

Solution Documentation & Approach

- Assignment selected: *Assignment - A: Flight Fares Calendar*
- Time of Submission: *16th September 2019 - 4 AM IST Timings*

About

To display flights fares for selected month in calendar view

- The Live working version of the web app can be [found here](#)
- Alternate Link - <https://flight-fares-calendar.herokuapp.com/>
- Github Repo can be found here - [Github Link](#)

Description

Your task is to find the flight status for a month and display them in a calendar format. You can use Skyscanner Flight Search for fetching flight fares. The flight fares should be for 3 routes (one way). The routes are:

1. SIN – KUL
2. KUL – SIN
3. KUL – SFO

Tech Stack - Libraries & Frameworks used

The following Provides the Detailed List of Libraries, Tools, Frameworks & Platforms used in building the Application

Frontend

- *React JS (v 16.9.0)*
- *Typescript (v 3.4.3)*
- *React-Calendar*
- *TailWindCSS*
- *React-Notifications*
- *React-Icons-Kit*
- *Enzyme*
- *Create React App*
- *ESLint*

Backend

- *Node JS (v 10.x)*

- *ES6 (Babel)*
- *Express (4.16.4)*
- *Rapid API (Unirest SDK)*
- *Chai*
- *Chai-http*
- *Mocha*
- *Nodemon*

Tools & Platforms

- *Github*
- *VS Code Editor*
- *Heroku (heroku cli)*
- *Adobe XD*

Points to Consider

- Find Flight status for the month. (Consider startDate as the date when app loads)
- Use **Skyscanner Flight Search** for fetching flight fares.
- Flight Fares considered for the following 3 routes
 - *SIN - KUL*
 - *KUL - SIN*
 - *KUL - SFO*
- Following are the constants to consider
 - *Flight fares for one adult only*
 - *Show calendar for 1 month from the current date*
 - *Information to be provided in calendar format*
 - *Use Country as *US**
 - *Use Currency as *USD**
- Create a Rapid API account and fetch Rapid API key
- Test Driven Development
- Working Demo deployed on Heroku
- Github link to be provided
- [API Link](#)

Approach

1. UI References

Looked up online for Several UI References. The Following 2 references came close to the Usecase / Design Requirement

1. Skyscanner Web Application

Bengaluru (BLR) - Singapore Changi (SIN)

Cheapest month | 1
adult | Economy

Estimated lowest prices only. Found in the last 8 days.

[Calendar](#) [Chart](#)

☐ Direct flights only

Depart February 2020

Mon	Tue	Wed	Thu	Fri	Sat	Sun
27	28	29	30	31	1 ₹ 9,123	2 ₹ 10,977
3 ₹ 10,780	4 ₹ 11,525	5 ₹ 10,244	6 ₹ 7,284	7 ₹ 9,972	8 ₹ 12,070	9 ₹ 9,675
10 ₹ 9,675	11 ₹ 11,476	12 ₹ 9,403	13 ₹ 8,078	14 ₹ 9,972	15 ₹ 12,018	16 ₹ 9,675
17 ₹ 9,403	18 ₹ 12,680	19 ₹ 8,967	20 ₹ 7,286	21 ₹ 9,972	22 ₹ 12,070	23 ₹ 9,675
24 ₹ 9,403	25 ₹ 10,780	26 ₹ 9,403	27 ₹ 10,235	28 ₹ 8,949	29 ₹ 8,954	1
2	3	4	5	6	7	8

