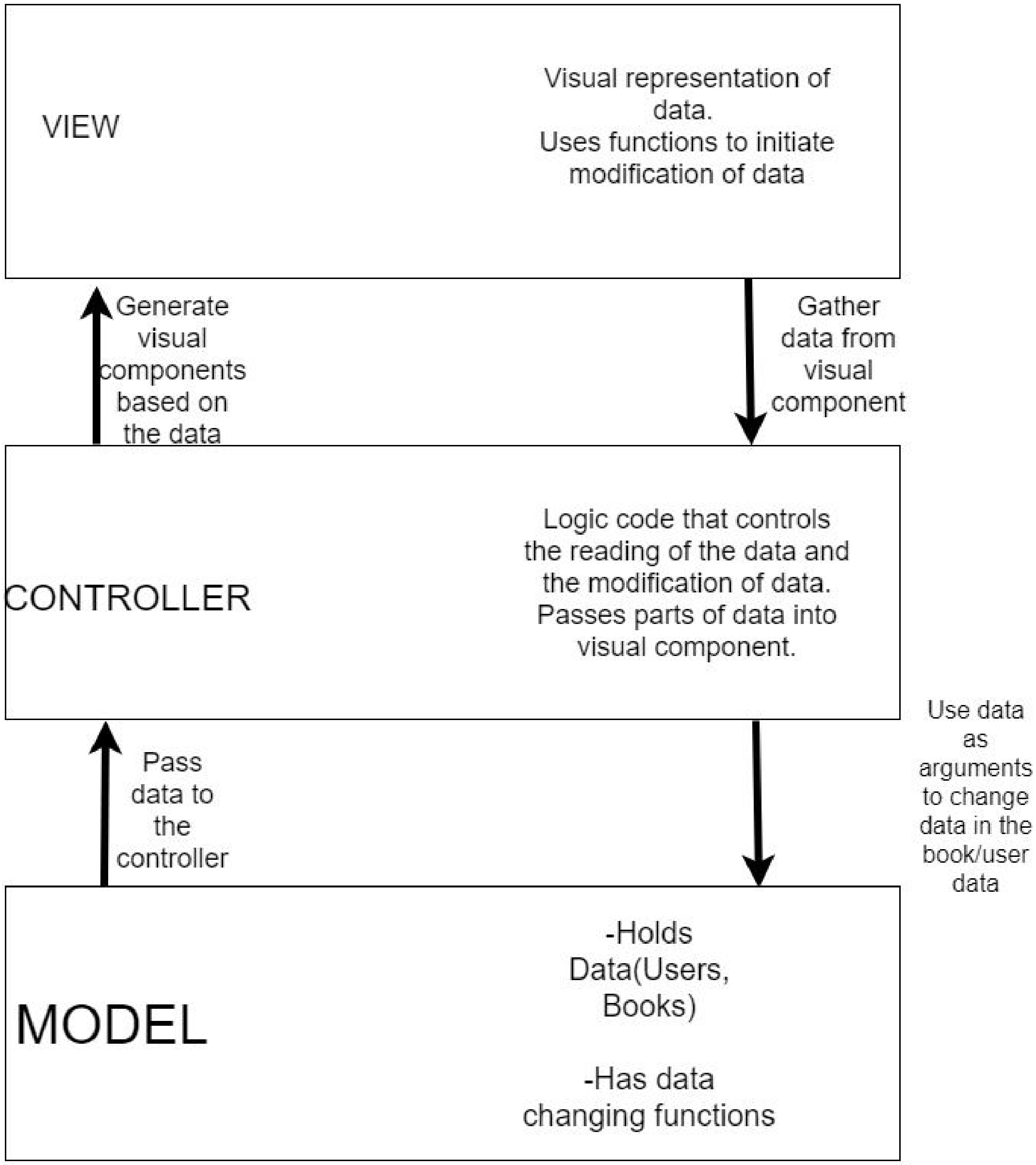
