Sabre

VIRTUAL Developer Bench

Sabre Red 360 SDK: Understanding Feature Fundamentals

The Developer Experience Team

2021

1 December 2021

On the Call

Alexandre Meneghello
Software Engineering

Avery Perkins
Product Marketing

Alex Xavier
Product Marketing

Red App ecosystem overview

Sabre Red 360 is responsible for an entire ecosystem involving Travel technology producers and consumers, through its innovative Red App Marketplace and Developer network.

Red App Certified Developer
Private Agency Developer
Developer Partner

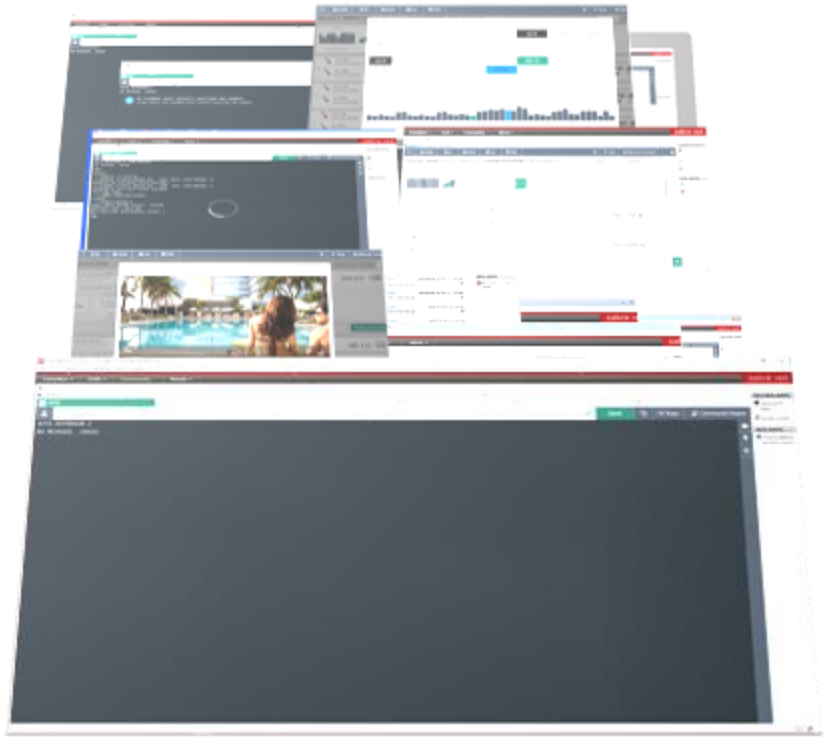
Red App Developers

Development Environment

Sabre Dev Studio

Apply → Build → Certify → Publish → Market

from Idea to Published Product



Travel Agencies
Travel Management companies
Leisure operators

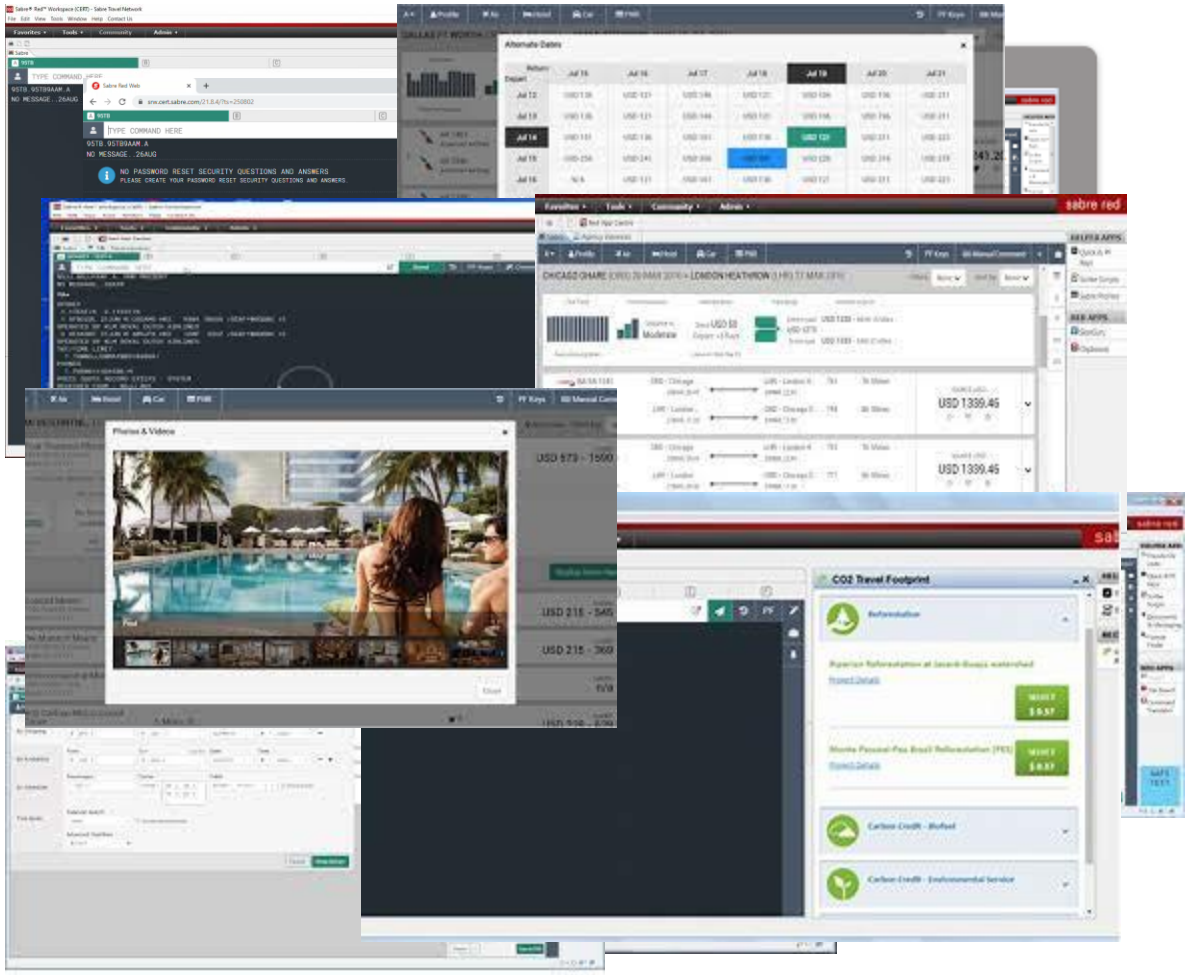
sabre red app centre

Technology Providers

Sabre Red 360

Technology Consumer

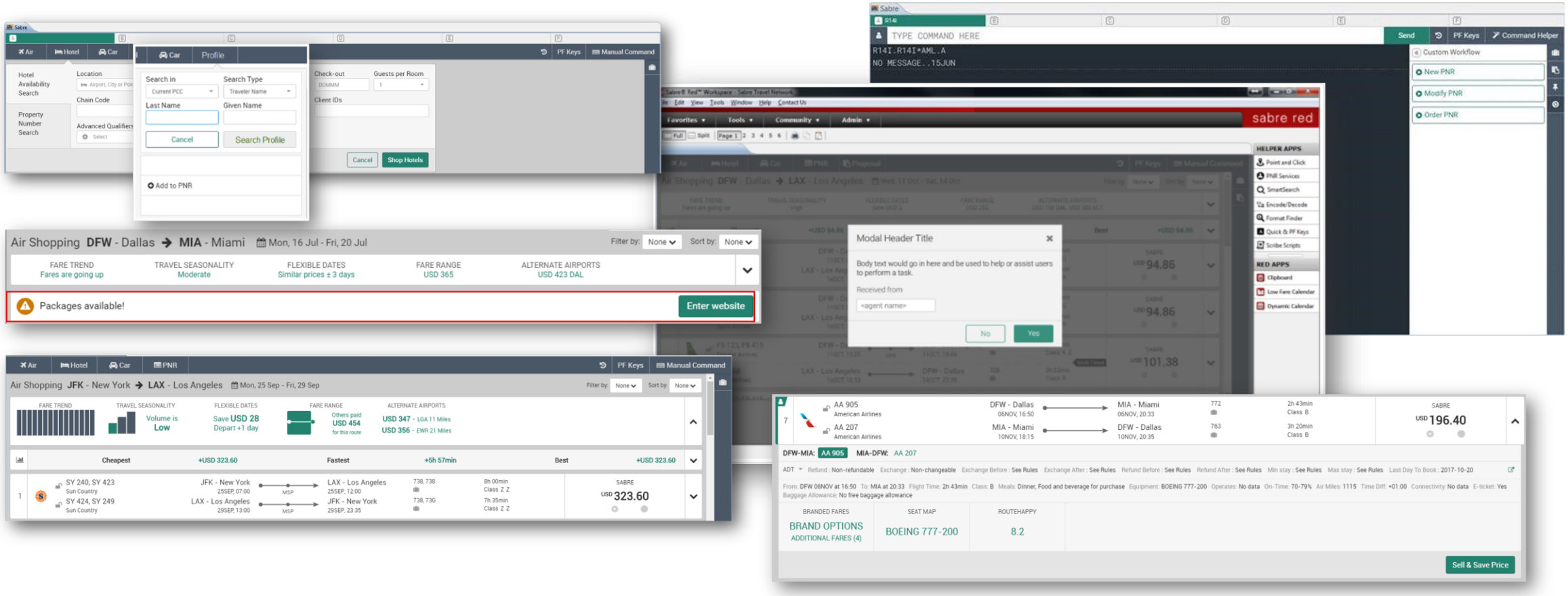
Sabre Red 360 & customization



- Sabre Red 360 is an end-to-end sales workspace that sells billions in Travel inventory each year. Could be extended by plugging-in modules, or Red Apps.
- Sabre Red 360 main View, offers advanced visualization capabilities and orchestration to the Travel consultant workflow.
  - Workflow, from Sabre Red 360 core, any business operation involved on **SHOP, SELL, BOOK, TICKET** and **MANAGE RESERVATIONS** of FLIGHT, HOTEL, CAR, CRUISE AND RAIL
- Sabre Red 360 Desktop App, it is a multi-windowed workspace application, which hosts Sabre's main view, complimentary user workflows and allows integrations with back-office and external systems.
- Sabre Red Web, Mobile design brings (big) part of sales workspace to Travel Consultant's pocket, or an internet café anywhere.

