NAME: AIRLINE MANAGEMENT SYSTEM

# Features:

This particular case study is about managing flight and passengers by two types of people: admin and staff.

**Functionalities**:

1. Following features will be available for Airline staff while check-in
   1. Select flight (from list) based on current schedule (time)
   2. Display flight details
   3. Display passenger list with name, ancillary services, seat number
   4. Check-in passenger by select the respective seat
   5. Undo check-in by selecting the respective seat
   6. Display details of passenger like Name, Ancillary services
   7. Filter passengers by checked in / not, wheel chair, infant
   8. Change seat of passenger (through passenger list)
2. Following features will be available for Airline admin
   1. Dashboard with option to manage passengers, ancillary services per flight
   2. List passengers (name, ancillary services, seat number)
   3. Filter passengers by missing mandatory requirements (passport, address, date of birth)
   4. Add / Update passenger – Name, passport details, address
   5. Add / Update / Delete ancillary services, special meals, shopping items per flight

# Implementation:

Create an application **airline-management-system** using **create-react-app tool.** The application should contain following folders and files:

* All methods to interact with RESTful API application must be created in a service class that should be placed inside “services” folder. Service class must follow singleton pattern.
* All components must be present in the “components” folder and container components should be in separate “containers” folder
* Any images, local data (if any) should be placed inside “assets” folder
* Axios should be used to interact with RESTful API application
* Entire application should be Hook based. In case error handling, if Error Boundary pattern is used, then that component can be a Class component
* The application must be designed using Material UI framework
* Application must use RWD
  + Must be responsive for at least 3 breakpoints (Small, Medium, Large)
  + Preferable Usage of Flex Layout (CSS)
* For form handling Formik used and for validation it can be merged with “Joi” or “Yup” package
* The application must be PWA ready
* Should use “boostrap” CSS for styling if necessary
* Usage of Sass / Scss for styling
* ESLint must be configured for the application
* Minimum 3 components must be tested using “jest/enzyme”
* The application must be built by all the team members using Git
* Should use Jenkins for CI/CD
* The application must follow DDD
* The application must be tested using Lighthouse for the performance and PWA readiness (Lighthouse report >= 80 (SEO & Accessibility))