EPAP MOBILE APPLICATION

Project Turnover Document

Prepared for: Dr. Aalap Shah & Dr. Vikas O'Reilly-Shah, Project Sponsors

Prepared by: Team Crawford Long, Terry College of Business, University of Georgia

April 1, 2018

Proposal Version 1.0

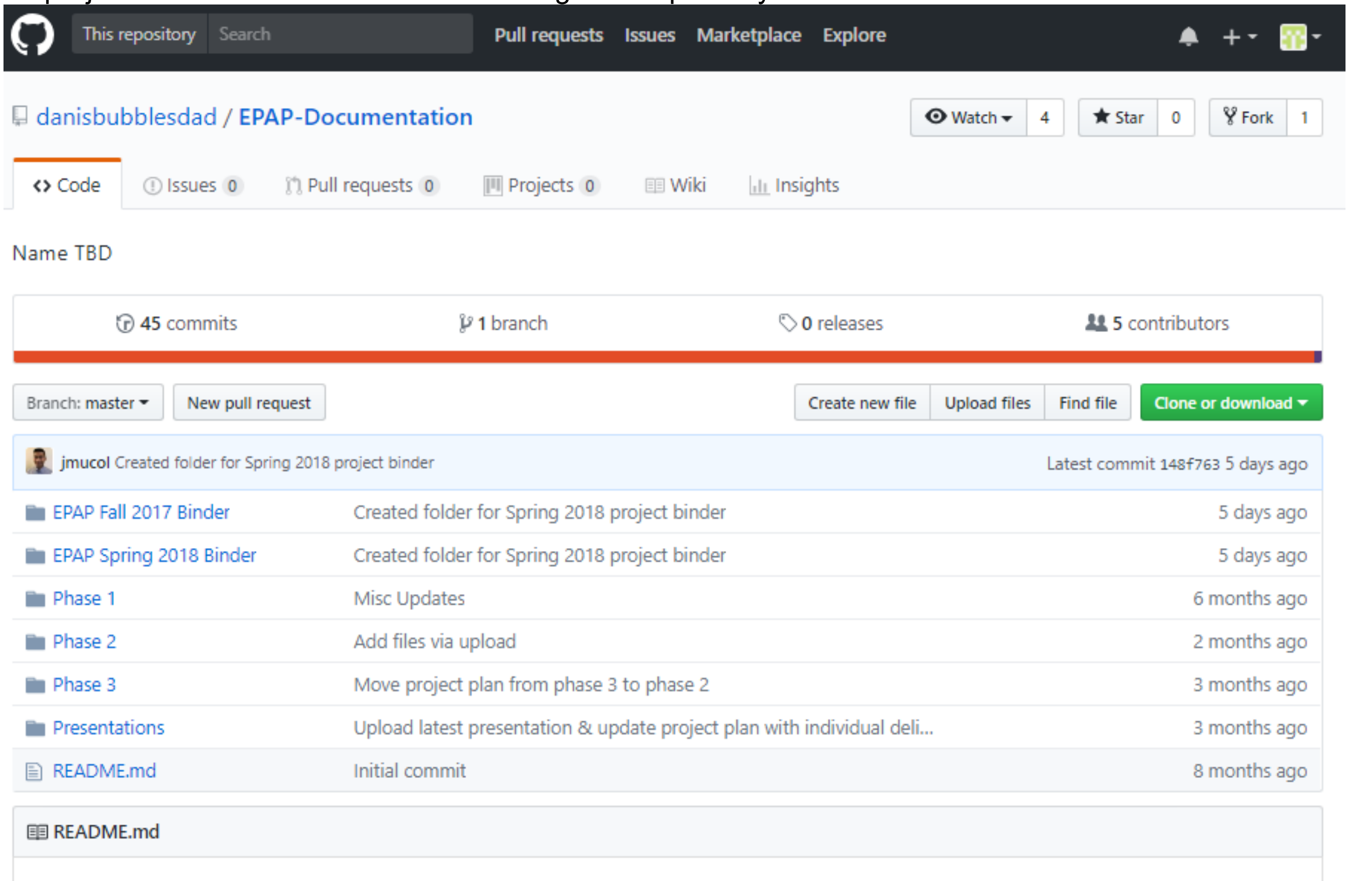
Document Purpose

As part of the transition of this application from Team Crawford Long to the team's project sponsors, this document will attempt to give them a general overview of the technologies involved in the development of the application. This will include the following:

- We will begin by sharing the github repository in which our code and documentation is located.
- We will share Kumulos access as well, making sure they have access and the ability to grant access to any future developers.
- An overview of some of the development tools we used on the application including:
    - React Native
    - NPM
    - Android SDK
- We will also document any outstanding defects, and any features that we were unable to deliver
- We will also share our test approach.
- Finally, we will share an overview of the code.

Github Repository

All project documentation is housed in the github repository EPAP-Documentation.



danisbubblesdad / EPAP-Documentation

Name TBD

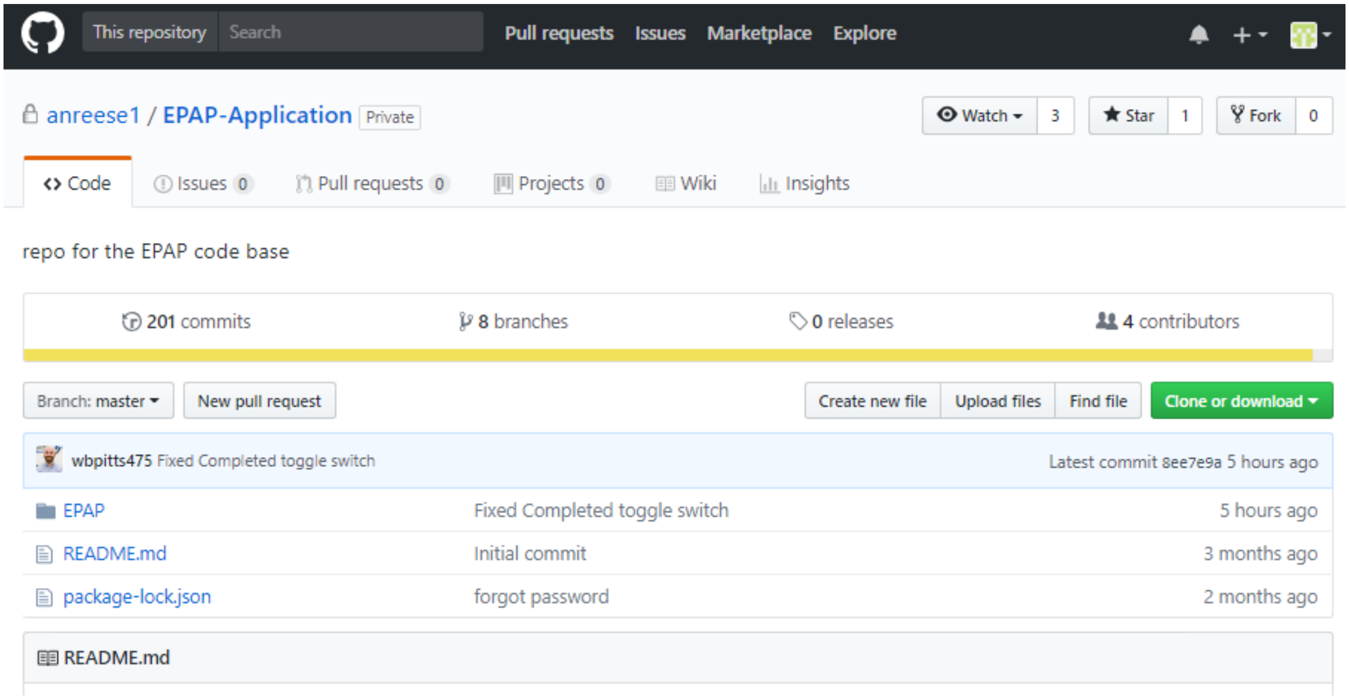
45 commits | 1 branch | 0 releases | 5 contributors