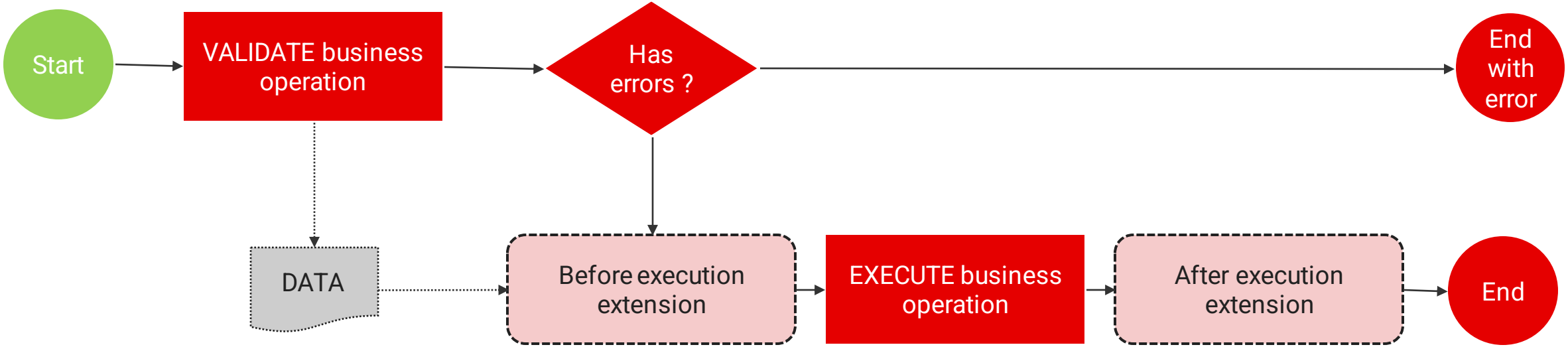
Contribute

Create Sabre Red native experience, create custom forms and widgets to communicate effectively with users.



Automate

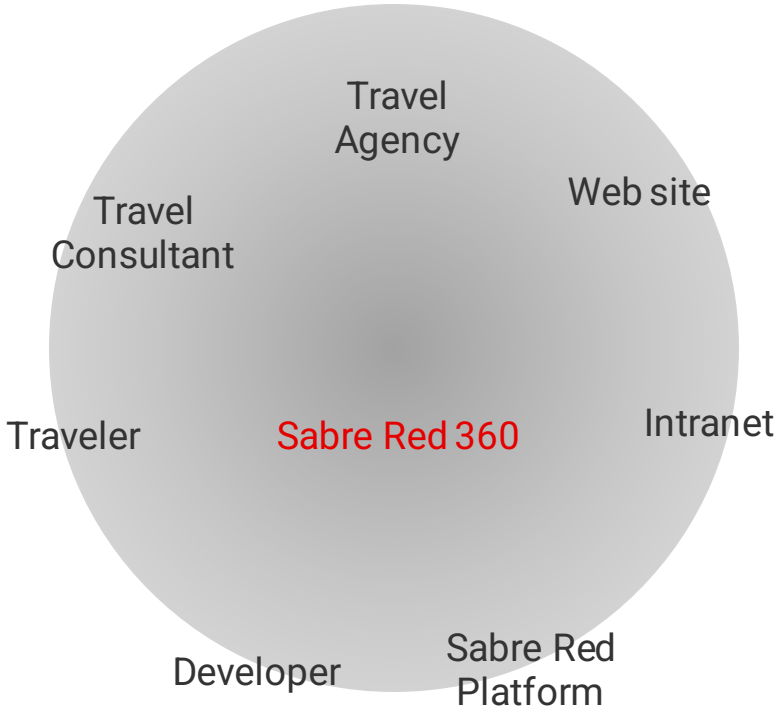
Customize pre-defined workflows, reduce or augment standard workflows seamlessly



SHOP, PRICE	BOOK	SELL	TICKET	MANAGE
Air, Hotel Exchange Hotel Details	Air, Hotel	Air, Hotel, Car Passive Add	Air	End PNR Modify PNR Queue Place

Integrate

Integrate adjacent systems, direct access to external workflows, content sources and APIs



Integration opportunities:

- ✓ Connectivity with external systems
- ✓ Back-office integration
- ✓ API client
- ✓ Database integration
- ✓ Customer profiles
- ✓ Intranet / Extranet
- ✓ integrate existing web sites to the SR360 workspace

Red App SDK, fundamental building blocks

Red App is a software package, created with Sabre Developer Tools, which bundles different extensions to customize Sabre Red 360 operations during runtime.

Contributions to the User Interface, allows Red Apps to customize the Travel consultant workspace (Eclipse) or the primary Sabre Red 360 UI (NGV)

Workflow extensions, allow access to data and controlled execution of Business operations done through Sabre Red 360 main View

Communication Foundations, allows Red Apps to access Sabre Red Platform Services exposed through SR 360 runtime and abroad

Tools and Utilities, general utility layer, allows to deal with user settings and authentication, as well integration with external systems and process automation.

UI Widgets	Workflow	Web Browser	Workspace
Pop-up Forms	Workflow panel button	CrossWindow	View / Editor window
Tile Widgets	Nudge	WebKit View / Editor	Red App Sidebar
Command button + Popover	Add to PNR form	Javascript API	Workspace Menus
React bootstrap	Refresh PNR		Notifications
	Interstitial		Modal Dialog

SHOP, PRICE	BOOK	SELL	TICKET	MANAGE
Air, Hotel	Air, Hotel	Air, Hotel, Car	Air	End PNR
Exchange		Passive Add		Modify PNR
Hotel Details				Queue Place

Sabre APIs	Sabre Commands	Web	Active Listening	Security
REST and SOAP API calls	Execute Sabre Commands	HTTP Request	EMU_RESPONSE Event	SSO Assertions
GetReservation	LLS Calls - HOST		EMU_COMMAND Service	

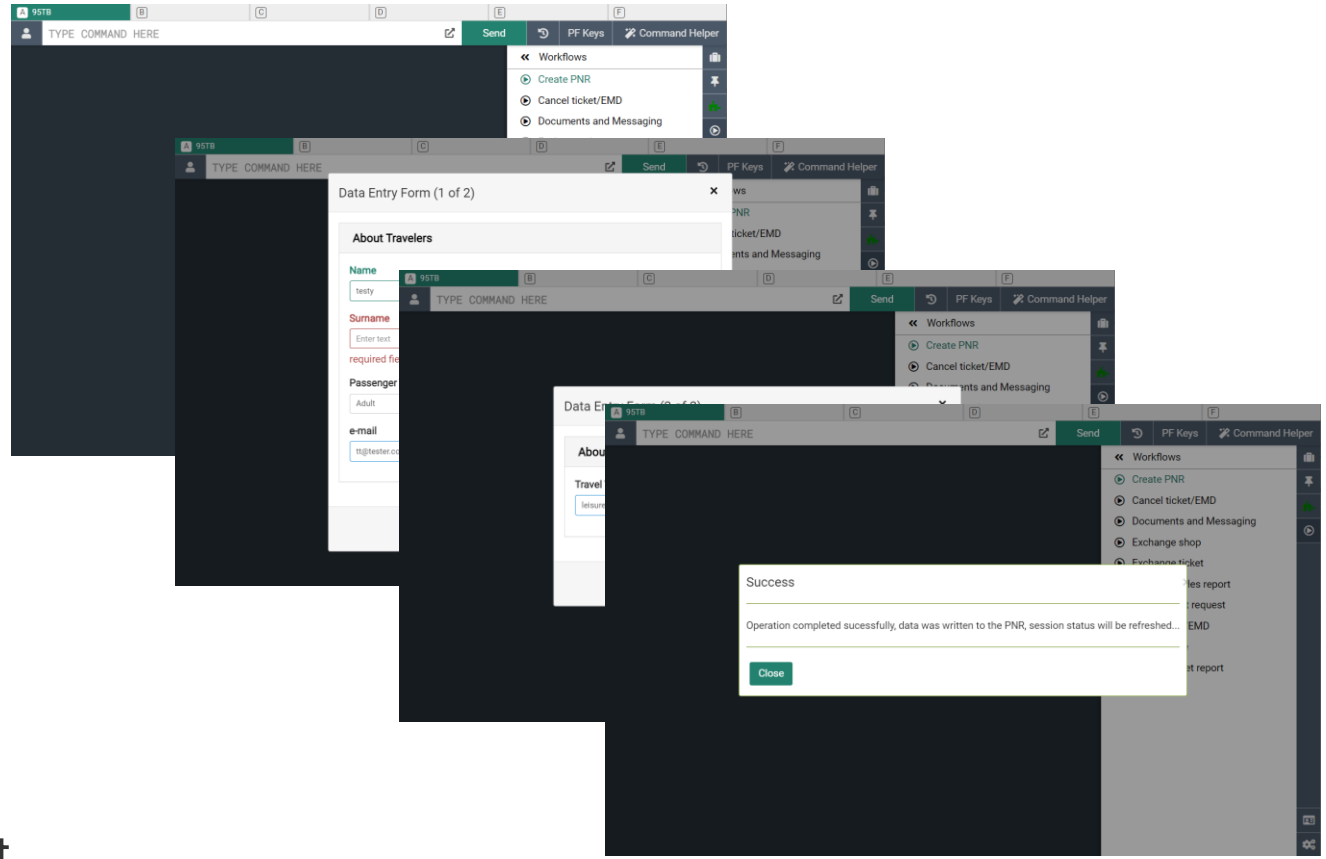
Agency	Support	Workspace
Agent Profile	Log services	Plugin resources
Config Services	First run	Internationalization