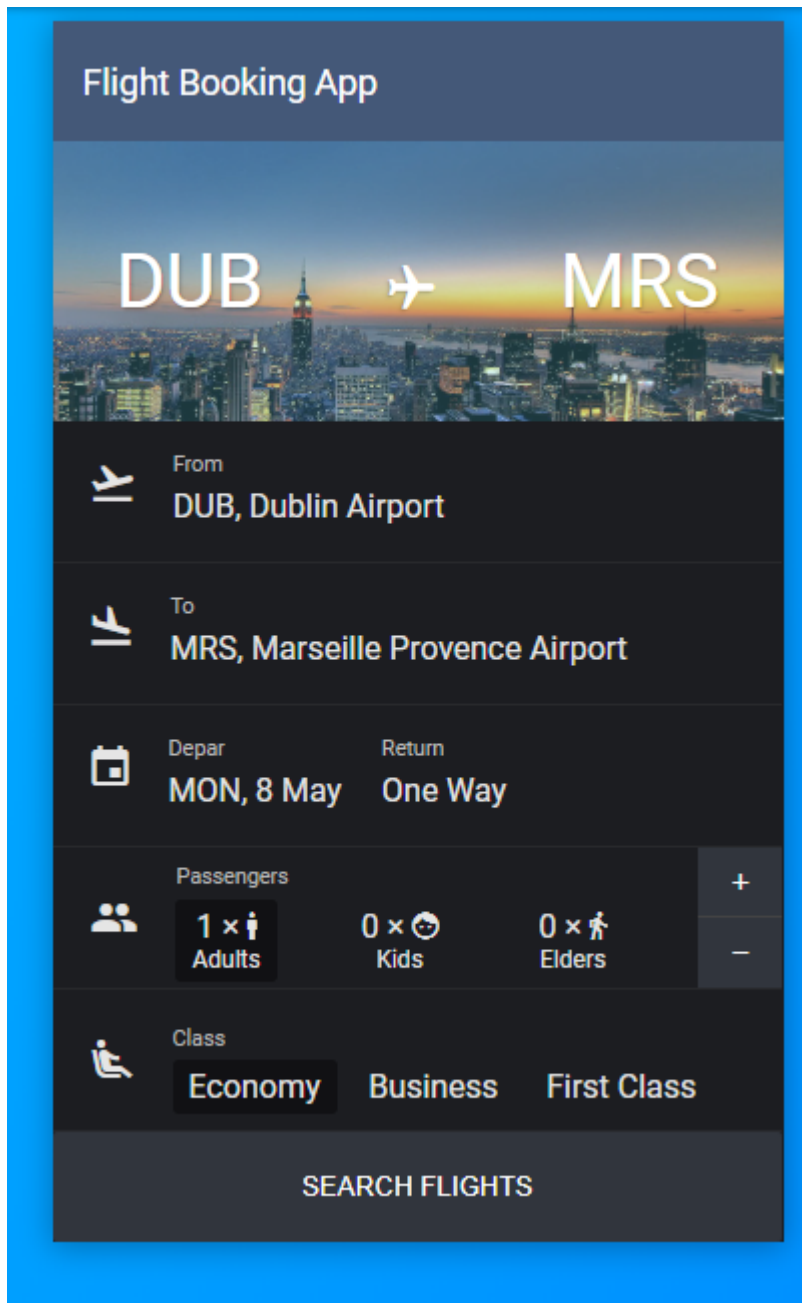
☐ Lowest - Highest ☐ Price needs to be checked ☐ 1+ stops only

Sat, 1 Feb

from ₹ 9,123
price per adult

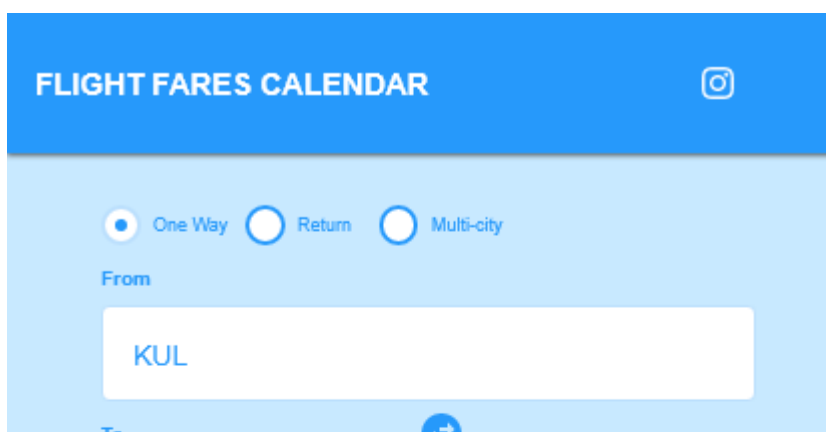
Show flights →

2. Flight Search Codepen Design



2. Design - Wireframes

- Using the References, Came up with a Low Fidelity Wireframes. Used **Adobe XD** to come up with the Wireframes
- Considered Mobile first approach to design, Aim was to ensure App is responsive and fits the mobile screens also