jmucol Created folder for Spring 2018 project binder | Latest commit 148f763 5 days ago

EPAP Fall 2017 Binder	Created folder for Spring 2018 project binder	5 days ago
EPAP Spring 2018 Binder	Created folder for Spring 2018 project binder	5 days ago
Phase 1	Misc Updates	6 months ago
Phase 2	Add files via upload	2 months ago
Phase 3	Move project plan from phase 3 to phase 2	3 months ago
Presentations	Upload latest presentation & update project plan with individual deli...	3 months ago
README.md	Initial commit	8 months ago

README.md

All project source code is housed in anreese1/EPAP-Application. **The final version of the code delivered by Team Crawford Long iis in the branch Phase 4 Master**



anreese1 / EPAP-Application Private

repo for the EPAP code base

201 commits 8 branches 0 releases 4 contributors

wbpitts475 Fixed Completed toggle switch Latest commit 8ee7e9a 5 hours ago

EPAP	Fixed Completed toggle switch	5 hours ago
README.md	Initial commit	3 months ago
package-lock.json	forgot password	2 months ago

README.md

Both Dr Aalap Shah and Dr Vikas O’Reilly-Shah have been given access to these two repositories.

Kumulos Mobile Backend

Background

Kumulos is a mobile backend as a service (MBaaS) utilized for the EPAP project. The EPAP application accesses and writes to the Kumulos database using API methods. It serves as the backend infrastructure on the server-side.

Data is stored in table structures and API methods are used by the application to access, add, modify, or delete the data stored in Kumulos. The requested data is returned and displayed in the app in various ways.

Accessing Kumulos

Kumulos can be accessed via a web browser at https://kumulos.app.delivery provided the user has login credentials. The project is listed under EPAP Prototype. As of April 2018, the service is subscribed and active. The project sponsors (Dr. Vikas O'Reilly-Shah and Dr Aalap Shah) also have been granted access. If new requests are needed for the project sponsors, please contact John Peeler at john.peeler25@uga.edu

Most of the features relevant to this project can be accessed in the left sidebar after logging in under the Build section, under which there are four subsections:

- Tables: New tables may be created and existing tables can be accessed. The contents of the individual tables, such as fields and the data itself, can be created or modified from this menu.
- API: Access to the data is achieved through custom API methods
- Deploy: Deployment platforms may be chosen in this menu (iOS and Android for this project)
- Download: Native libraries may be downloaded

Build: Tables

Fourteen tables are used by the EPAP application:

Table Name	Fields
Drug Plan Items	drugPlanItemNotes, drugPlanItemDosage, drugPlan, drug
Drug Plans	drugPlanStatus, drugPlanNotesToPhysician, drugPlanNotesToPharmacist, drugPlanExtubation, drugPlanMaintenance, primaryPharmacist, plan

Drugs	drugName, drugCategory, drugType, drugUnitOfMeasure, drugNeedsNotes, drugNeedsDosage, createdByUser
Equipment	equipmentName, equipmentCategory, equipmentType, equipmentUnitOfMeasure, equipmentNeedsLocation, equipmentNeedsSize, equipmentNeedsNotes, createdByUser
Equipment Plan Items	equipmentPlanItemLocation, equipmentPlanItemSize, equipmentPlanItemNotes, equipmentPlanItemQuantity, equipment, equipmentPlan
Equipment Plans	equipmentPlanStatus, equipmentPlanNotesToPhysician, equipmentPlanLabValuesType, equipmentPlanHasCSpineOrTMJIssue, equipmentPlanPosition, equipmentPlanTrend, equipmentPlanProneInfo, equipmentPlanNotesToTech, equipmentPlanRotation, plan, primaryTechnician
Messages	messageSubject, messageBody, messageSender, messageRecipient, messagePlan
Plans	planDescription, planStatus, planPatientInitials, planPatientHeight, planPatientWeight, planPatientAge, planPatientSex, planORDate, planAnesthesiaType, planNPODate, planSavePoint, primaryPhysician, procedure, equipmentPlan, drugPlan, createdByUser, startRoom, operatingRoom, endRoom
Procedure Drugs	procedureDrugQuantity, pharmaceutical, procedure
Procedure Equipments	procedureEquipmentQuantity, equipment, procedure
Procedures	procedureName, procedureDescription, createdByUser
	roleName

Roles	
Rooms	roomNumber, roomType
Users	username, userPassword, userFirstName, userLastName, userIsActive, userIsAdministrator, userJoinDate, userAuthToken, userResetCode, userIsRegistered, userResetCodeDate, userAuthTokenDate, role

Build: API

There are numerous API methods used by the EPAP application:

Table	Aliases
Drug Plan Items	• createDrugPlanItem • deleteDrugPlanItems • getDrugPlanItems
Drug Plans	• createDrugPlan • getDrugPlan • updateDrugPlan • updateDrugPlanStatus
Drugs	• addDrug • deleteDrug • drugDQ • getAllDrugs • getDrug • updateDrug
Equipment	• addEquipment • deleteEquipment • getAllEquipment • getEquipment • updateEquipment
Equipment Plan Items	• createEquipmentPlanItem • deleteEquipmentPlanItems • epiDQ • getEquipmentPlanItems
Equipment Plans	• createEquipmentPlan • getEquipmentPlan • updateEquipmentPlan • updateEquipmentPlanStatus

Messages	• addMessage • deleteMessage • getAllMessages • getMessage
Plans	• createPlan • deletePlan • getAllPlans • getPlan • getPlanRawData • getPlansByStatus • getUserPlans • getUserPlansByStatus • planDQ • updatePlan • updatePlanStatus
Roles	• createRole • selectRoles
Rooms	• addRoom • deleteRoom • getAllRooms • updateRoom
Users	• authenticateUser • changeRole • createUser • getUser • selectAllUsers • setUserAuthToken • setUserResetCode • updateUserByAdmin • updateUserPassword • updateUserProfile • validateResetCode

React Native Overview

React Native lets you build mobile apps using only JavaScript. It uses the same design as React, letting you compose a rich mobile UI from declarative components. With React Native, you don't build a "mobile web app", an "HTML5 app", or a "hybrid app". You build a real mobile app that's indistinguishable from an app built using Objective-C or Java. React Native uses the same fundamental UI building blocks as regular iOS and Android apps. You just put those building blocks together using JavaScript and React.

https://facebook.github.io/react-native/

Android SDK Tools

The Android SDK provides you the API libraries and developer tools necessary to build, test, and debug apps for Android.