RED APP EXAMPLE

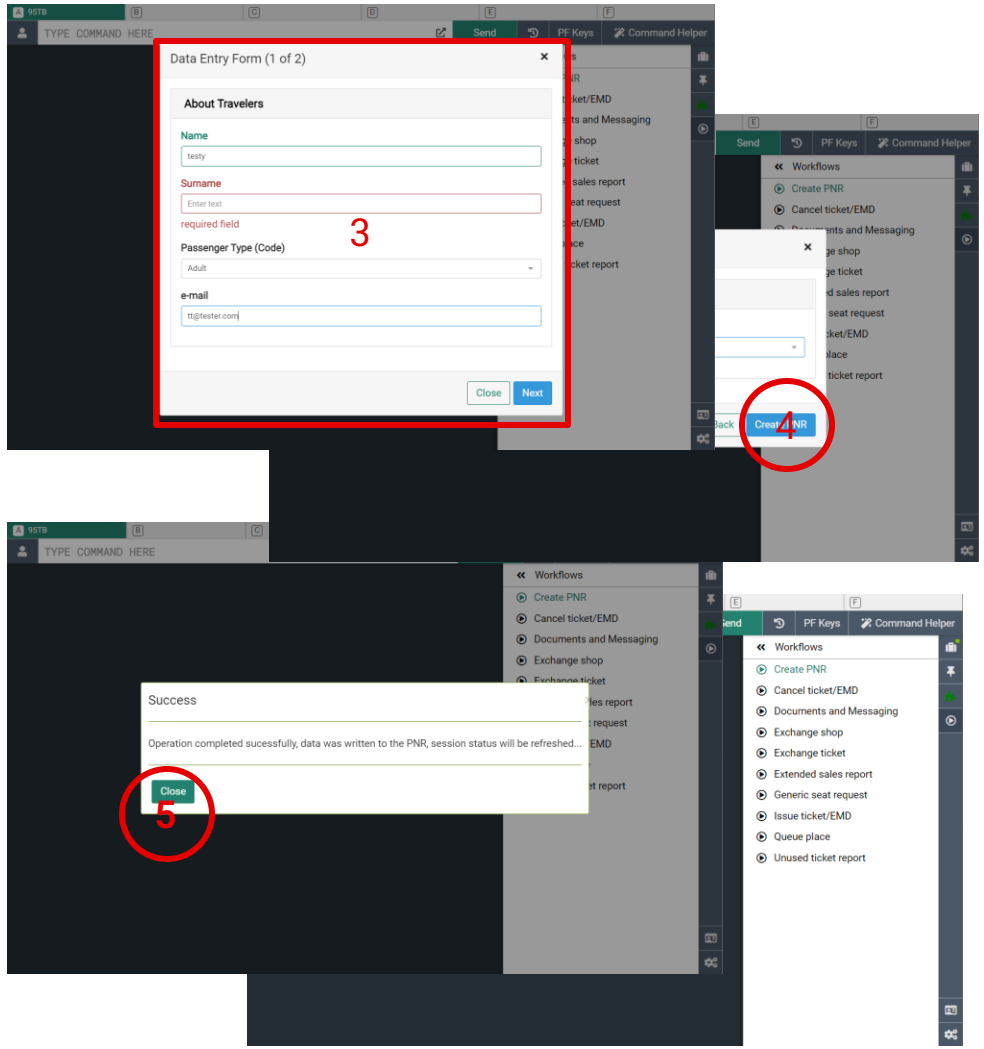
Let´s create a Red App to demonstrate common use cases and building blocks provided by the Red App SDK.

Red App project – PNR Shell

- The main use case for this Red App would be to prepare a new Reservation record, gathering details about the trip and travelers it will help personalize content during travel consultant workflow.
- User will start workflow by clicking Create PNR button on the Sidebar panel.
- A multi-stage form will be presented to capture information about passenger and travel details.
- Once submitted, the information gathered will be written to the current PNR on the session.

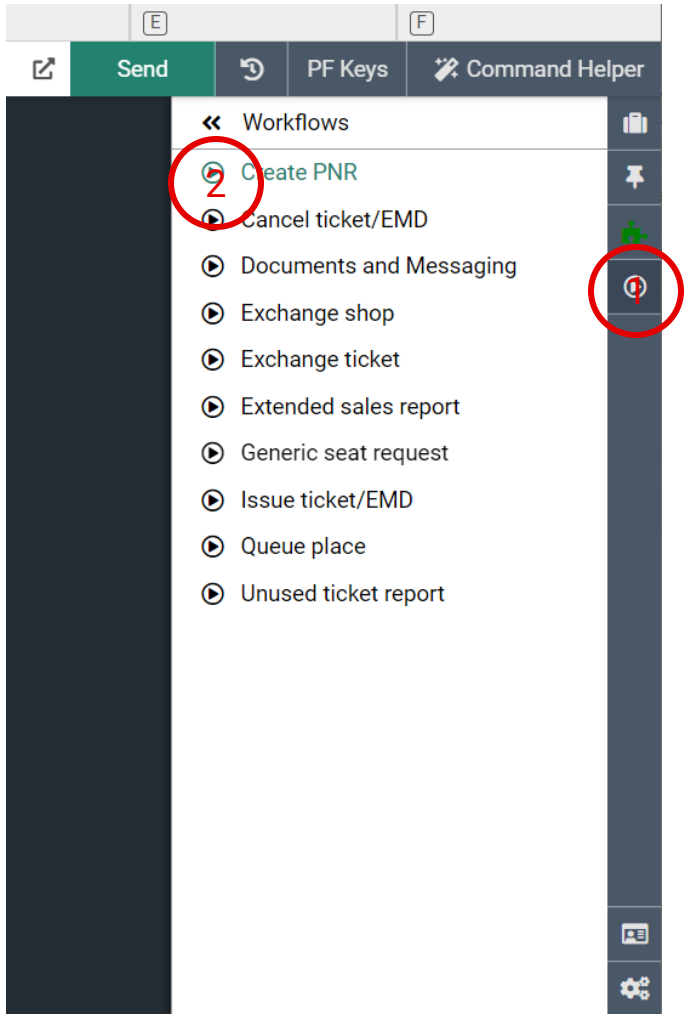


Implementation details



SDK features used :

1. Contribution to custom workflow panel
2. Sidebar button
3. Custom pop-up Dialog provided by ***Layer Service*** and react-bootstrap component library
4. Upon form submission Red App utilizes ***SoapAPI Service*** and send an XML request to ***UpdateReservationRQ*** API
5. After writing data to PNR, Red App user ***PnrPublic Service*** to refresh the front-end session

