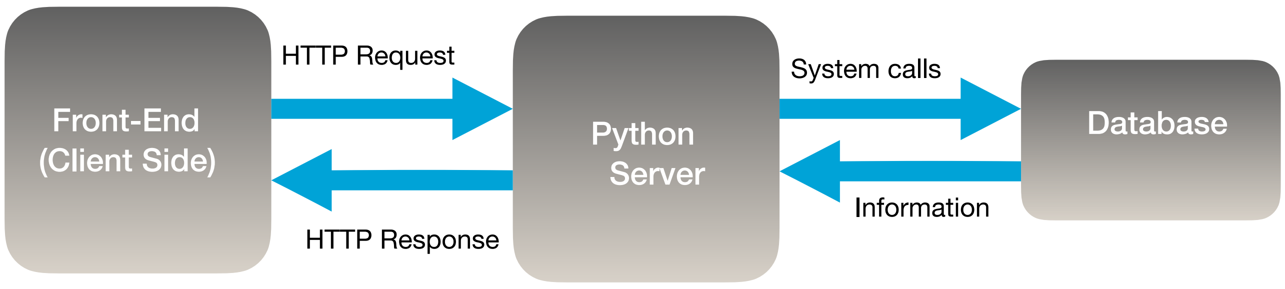
The entire system was developed using a team of five engineers - 2 developers, 1 tester, 1 datastore engineer and 1 front-end designer.

• **Sourav Johar**	**Lead Developer**
• **Preetham TK**	**Developer**
• **Abinanth C**	**Testing Engineer**
• **Madhuri Palanivelu**	**DataStore Engineer**
• **Mukesh V**	**Front-end designer**

The front-end of the system was written in HTML5, along with JavaScript and CSS. The responsive front-end design was achieved by using state-of-the art frameworks like BootStrap4 and JQuery.

The backend was written completely in Python along with the Flask framework. Using a general programming language along with a Web Framework like Flask ensured that the integration of all modules was neat and concise. This backend handles all routes and endpoints which are visitable from the client side. Care was taken to make sure that all the data which flows through the entire system was compatible at each tier.

The datastore was a combination of different file storing techniques, predominantly SQLite (relational database) and JSON (JavaScript Object Notation).


Front-End (Client Side)
HTTP Request
HTTP Response
Python Server
System calls
Information
Database