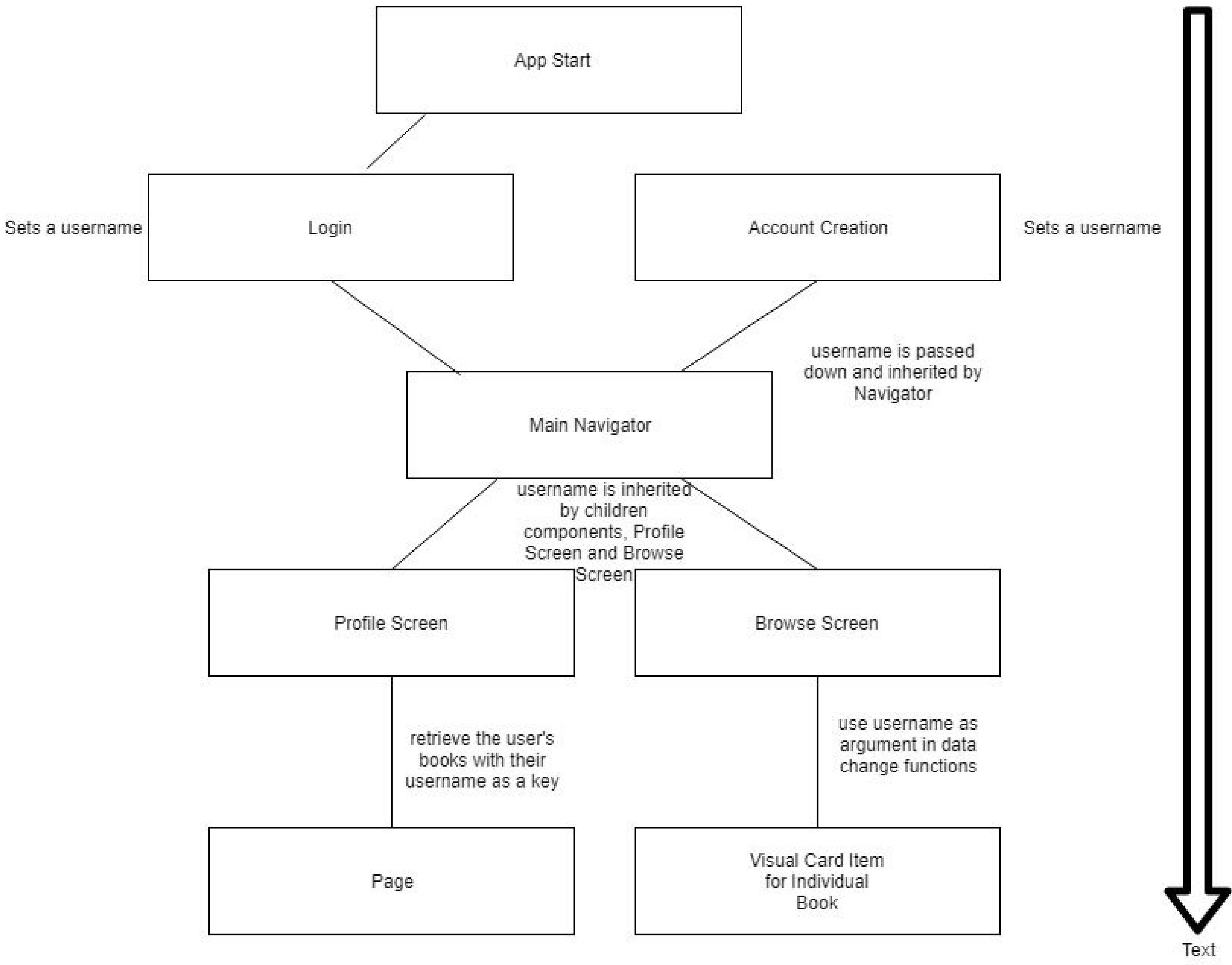