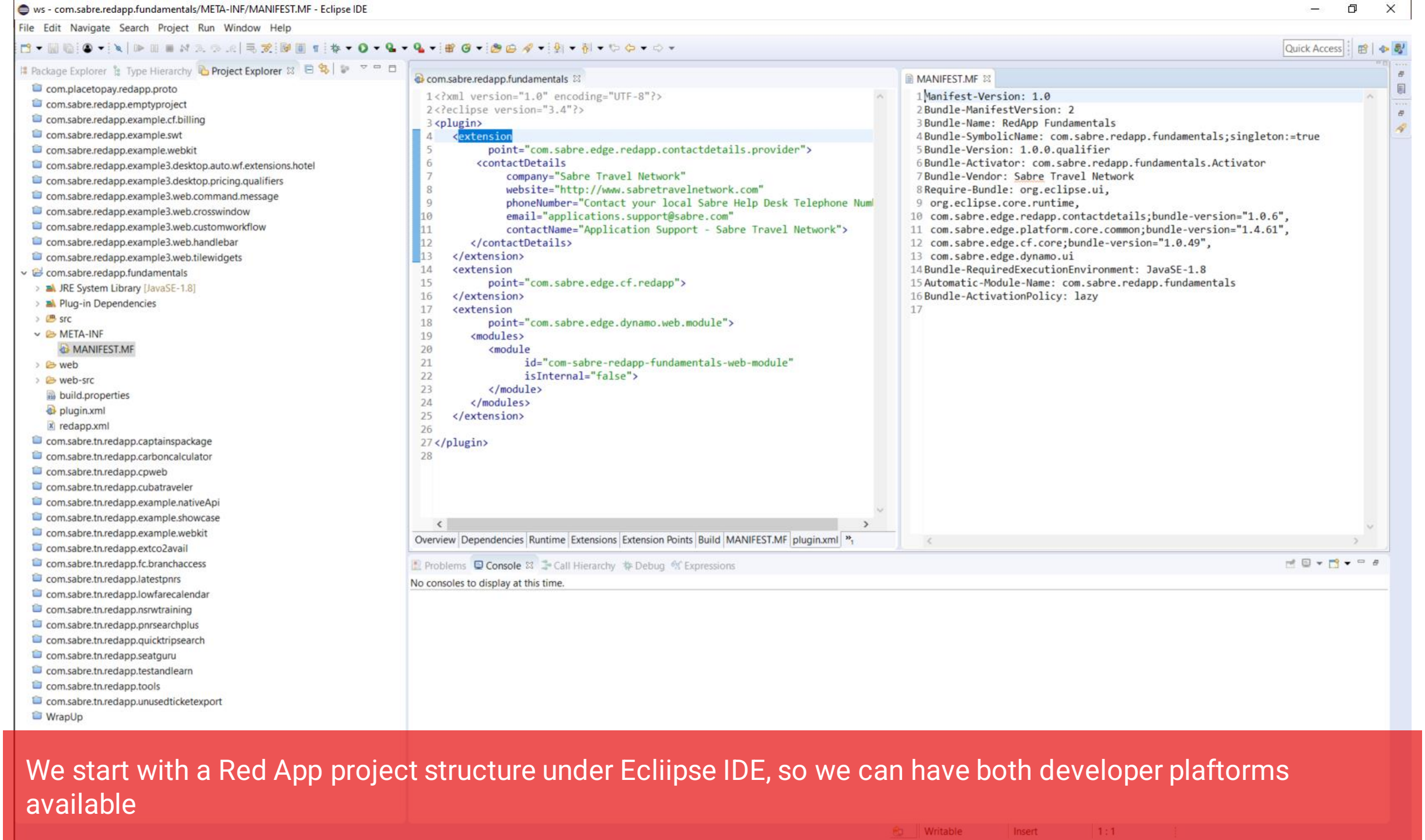


DEVELOPMENT SLIDESHOW

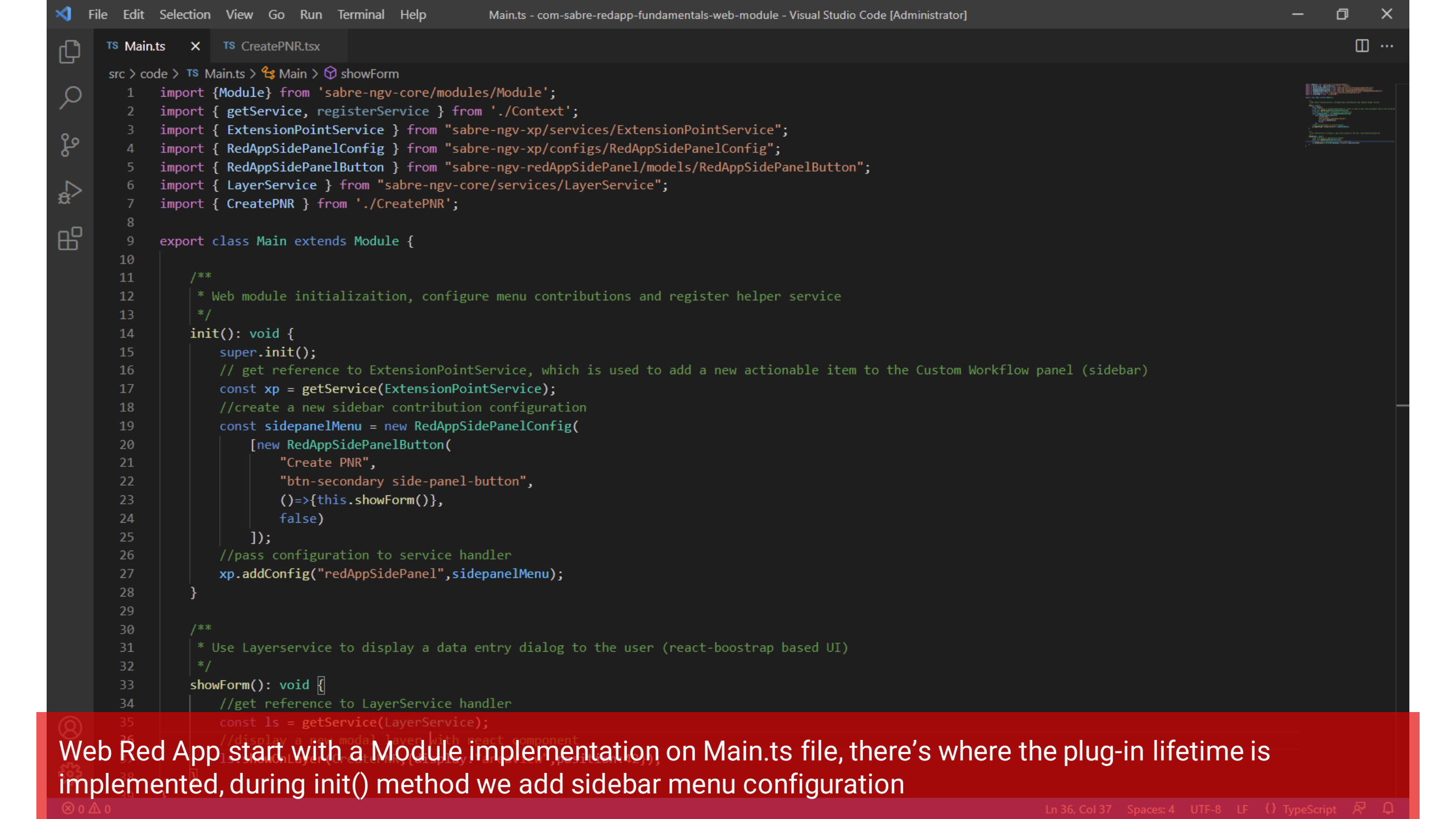
Walk through implementation of a Red App that presents a form to the user, collect some date and consume Sabre API service to write information to the reservation. (PNR)

*** The source code for this project could be found at:

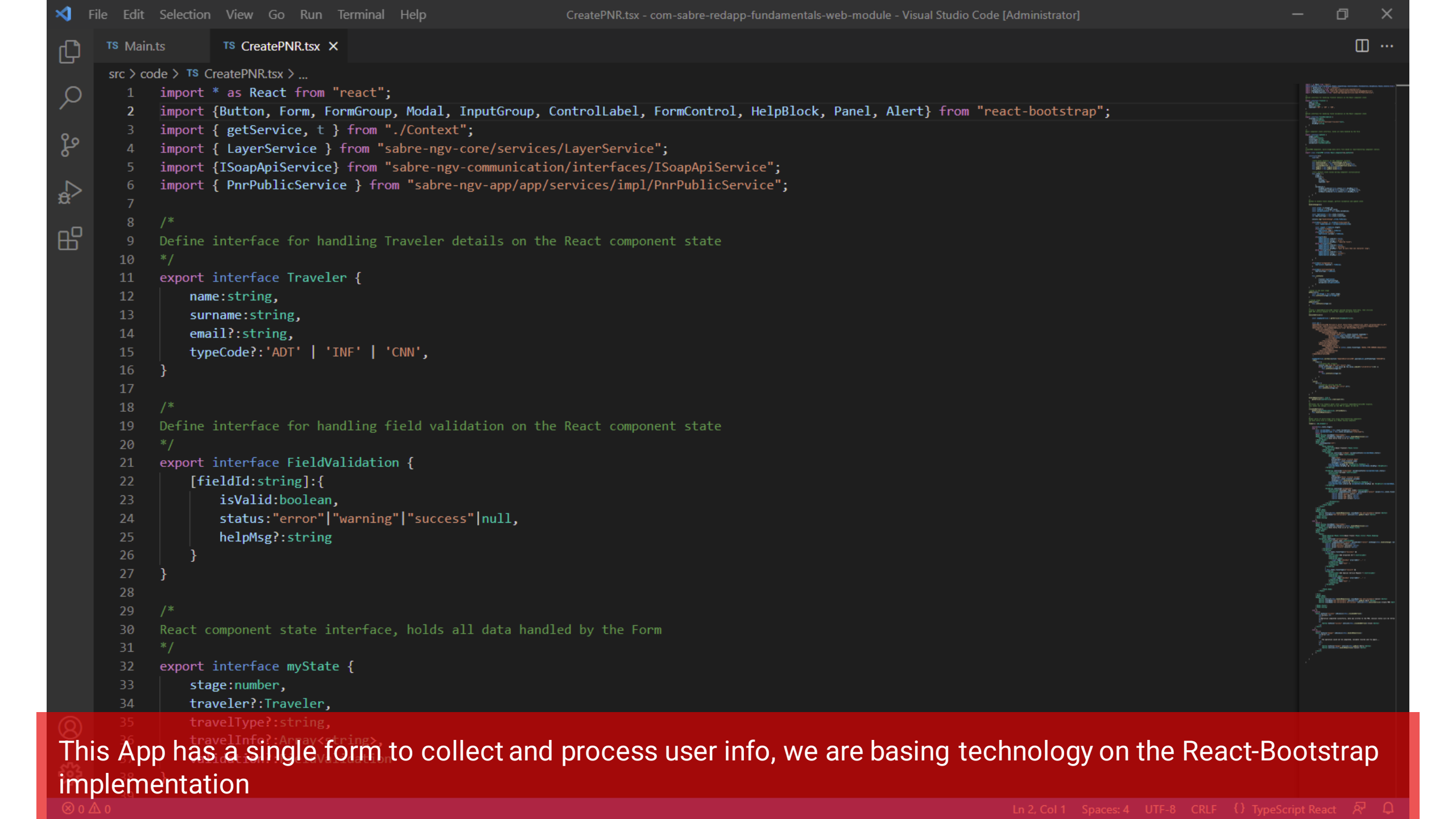
GitHub - SabreDevStudio/RedApp-Fundamentals: Sabre sample Web Red App showing SDK fundamentals