If you're a new Android developer, we recommend you download the ADT Bundle to quickly start developing apps. It includes the essential Android SDK components and a version of the Eclipse IDE with built-in **ADT (Android Developer Tools)** to streamline your Android app development.

With a single download, the ADT Bundle includes everything you need to begin developing apps:

- Eclipse + ADT plugin
- Android SDK Tools
- Android Platform-tools
- The latest Android platform
- The latest Android system image for the emulator

Team Crawford Long used Atom as the IDE to develop the initial version of the EPAP application. It interfaced incredibly well with GitHub, Android SDK, and npm.

To get the ADT Bundle or stand-alone SDK Tools, please visit the web site at developer.android.com/sdk/

https://stuff.mit.edu/afs/sipb/project/android/docs/sdk/index.html

NPM

npm has the world's largest software registry, with approximately 3 billion downloads per week and opens up a whole world of javascript for developers. The registry contains over 600,000 packages of source code for all developers to download as they see fit. Open-source developers from every continent use npm to share and borrow packages, and many organizations use npm to manage private development as well.
The EPAP application was built using npm. A brief introduction to npm can be found at the following link:

https://www.youtube.com/watch?v=pa4dc480Apo&list=PLQso55XhxkgBMeiYmFEHzz1axDUBjTLC6

Here are a few of the ways developers use npm.

- Adapt packages of code to your apps, or incorporate packages as they are.
- Download standalone tools you can use right away.
- Run packages without downloading using npx.
- Share code with any npm user, anywhere.
- Restrict code to specific developers.
- Form Orgs (organizations) to coordinate package maintenance, coding, and developers.
- Form virtual teams by using Orgs.
- Manage multiple versions of code and code dependencies.
- Update applications easily when underlying code is updated.
- Discover multiple ways to solve the same puzzle.
- Find other developers who are working on similar problems and projects.

The download for npm can be found at https://www.npmjs.com/get-npm.

https://docs.npmjs.com/getting-started/what-is-npm

Defects & Features Outstanding

- Edit Plan Functionality is incomplete
- A user cannot view themselves in the user listing or deactivate/revoke admin privileges for themselves
- The notifications/messages tab does not have error handling in place for if a plan is removed/discarded
- Plans cannot be transferred
- Procedure creation and usage functionality was not implemented

Test Approach

Test approaches were based mainly on the use cases defined in the analysis and design phases of the project. As the project flowed and changed test approaches were added where necessary.

Administrator

User Registration - verified that a new user can enter their e-mail address, step through the email process and log into the application as a new user.

User Login - verified that an existing user can login to the application

Reset Password - verified that the user can request a password reset, receive the reset e-mail and successfully reset their password

Add Medical Supplies - verified that a user can successfully add medical supplies to the application

Add Pharmaceuticals - verified that a user can successfully add pharmaceutical supplies to the application

Physician

Create Plan verified that a user can successfully create a EPAP plan in the application

Enter Patient Medical Data – verified that a user can enter medical details on their patients

DoS View – verified that each user has a day of surgery view displayed for their procedures

Forward plan request – verified that a plan can be submitted and that it forwards to the next user

Enter Procedure Information – verified that procedure information can be entered for a patient

Enter Plan Notes – verified that a user can enter plan notes on certain items within a plan

Enter Messages – verified that a user can enter messages and forward them to other users

“Clawback” Submitted Plan – verified that a user can claw back a plan that has been submitted

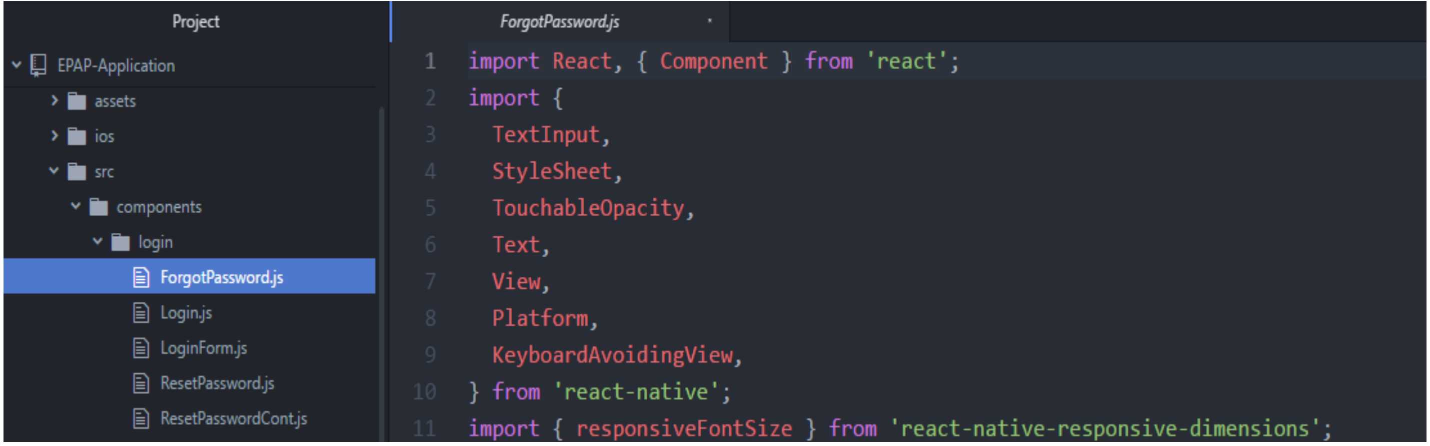
Pharmacist/Technicians

Update Plan Status – verified that a user can update the status/complete a plan

Enter Messages – verified that a user can enter messages and forward them to another users