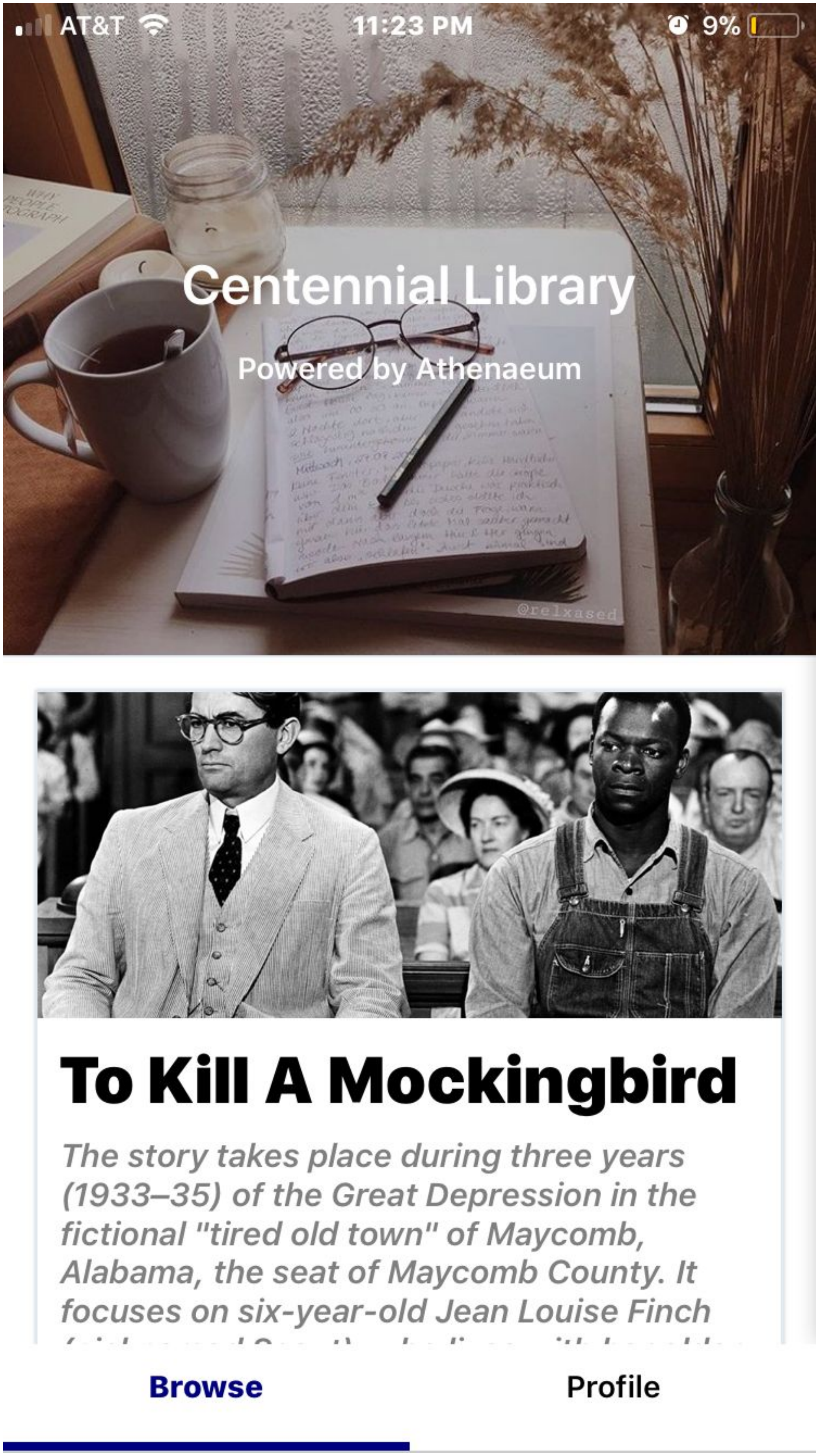
Data Structure