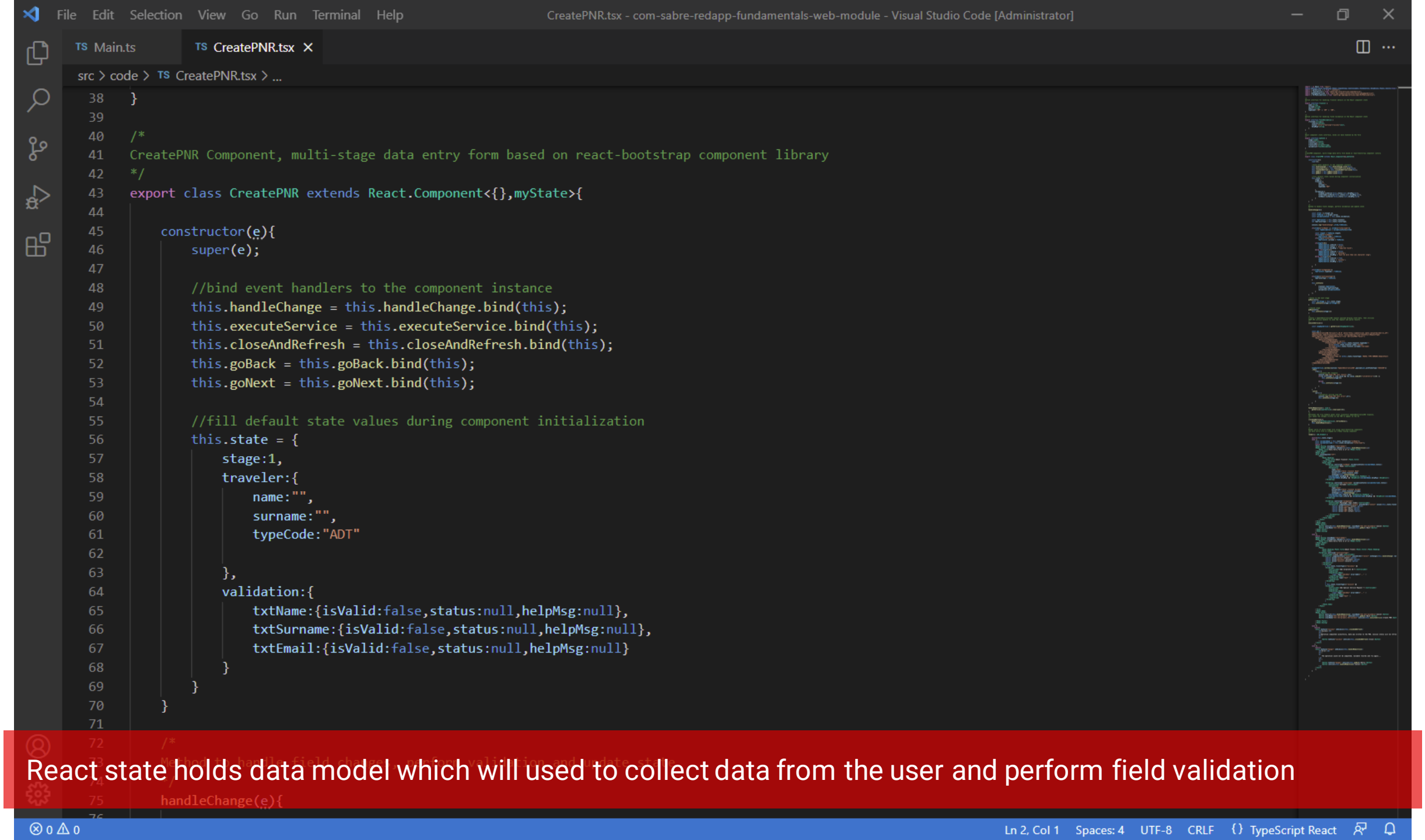
We start with a Red App project structure under Ecliipse IDE, so we can have both developer plaftorms available