SIN

Depart

Cheapest Month

Cabin, Class & Travellers

1 Adult, Economy

« ‹ September 2019 › »

SUN	MON	TUE	WED	THU	FRI	SAT
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	1	2	3	4	5

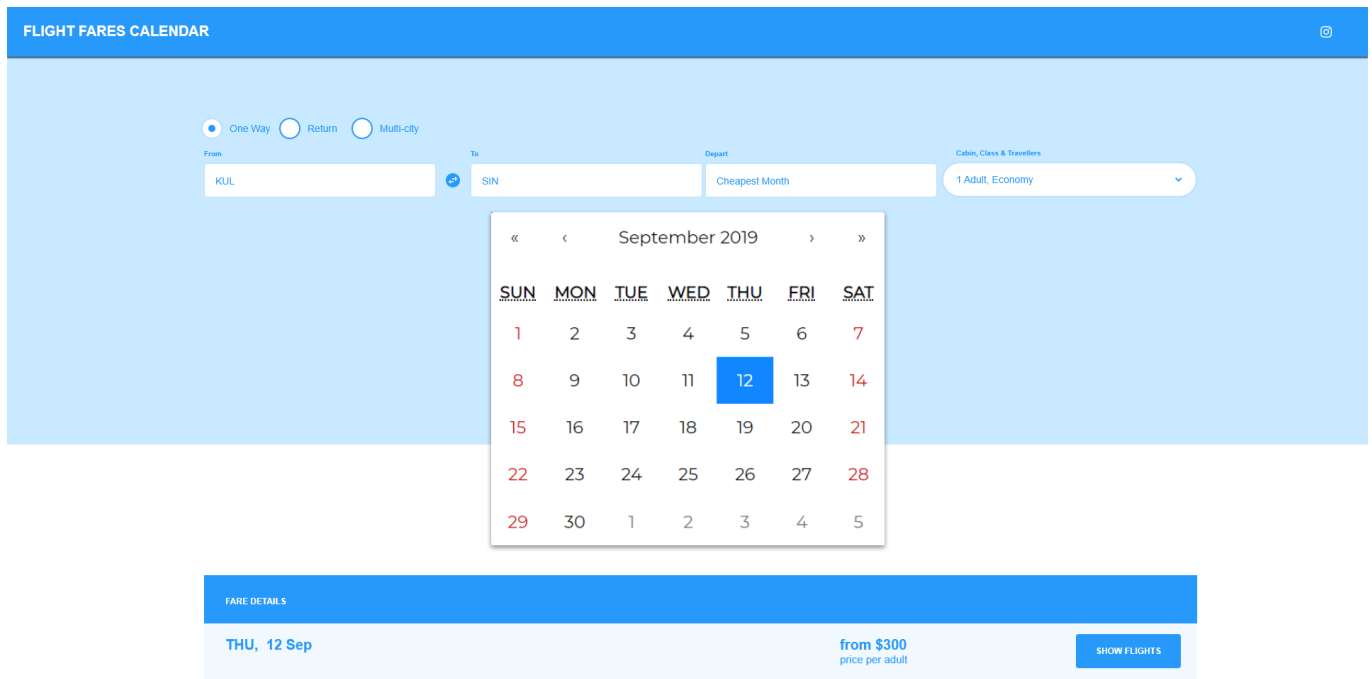
FARE DETAILS

THU, 12 Sep

from \$300
price per adult

SHOW FLIGHTS

- Designed Low Fidelity Wireframe to Large Desktop / Tablet Devices



The image shows a UI mockup for a 'FLIGHT FARES CALENDAR' application. At the top, there's a blue header with the title and a social media icon. Below the header, the main form area has a light blue background. It includes radio buttons for 'One Way' (selected), 'Return', and 'Multi-city'. There are input fields for 'From' (KUL), 'To' (SIN), and 'Depart' (Cheapest Month). A dropdown menu for 'Cabin, Class & Travellers' shows '1 Adult, Economy'. A calendar for September 2019 is displayed, with the 12th of September highlighted in blue. Below the calendar, a 'FARE DETAILS' section shows 'THU, 12 Sep' and a price of 'from \$300 price per adult', with a 'SHOW FLIGHTS' button.