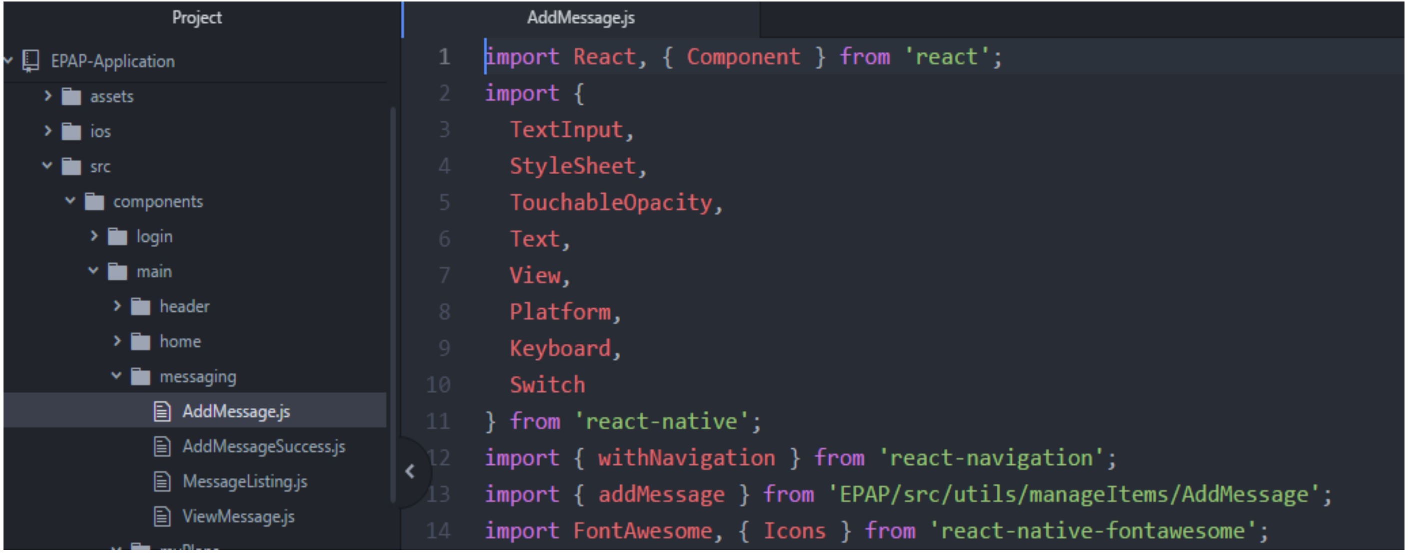
Code Overview

Administrative Login/Password Resets



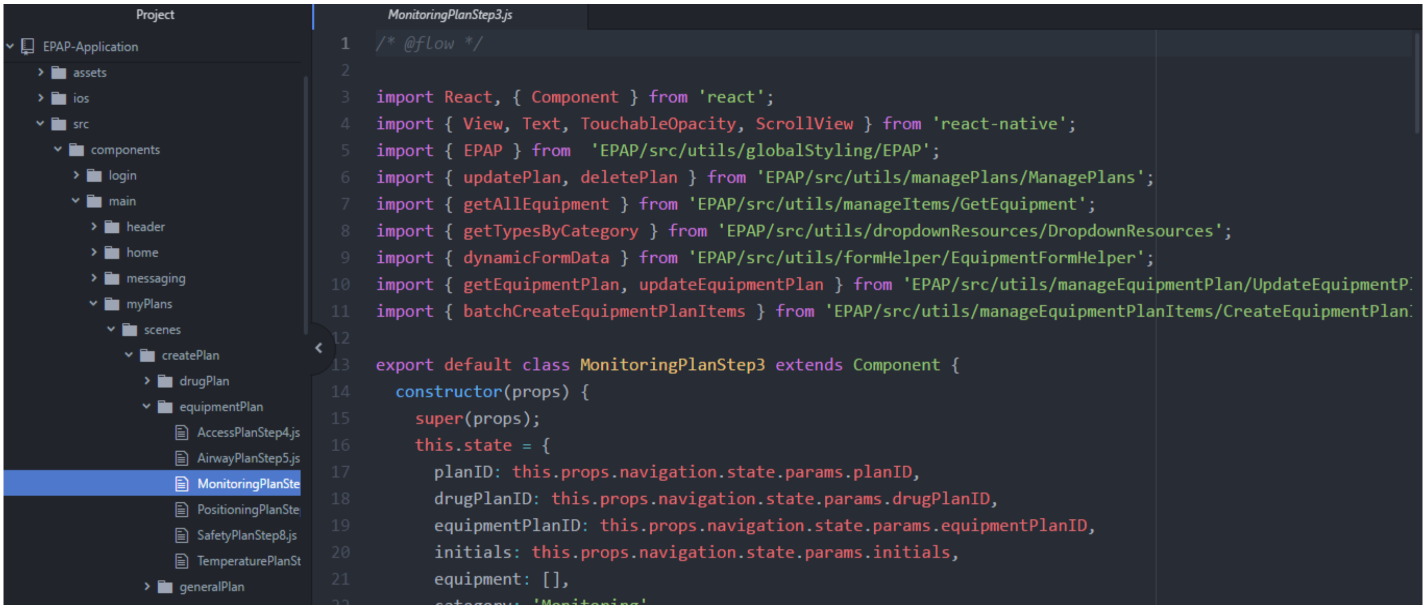
```
import React, { Component } from 'react';
import {
  TextInput,
  StyleSheet,
  TouchableOpacity,
  Text,
  View,
  Platform,
  KeyboardAvoidingView,
} from 'react-native';
import { responsiveFontSize } from 'react-native-responsive-dimensions';
```

Messaging



```
import React, { Component } from 'react';
import {
  TextInput,
  StyleSheet,
  TouchableOpacity,
  Text,
  View,
  Platform,
  Keyboard,
  Switch
} from 'react-native';
import { withNavigation } from 'react-navigation';
import { addMessage } from 'EPAP/src/utils/manageItems/AddMessage';
import FontAwesome, { Icons } from 'react-native-fontawesome';
```