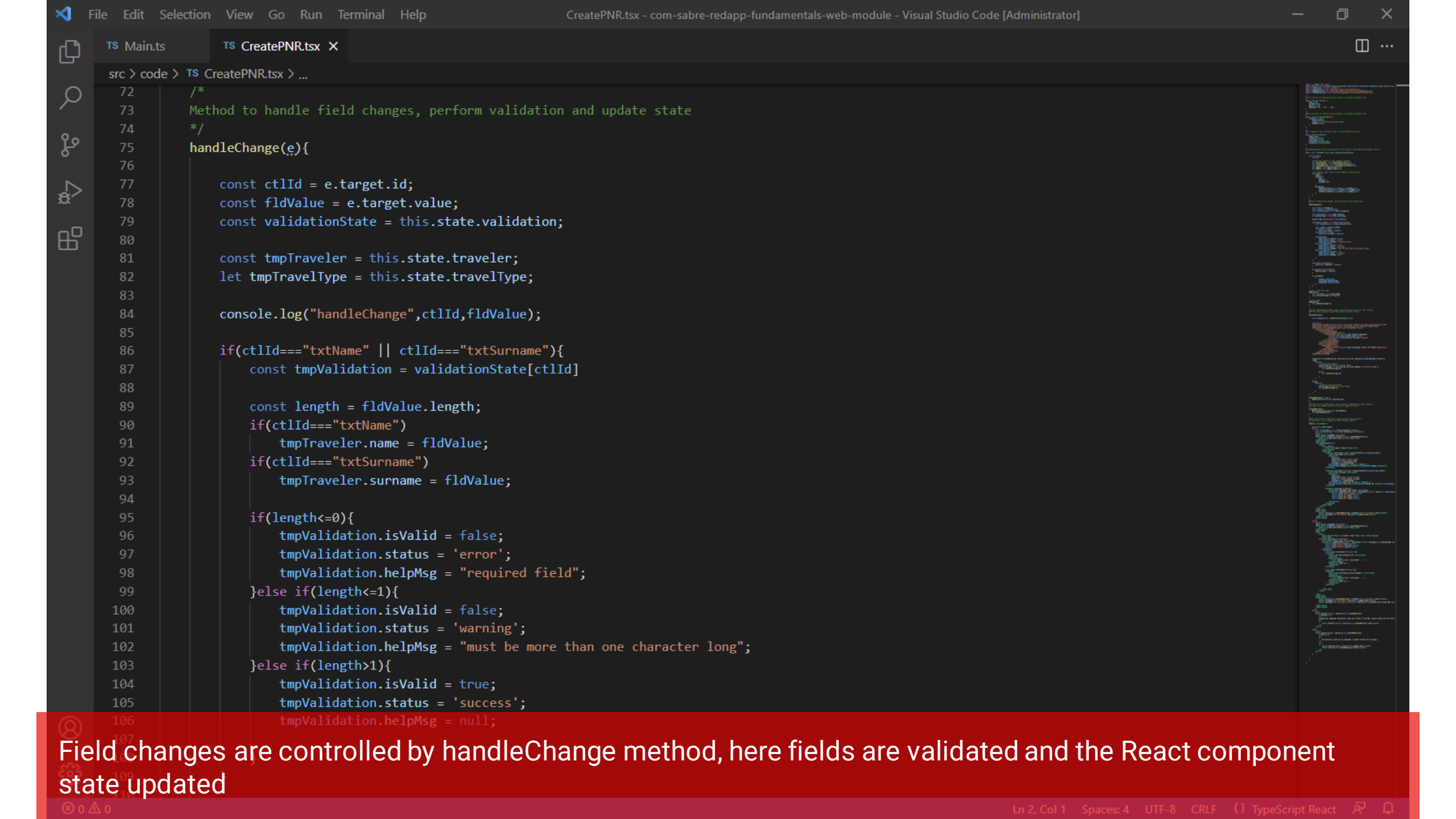


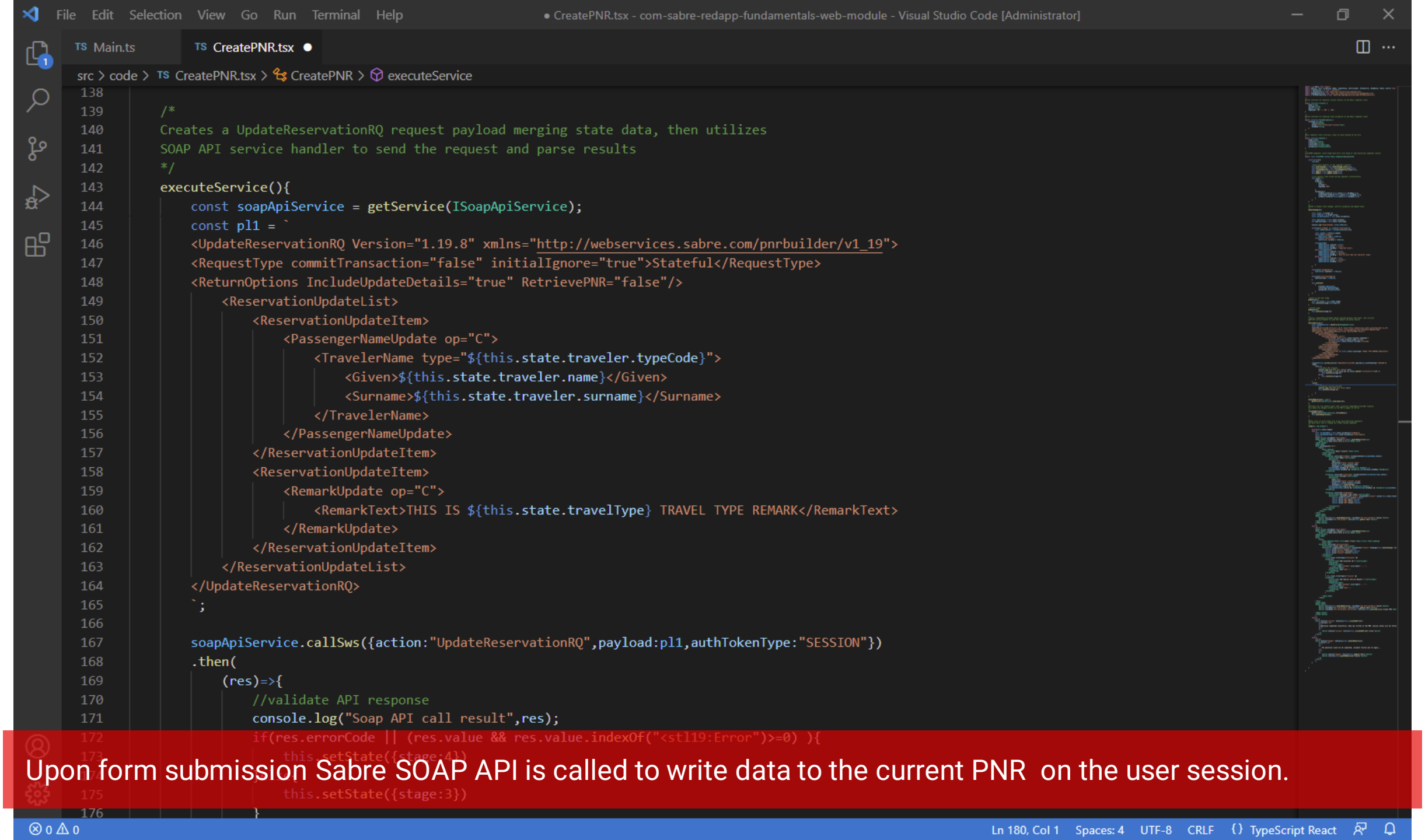
Web Red App start with a Module implementation on Main.ts file, there's where the plug-in lifetime is implemented, during init() method we add sidebar menu configuration

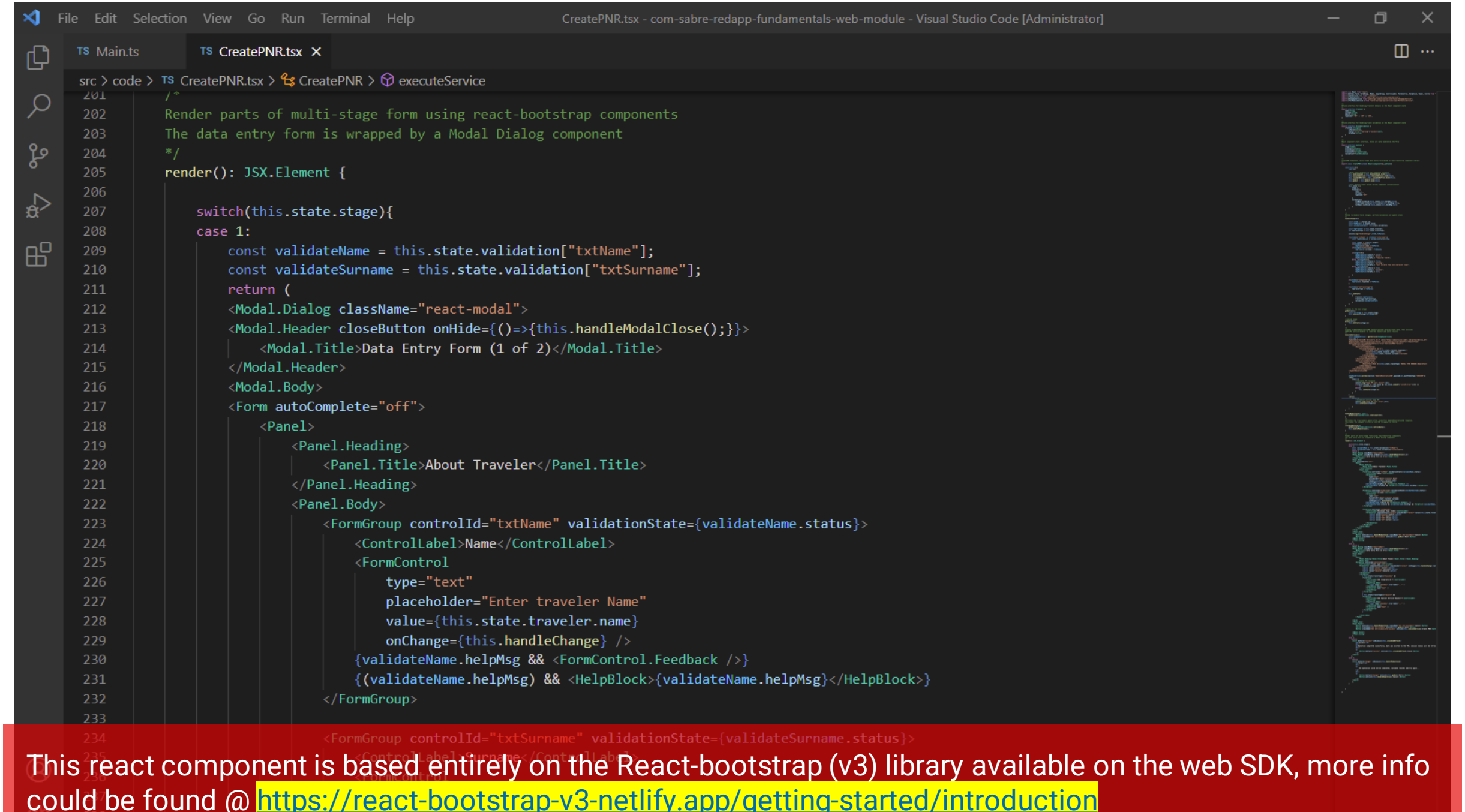


This App has a single form to collect and process user info, we are basing technology on the React-Bootstrap implementation