FLIGHT FARES CALENDAR

One Way Return Multi-city

From: KUL To: SIN Depart: Cheapest Month Cabin, Class & Travellers: 1 Adult, Economy

September 2019

SUN	MON	TUE	WED	THU	FRI	SAT
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	1	2	3	4	5

FARE DETAILS

THU, 12 Sep from \$300 price per adult SHOW FLIGHTS


3. Tech Stack to consider

Once the wireframe was designed, It was clear to select which libraries / frameworks to work with. The following were considered.

- *React JS + Typescript + TailWind CSS + Axios*
- *Node (Express) + ES6 + Babel*
- *Github for Versioning, CI/CD pipeline*
- *Heroku for Cloud Deployment*

4. Design Application

The Application was designed with a **Mobile First Approach in mind**. The Aim was to make *the Calendar accessible and readable even on handheld devices with small width*.


FLIGHT FARES CALENDAR

About

FILL IN THE BELOW FLIGHT DETAILS

☒ One Way
 ☐ Return
 ☐ Multi-city

FROM

(SIN) Singapore

↑↓

TO

(KUL) Kuala Lumpur

SELECT MONTH


September 2019

CABIN, CLASS & TRAVELLERS

1 Adult, Economy

Find Fares

"Fill Form Details to display Fares for the month in Calendar view"


FLIGHT FARES CALENDAR

About

2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17 \$ 48	18 \$ 48	19 \$ 48	20 \$ 68	21 \$ 48	22 \$ 46
23 \$ 47	24 \$ 49	25 \$ 47	26 \$ 47	27 \$ 60	28 \$ 48	29 \$ 43
30 \$ 43						

FARE DETAILS

WED, 18 September
\$ 48 (price per adult)

SHOW FLIGHTS

This would ensure responsiveness even for large screens as the devices start getting larger

FLIGHT FARES CALENDAR

One Way Return Multi-city

FROM (SFO) San Fransico TO (KUL) Kuala Lampu

SELECT MONTH 1 Adult, Economy

CABIN, CLASS & TRAVELLERS 1 Adult, Economy

Find Fares

September 2019

MON	TUE	WED	THU	FRI	SAT	SUN
						1 \$ 100
2 \$ 300	3 \$ 400	4 \$ 450	5 \$ 500	6 \$ 600	7 \$ 650	8 \$ 100
9 \$ 300	10 \$ 400	11 \$ 450	12 \$ 500	13 \$ 600	14 \$ 650	15 \$ 100
16 \$ 300	17 \$ 400	18 \$ 450	19 \$ 500	20 \$ 600	21 \$ 650	22 \$ 100
23 \$ 300	24 \$ 400	25 \$ 450	26 \$ 500	27 \$ 600	28 \$ 650	29 \$ 100
30 \$ 300						

FARE DETAILS

THU, 14 Sep \$ 650 (price per adult) SHOW FLIGHTS

Design Patterns

My Aim in any given application development / prototype is to always limit the excess use of any libraries for just one / two tasks and try to leverage the selected library / platform to its full features offered.

a. Backend

1. MVC Design Pattern

For the Backend, Express provides out of the box routing feature, which is great for building **MVC - Model View(Service) Controller** where the View is actually the main routing that appens within the application.

2. Babel + Test Driven Development

Wrote custom Node scripts with a **Test Driven Development** approach in mind. Also Babel ensures that the final applciation is all available, bundled under `build` folder.

NOTE: If the Test fails, The Build will not get generated

b. Frontend

1. React (Hooks) + Typescript

For the Frontend, Typescript + React version 16.x which offers the new feature - Hooks. The following React Design Patterns were implemented:

2. Compound Patterns

With respect to React components, this could mean a component that is composed of two or more separate components. Any React component can be composed of 2 or more separate components. So, that's really not a good way to describe compound components. The main component is usually called the parent, and the separate composed components, children.

- For Example, in the current application:
- **Parent** : FareDetailsForm Component
- **Child** : OriginDestinationSelect Component
- **Child** : Select Component

3. Use of Hooks + Context API

the Application showcases use of both Context & Hooks. **All Components are mainly Functional Components** in this application. This provides an ease when it comes to testing also makes the application modular and reusable.