Create Plans for drug equipment and general plans

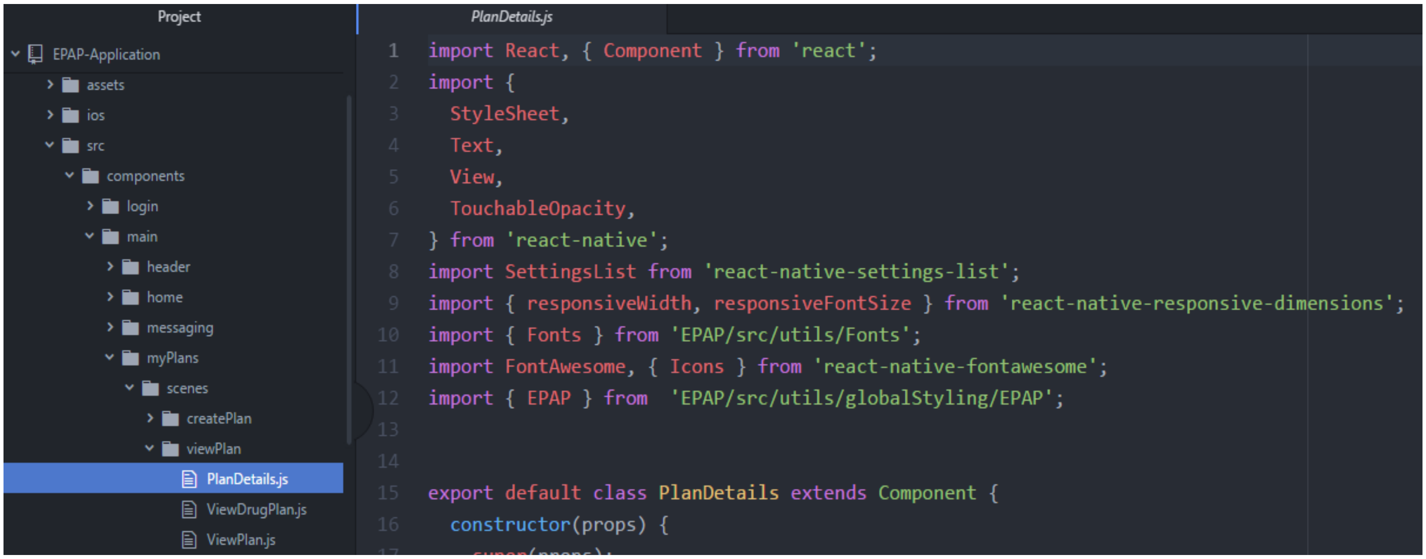


```
/* @flow */

import React, { Component } from 'react';
import { View, Text, TouchableOpacity, ScrollView } from 'react-native';
import { EPAP } from  'EPAP/src/utils/globalStyling/EPAP';
import { updatePlan, deletePlan } from 'EPAP/src/utils/managePlans/ManagePlans';
import { getAllEquipment } from 'EPAP/src/utils/manageItems/GetEquipment';
import { getTypesByCategory } from 'EPAP/src/utils/dropdownResources/DropdownResources';
import { dynamicFormData } from 'EPAP/src/utils/formHelper/EquipmentFormHelper';
import { getEquipmentPlan, updateEquipmentPlan } from 'EPAP/src/utils/manageEquipmentPlan/UpdateEquipmentP
import { batchCreateEquipmentPlanItems } from 'EPAP/src/utils/manageEquipmentPlanItems/CreateEquipmentPlan

export default class MonitoringPlanStep3 extends Component {
  constructor(props) {
    super(props);
    this.state = {
      planID: this.props.navigation.state.params.planID,
      drugPlanID: this.props.navigation.state.params.drugPlanID,
      equipmentPlanID: this.props.navigation.state.params.equipmentPlanID,
      initials: this.props.navigation.state.params.initials,
      equipment: [],
      category: 'Monitoring',
```

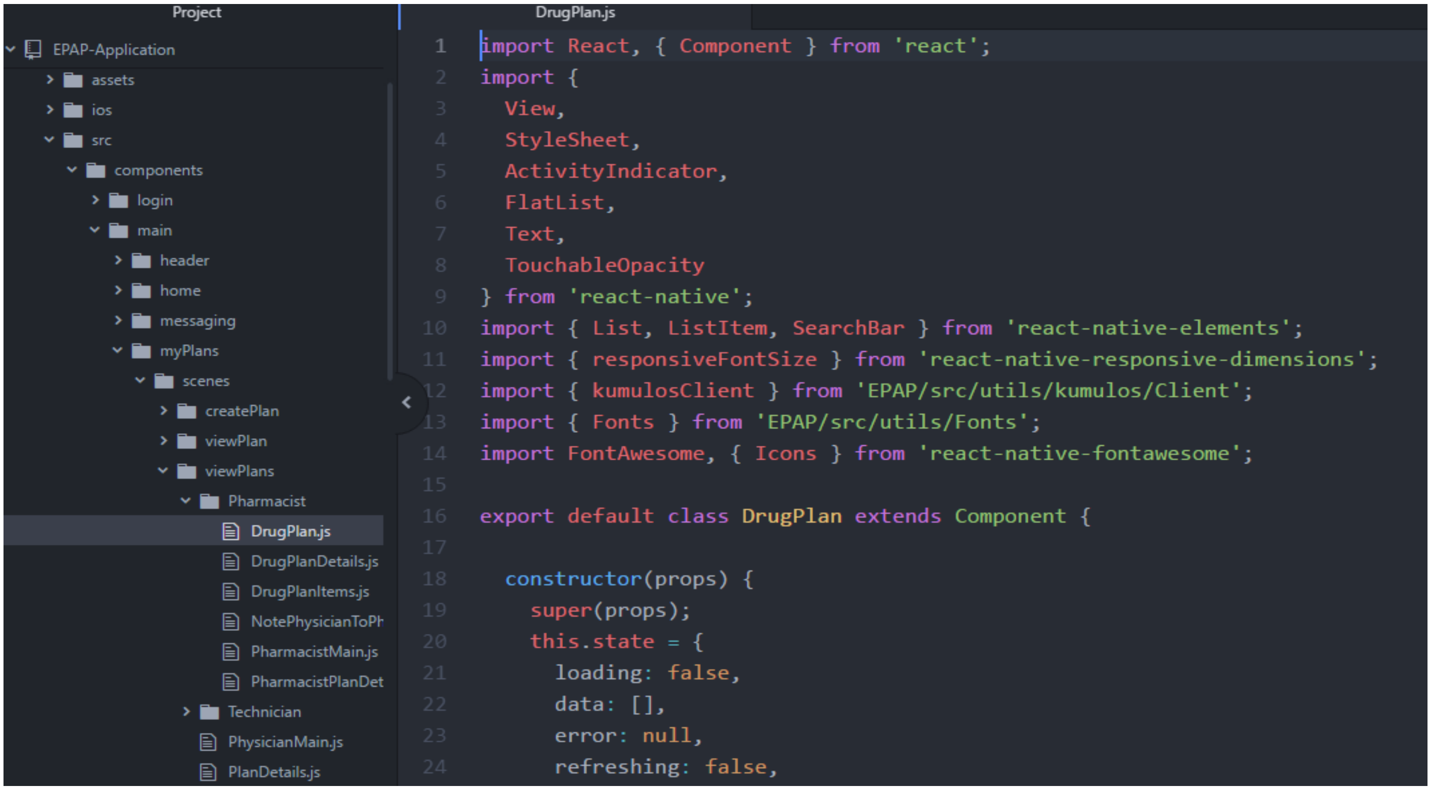
View Plan Details



```
import React, { Component } from 'react';
import {
  StyleSheet,
  Text,
  View,
  TouchableOpacity,
} from 'react-native';
import SettingsList from 'react-native-settings-list';
import { responsiveWidth, responsiveFontSize } from 'react-native-responsive-dimensions';
import { Fonts } from 'EPAP/src/utils/Fonts';
import FontAwesome, { Icons } from 'react-native-fontawesome';
import { EPAP } from  'EPAP/src/utils/globalStyling/EPAP';


export default class PlanDetails extends Component {
  constructor(props) {
    super(props);
```