VIEW
Visual representation of data.
Uses functions to initiate modification of data
Generate visual components based on the data
Gather data from visual component
CONTROLLER
Logic code that controls the reading of the data and the modification of data.
Passes parts of data into visual component.
Use data as arguments to change data in the book/user data
Pass data to the controller
MODEL
-Holds Data(Users, Books)
-Has data changing functions

MVC Flowchart of Data


App Start
Sets a username
Login
Account Creation
Sets a username
username is passed down and inherited by Navigator
Main Navigator
username is inherited by children components, Profile Screen and Browse Screen
Profile Screen
Browse Screen
retrieve the user's books with their username as a key
use username as argument in data change functions
Page
Visual Card Item for Individual Book
Text

Username Data Flow in App

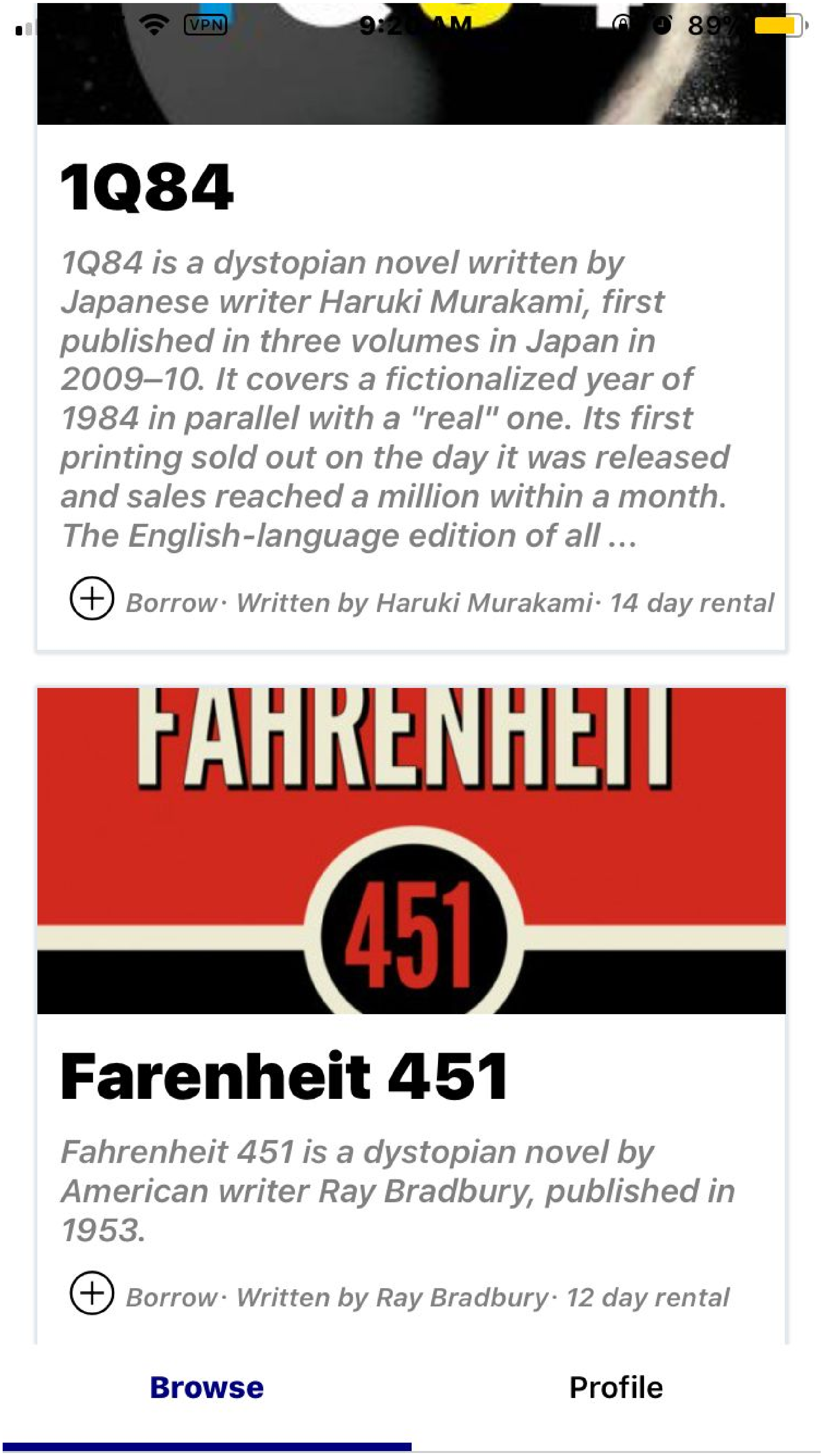


Centennial Library

Powered by Athenaeum

To Kill A Mockingbird

The story takes place during three years (1933–35) of the Great Depression in the fictional "tired old town" of Maycomb, Alabama, the seat of Maycomb County. It focuses on six-year-old Jean Louise Finch