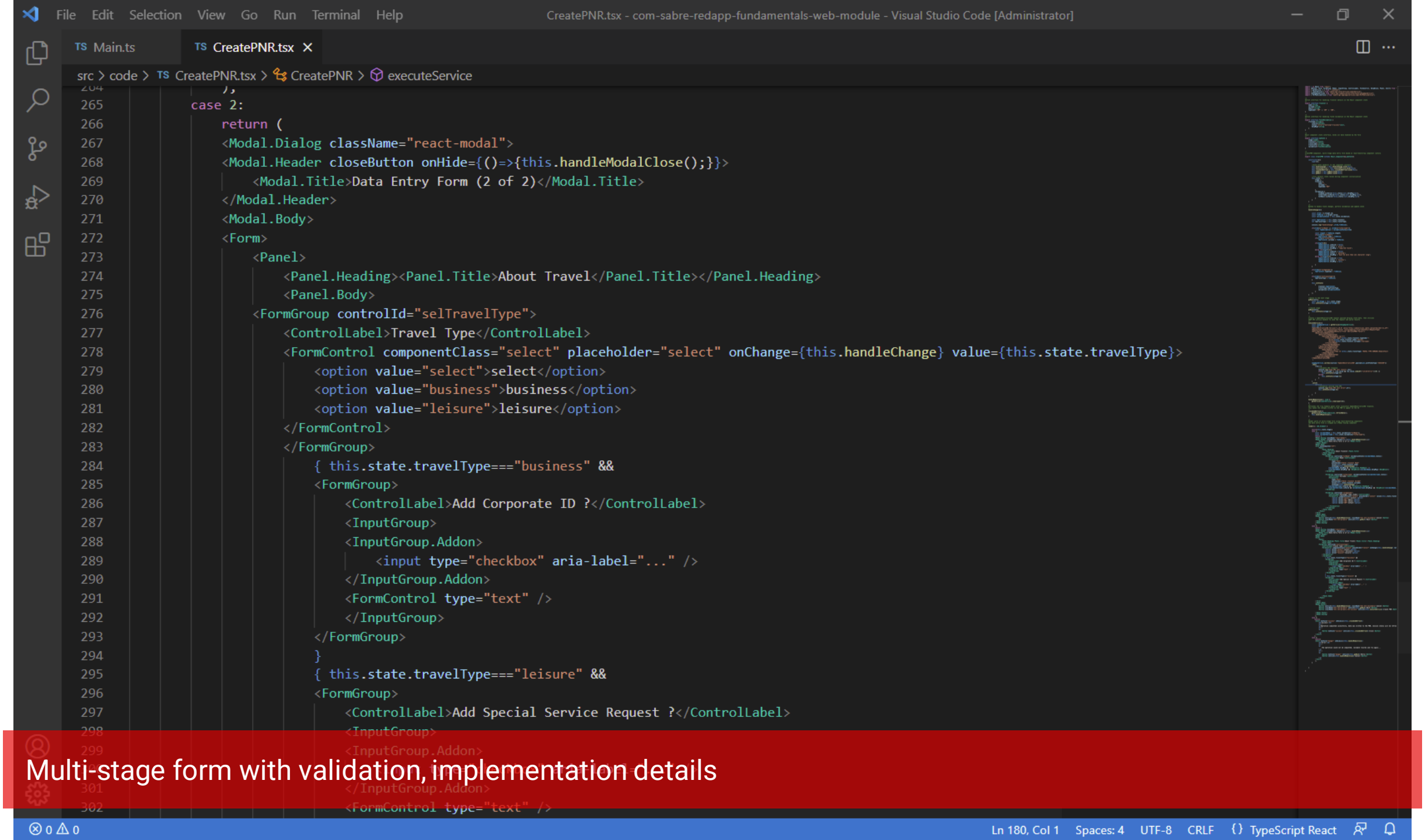

File Edit Selection View Go Run Terminal Help
CreatePNR.tsx - com-sabre-redapp-fundamentals-web-module - Visual Studio Code [Administrator]
Main.ts
CreatePNR.tsx
src > code > CreatePNR.tsx > ...
38 }
39
40 /*
41 CreatePNR Component, multi-stage data entry form based on react-bootstrap component library
42 */
43 export class CreatePNR extends React.Component<{},myState>{
44
45 constructor(e){
46 super(e);
47
48 //bind event handlers to the component instance
49 this.handleChange = this.handleChange.bind(this);
50 this.executeService = this.executeService.bind(this);
51 this.closeAndRefresh = this.closeAndRefresh.bind(this);
52 this.goBack = this.goBack.bind(this);
53 this.goNext = this.goNext.bind(this);
54
55 //fill default state values during component initialization
56 this.state = {
57 stage:1,
58 traveler:{
59 name:"",
60 surname:"",
61 typeCode:"ADT"
62
63 },
64 validation:{
65 txtName:{isValid:false,status:null,helpMsg:null},
66 txtSurname:{isValid:false,status:null,helpMsg:null},
67 txtEmail:{isValid:false,status:null,helpMsg:null}
68 }
69 }
70 }
71
72 /*
75 handleChange(e){
React state holds data model which will used to collect data from the user and perform field validation
0 0
Ln 2, Col 1 Spaces: 4 UTF-8 CRLF TypeScript React


File Edit Selection View Go Run Terminal Help
CreatePNR.tsx - com-sabre-redapp-fundamentals-web-module - Visual Studio Code [Administrator]
Main.ts
CreatePNR.tsx
src > code > CreatePNR.tsx > ...
/*
Method to handle field changes, perform validation and update state
*/
handleChange(e){
const ctlId = e.target.id;
const fldValue = e.target.value;
const validationState = this.state.validation;
const tmpTraveler = this.state.traveler;
let tmpTravelType = this.state.travelType;
console.log("handleChange",ctlId,fldValue);
if(ctlId==="txtName" || ctlId==="txtSurname"){
const tmpValidation = validationState[ctlId]
const length = fldValue.length;
if(ctlId==="txtName")
tmpTraveler.name = fldValue;
if(ctlId==="txtSurname")
tmpTraveler.surname = fldValue;
if(length<=0){
tmpValidation.isValid = false;
tmpValidation.status = 'error';
tmpValidation.helpMsg = "required field";
}else if(length<=1){
tmpValidation.isValid = false;
tmpValidation.status = 'warning';
tmpValidation.helpMsg = "must be more than one character long";
}else if(length>1){
tmpValidation.isValid = true;
tmpValidation.status = 'success';
tmpValidation.helpMsg = null;
Field changes are controlled by handleChange method, here fields are validated and the React component state updated
Ln 2, Col 1
Spaces: 4
UTF-8
CRLF
TypeScript React

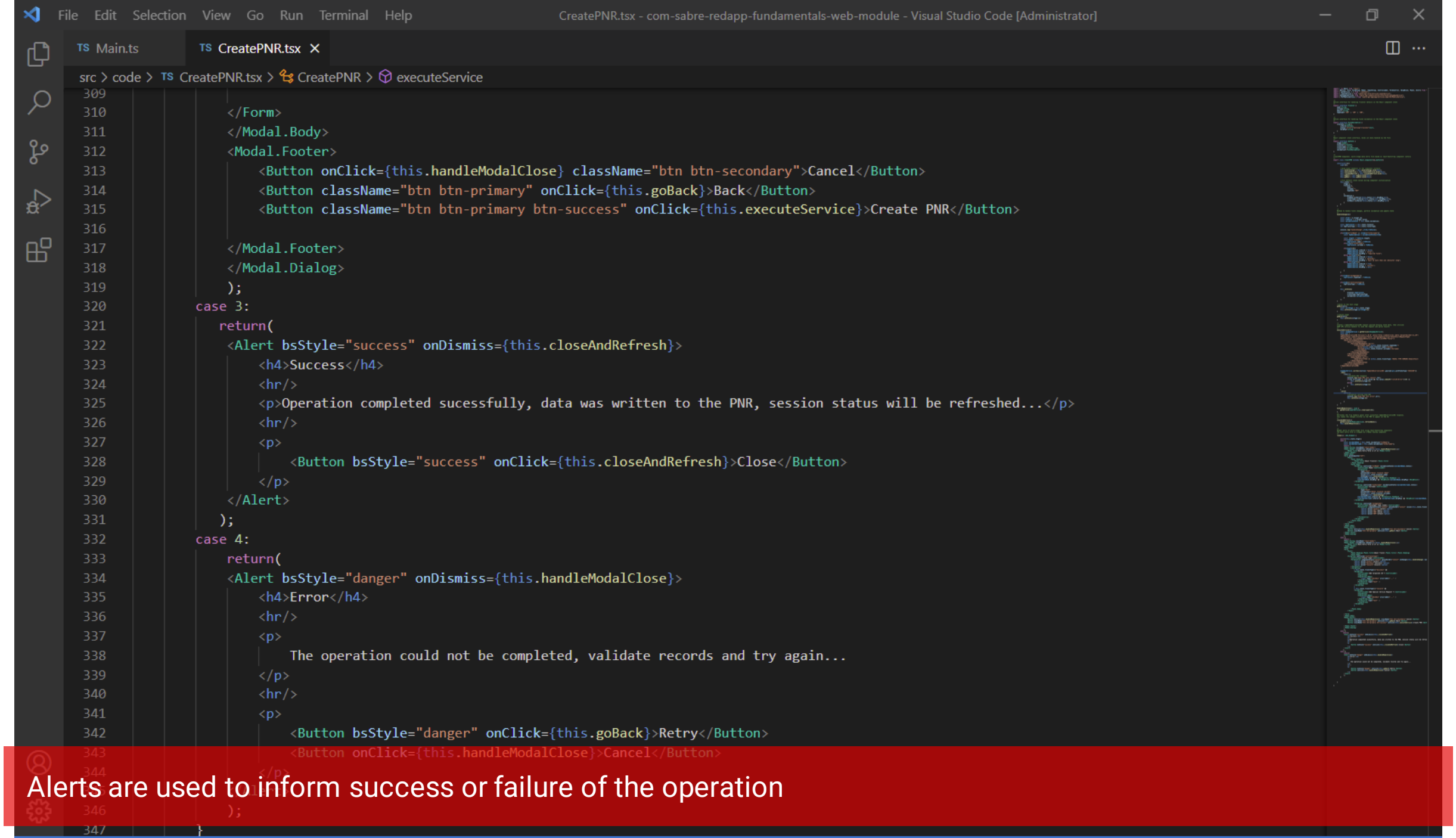

File Edit Selection View Go Run Terminal Help
• CreatePNR.tsx - com-sabre-redapp-fundamentals-web-module - Visual Studio Code [Administrator]
Main.ts
CreatePNR.tsx
src > code > CreatePNR.tsx > CreatePNR > executeService
/*
Creates a UpdateReservationRQ request payload merging state data, then utilizes
SOAP API service handler to send the request and parse results
*/
executeService(){
const soapApiService = getService(ISoapApiService);
const pl1 = `
<UpdateReservationRQ Version="1.19.8" xmlns="http://webservices.sabre.com/pnrbuilder/v1_19">
<RequestType commitTransaction="false" initialIgnore="true">Stateful</RequestType>
<ReturnOptions IncludeUpdateDetails="true" RetrievePNR="false"/>
<ReservationUpdateList>
<ReservationUpdateItem>
<PassengerNameUpdate op="C">
<TravelerName type="${this.state.traveler.typeCode}">
<Given>${this.state.traveler.name}</Given>
<Surname>${this.state.traveler.surname}</Surname>
</TravelerName>
</PassengerNameUpdate>
</ReservationUpdateItem>
<ReservationUpdateItem>
<RemarkUpdate op="C">
<RemarkText>THIS IS ${this.state.travelType} TRAVEL TYPE REMARK</RemarkText>
</RemarkUpdate>
</ReservationUpdateItem>
</ReservationUpdateList>
</UpdateReservationRQ>
`;
soapApiService.callSws({action:"UpdateReservationRQ",payload:pl1,authTokenType:"SESSION"})
.then(
(res)=>{
//validate API response
console.log("Soap API call result",res);
if(res.errorCode || (res.value && res.value.indexOf("<stl19:Error")>=0)){
this.setState({stage:4})
this.setState({stage:3})
}
Upon form submission Sabre SOAP API is called to write data to the current PNR on the user session.
Ln 180, Col 1 Spaces: 4 UTF-8 CRLF TypeScript React



This react component is based entirely on the React-bootstrap (v3) library available on the web SDK, more info could be found @ https://react-bootstrap-v3-netlify.app/getting-started/introduction