View Plans Pharmacist and Technician

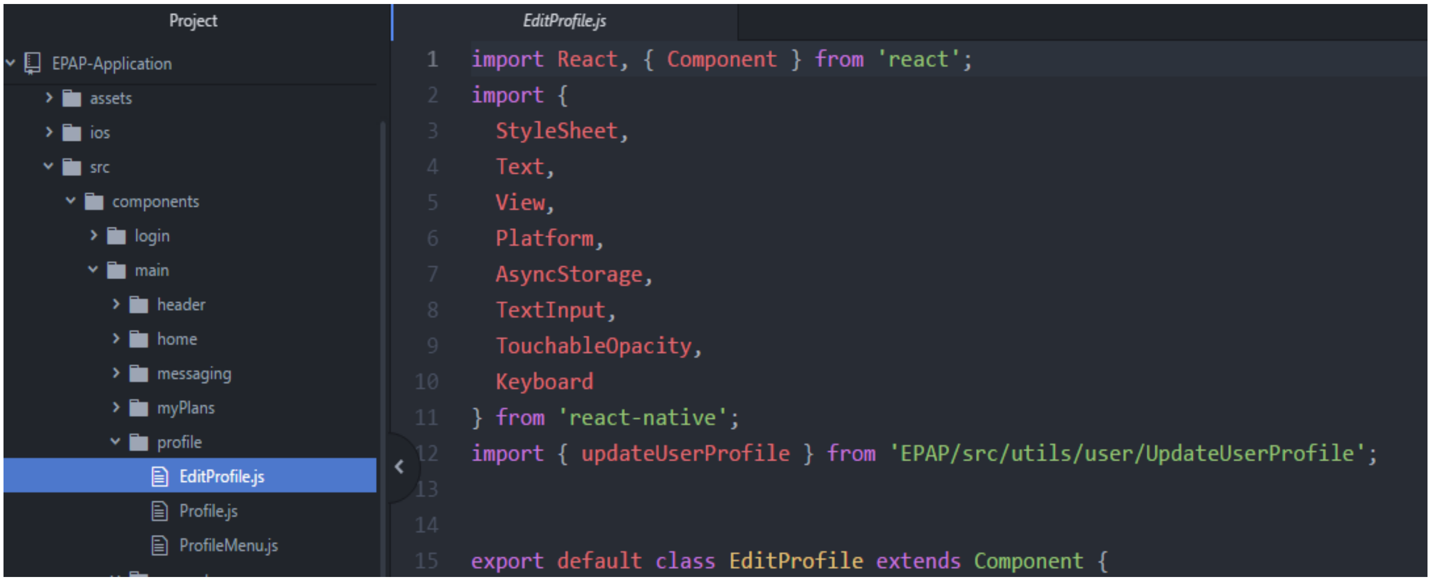


```
import React, { Component } from 'react';
import {
  View,
  StyleSheet,
  ActivityIndicator,
  FlatList,
  Text,
  TouchableOpacity
} from 'react-native';
import { List, ListItem, SearchBar } from 'react-native-elements';
import { responsiveFontSize } from 'react-native-responsive-dimensions';
import { kumulosClient } from 'EPAP/src/utils/kumulos/Client';
import { Fonts } from 'EPAP/src/utils/Fonts';
import FontAwesome, { Icons } from 'react-native-fontawesome';

export default class DrugPlan extends Component {

  constructor(props) {
    super(props);
    this.state = {
      loading: false,
      data: [],
      error: null,
      refreshing: false,
```

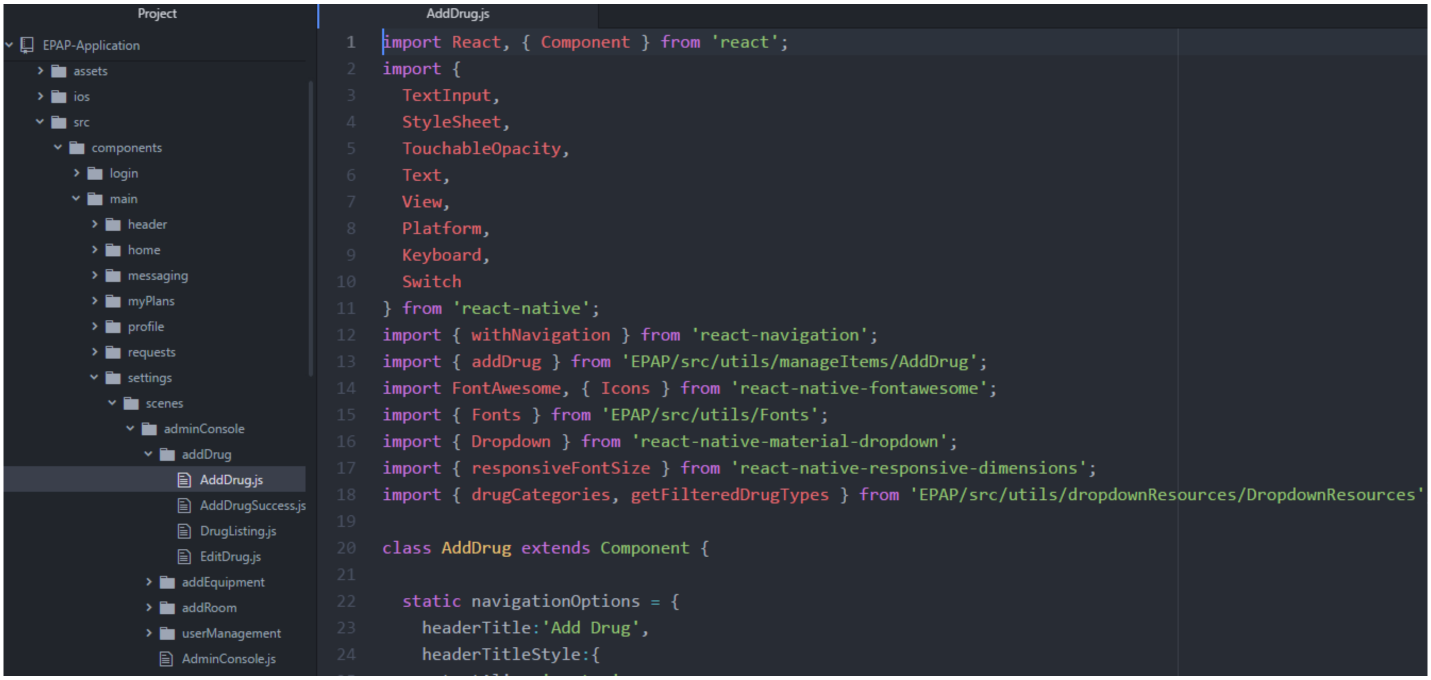
Edit User Profile



```
import React, { Component } from 'react';
import {
  StyleSheet,
  Text,
  View,
  Platform,
  AsyncStorage,
  TextInput,
  TouchableOpacity,
  Keyboard
} from 'react-native';
import { updateUserProfile } from 'EPAP/src/utils/user/UpdateUserProfile';


export default class EditProfile extends Component {
```