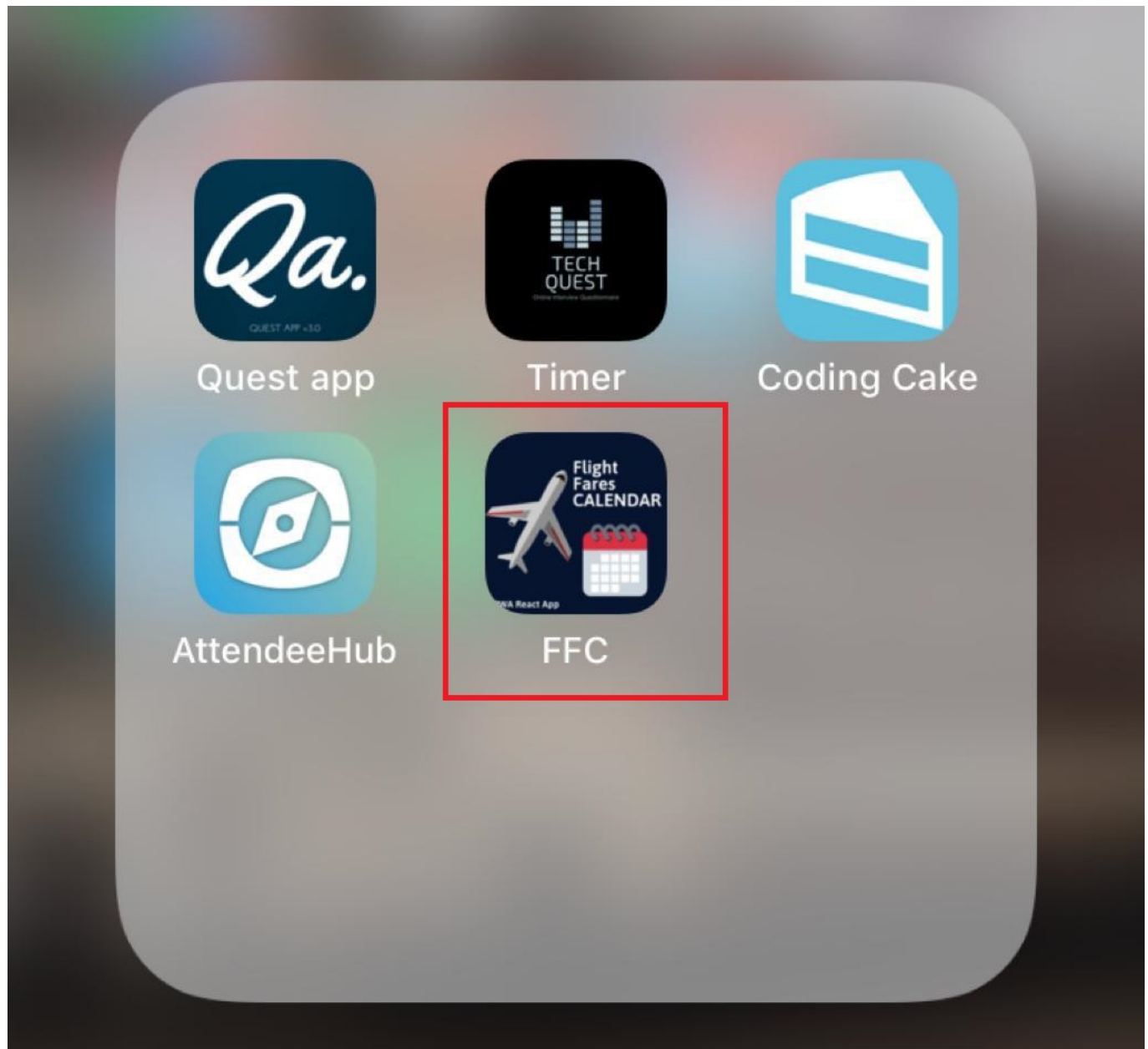
- useState
- useContext
- useRef

4. Leveraging Typescript

Even though most of the components, are Functional components, using Typescript we can still gain the upper ground of ensuring proper Props being passed to each component, and strict datatyping.

5. PWA

In order to provide a consistent feel across Devices (both desktop & mobile) - Leveraged PWA so that the application can be saved on the Iphone / Android mobile as a Native Icon



Color Palette

- #0086FF
 - #00B4FF
 - #47CF73
 - #445878
 - #9ca0b1
-