Multi-stage form with validation, implementation details


File Edit Selection View Go Run Terminal Help
CreatePNR.tsx - com-sabre-redapp-fundamentals-web-module - Visual Studio Code [Administrator]
Main.ts
CreatePNR.tsx
src > code > CreatePNR.tsx > CreatePNR > executeService
</Form>
</Modal.Body>
<Modal.Footer>
<Button onClick={this.handleModalClose} className="btn btn-secondary">Cancel</Button>
<Button className="btn btn-primary" onClick={this.goBack}>Back</Button>
<Button className="btn btn-primary btn-success" onClick={this.executeService}>Create PNR</Button>
</Modal.Footer>
</Modal.Dialog>
);
case 3:
return(
<Alert bsStyle="success" onDismiss={this.closeAndRefresh}>
<h4>Success</h4>
<hr/>
Operation completed sucessfully, data was written to the PNR, session status will be refreshed...
<hr/>

<Button bsStyle="success" onClick={this.closeAndRefresh}>Close</Button>

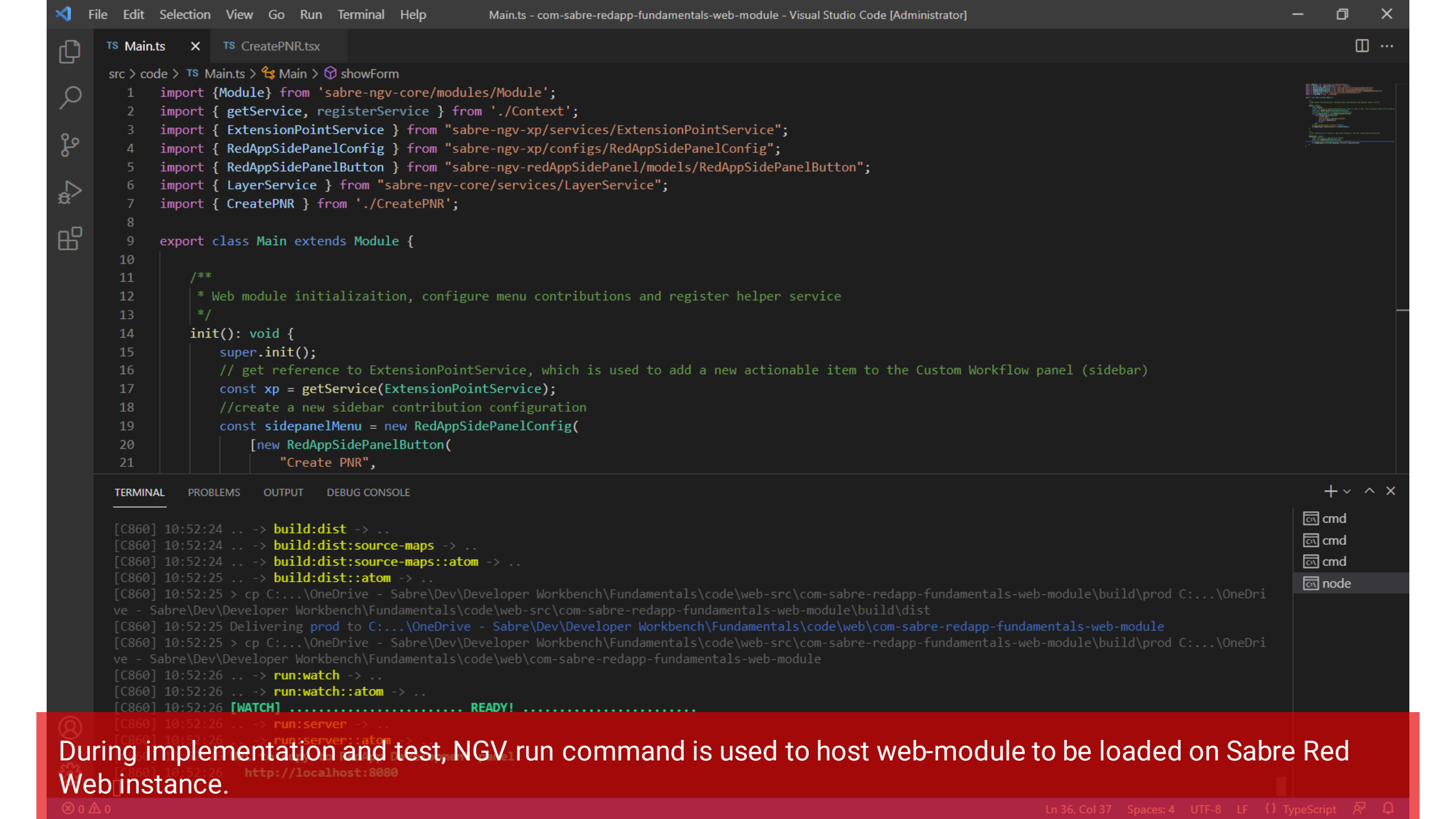
</Alert>
);
case 4:
return(
<Alert bsStyle="danger" onDismiss={this.handleModalClose}>
<h4>Error</h4>
<hr/>

The operation could not be completed, validate records and try again...

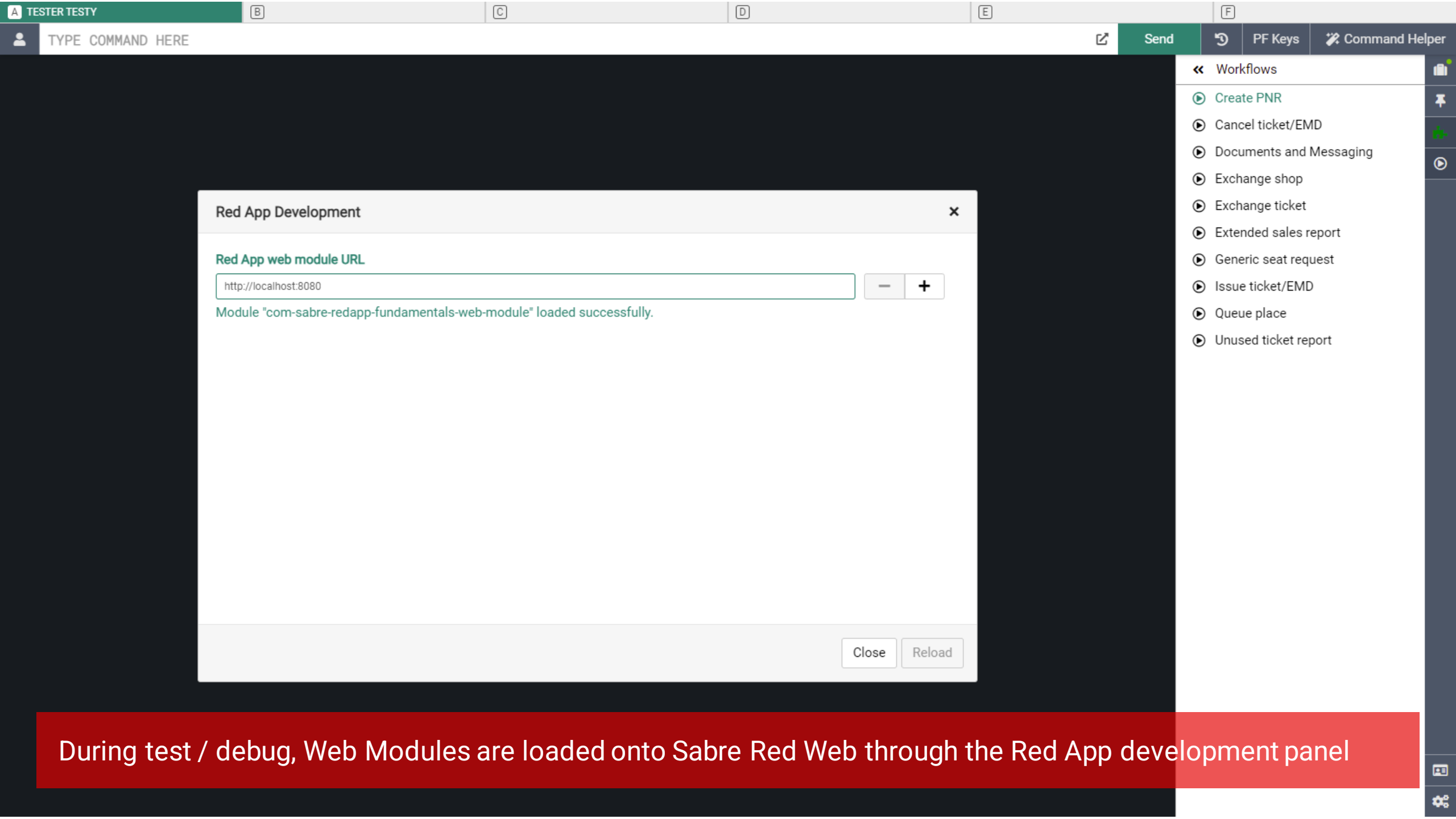
<hr/>

<Button bsStyle="danger" onClick={this.goBack}>Retry</Button>
<Button onClick={this.handleModalClose}>Cancel</Button>

);
}
Alerts are used to inform success or failure of the operation
Ln 180, Col 1
Spaces: 4
UTF-8
CRLF
TypeScript React



During implementation and test, NGV run command is used to host web-module to be loaded on Sabre Red Web instance.


A TESTER TESTY
B
C
D
E
F
TYPE COMMAND HERE
Send
PF Keys
Command Helper
Workflows
Create PNR
Cancel ticket/EMD
Documents and Messaging
Exchange shop
Exchange ticket
Extended sales report
Generic seat request
Issue ticket/EMD
Queue place
Unused ticket report
Red App Development
Red App web module URL
http://localhost:8080
Module "com-sabre-redapp-fundamentals-web-module" loaded successfully.
Close
Reload
During test / debug, Web Modules are loaded onto Sabre Red Web through the Red App development panel


Sabre®