Administrator Console for Add Drugs, Add Equipment, Add Room, User Management

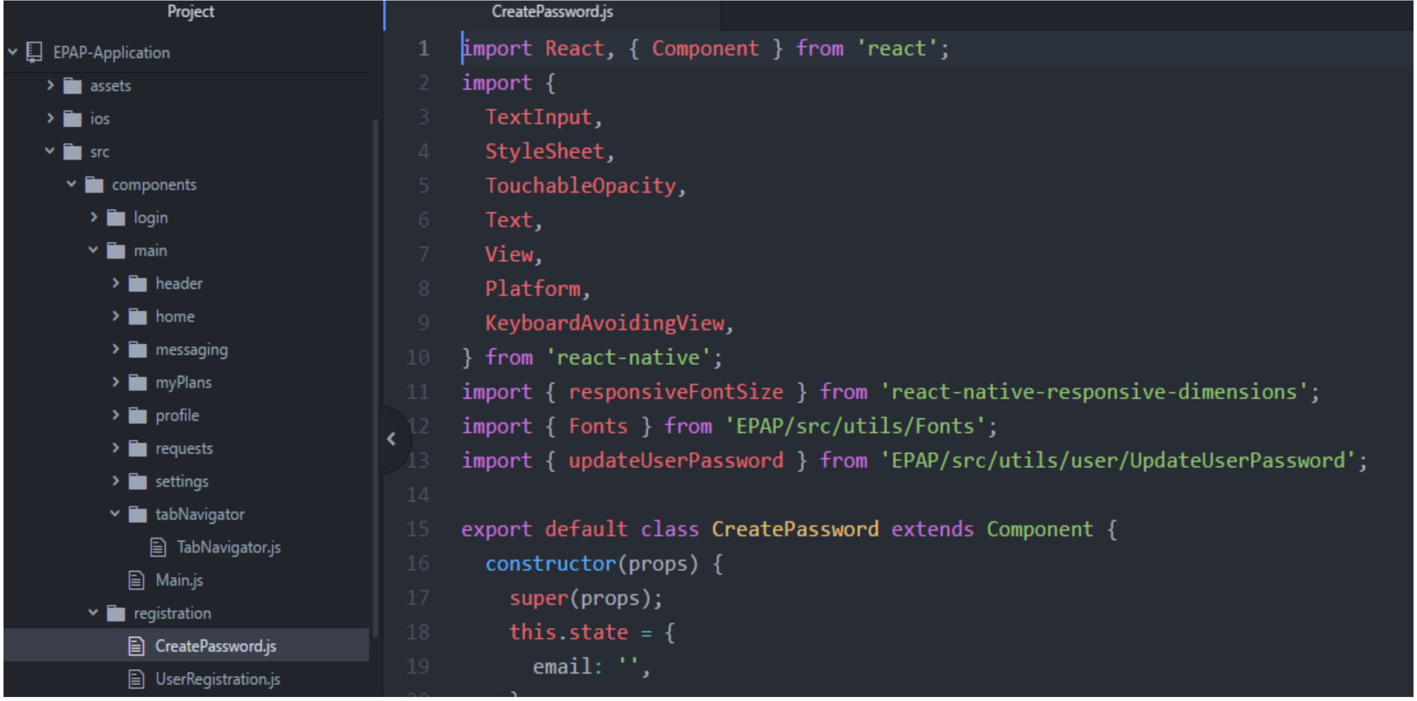


```
import React, { Component } from 'react';
import {
  TextInput,
  StyleSheet,
  TouchableOpacity,
  Text,
  View,
  Platform,
  Keyboard,
  Switch
} from 'react-native';
import { withNavigation } from 'react-navigation';
import { addDrug } from 'EPAP/src/utils/manageItems/AddDrug';
import FontAwesome, { Icons } from 'react-native-fontawesome';
import { Fonts } from 'EPAP/src/utils/Fonts';
import { Dropdown } from 'react-native-material-dropdown';
import { responsiveFontSize } from 'react-native-responsive-dimensions';
import { drugCategories, getFilteredDrugTypes } from 'EPAP/src/utils/dropdownResources/DropdownResources'

class AddDrug extends Component {

  static navigationOptions = {
    headerTitle:'Add Drug',
    headerTitleStyle:{
```

User Password and Registration



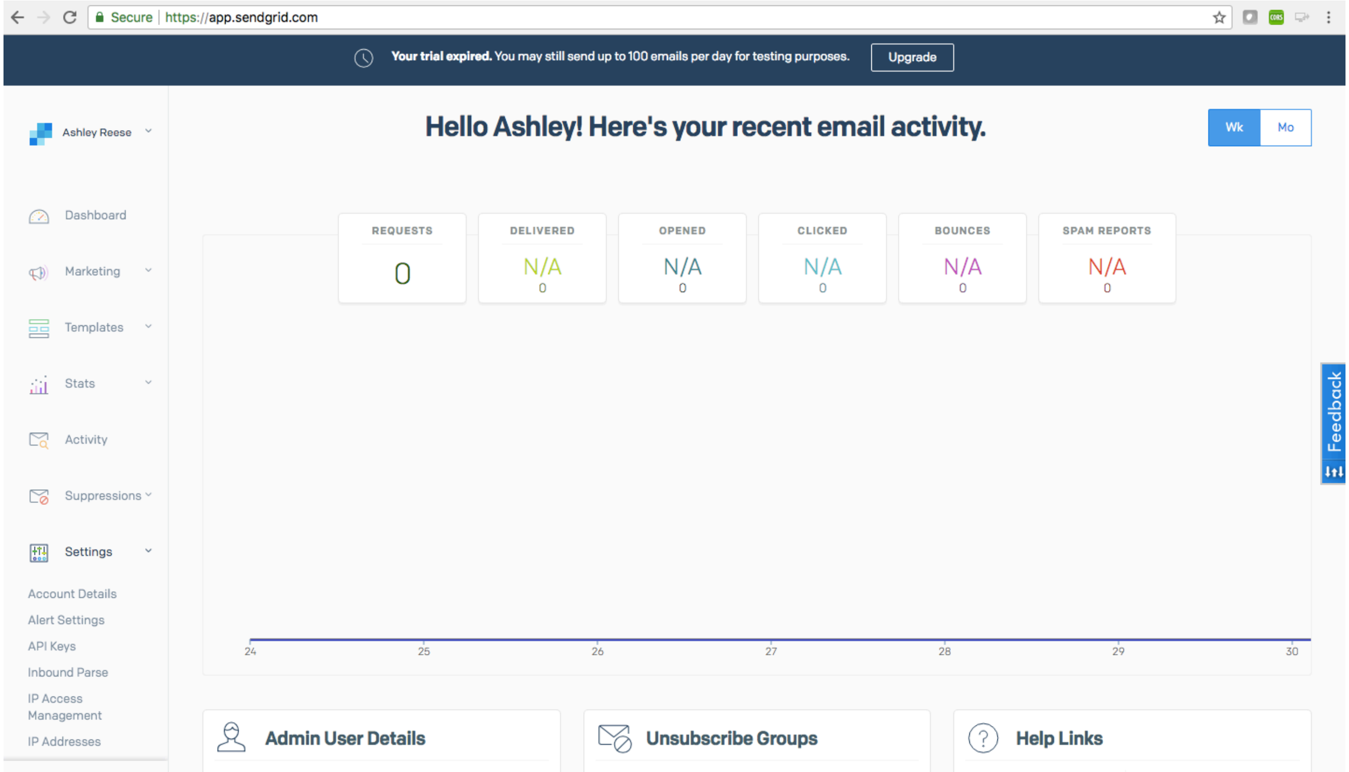
```
import React, { Component } from 'react';
import {
  TextInput,
  StyleSheet,
  TouchableOpacity,
  Text,
  View,
  Platform,
  KeyboardAvoidingView,
} from 'react-native';
import { responsiveFontSize } from 'react-native-responsive-dimensions';
import { Fonts } from 'EPAP/src/utils/Fonts';
import { updateUserPassword } from 'EPAP/src/utils/user/UpdateUserPassword';

export default class CreatePassword extends Component {
  constructor(props) {
    super(props);
    this.state = {
      email: '',
```

SendGrid Overview

SendGrid is a cloud-based SMTP provider that allows you to send email without having to maintain email servers. SendGrid is being used in the EPAP Application as an email server for the new user registration and the forgot password functionality.

UserName: EPAP
Password: EP@P2018

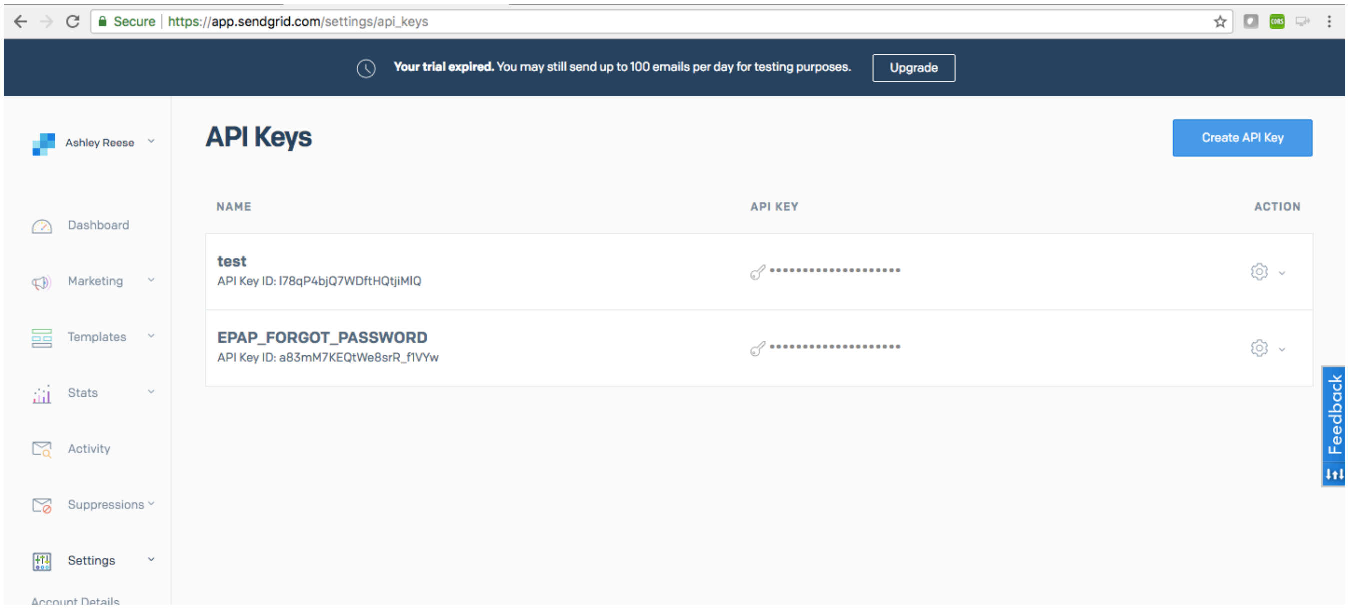


Your trial expired. You may still send up to 100 emails per day for testing purposes.

Hello Ashley! Here's your recent email activity.

REQUESTS	DELIVERED	OPENED	CLICKED	BOUNCES	SPAM REPORTS
0	N/A 0	N/A 0	N/A 0	N/A 0	N/A 0

To navigate to the API key for SendGrid click API keys in the left hand menu from the home page.

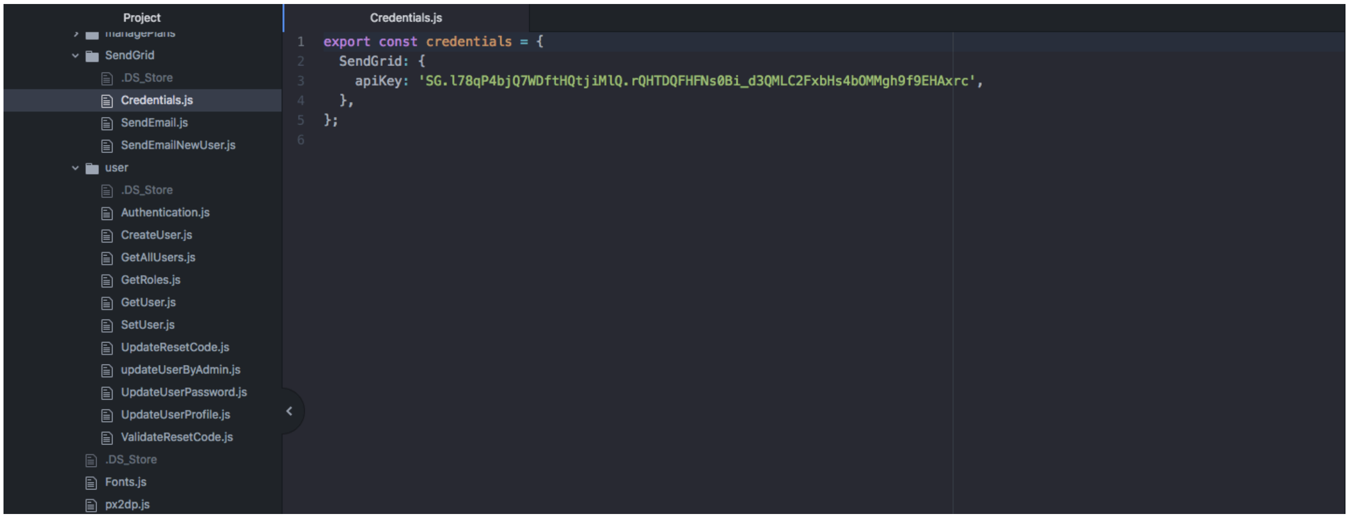


Your trial expired. You may still send up to 100 emails per day for testing purposes.

API Keys

NAME	API KEY	ACTION
test API Key ID: l78qP4bjQ7WDftHQtjiMlQ	••••••••••••••••••••	
EPAP_FORGOT_PASSWORD API Key ID: a83mM7KEQtWe8srR_f1VYw	••••••••••••••••••••	

SendGrid Credentials



```
export const credentials = {
  SendGrid: {
    apiKey: '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',
  },
};
```