1Q84

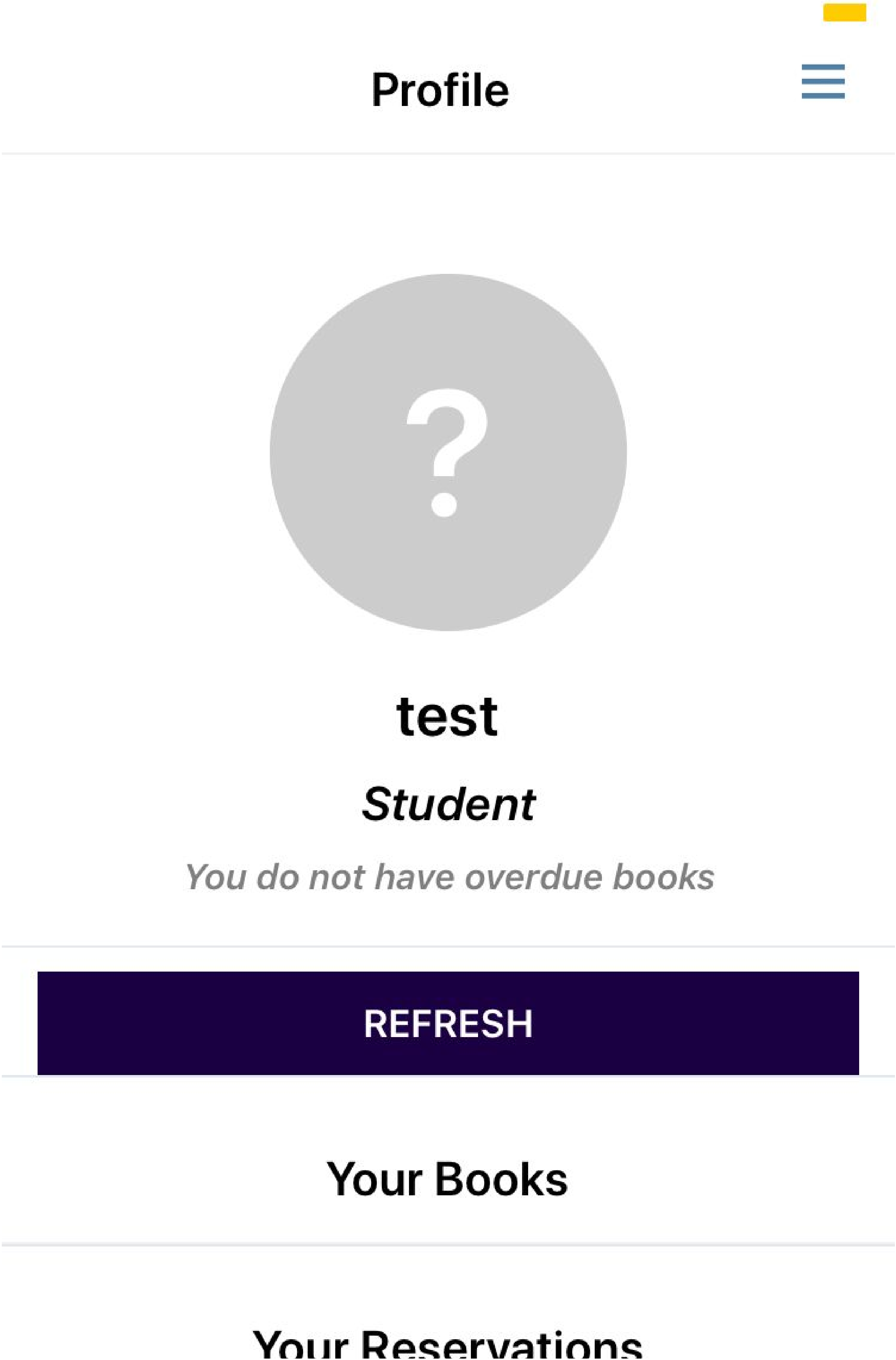
1Q84 is a dystopian novel written by Japanese writer Haruki Murakami, first published in three volumes in Japan in 2009–10. It covers a fictionalized year of 1984 in parallel with a "real" one. Its first printing sold out on the day it was released and sales reached a million within a month. The English-language edition of all ...

⊕ *Borrow · Written by Haruki Murakami · 14 day rental*

Farenheit 451

Fahrenheit 451 is a dystopian novel by American writer Ray Bradbury, published in 1953.

⊕ *Borrow · Written by Ray Bradbury · 12 day rental*



Profile

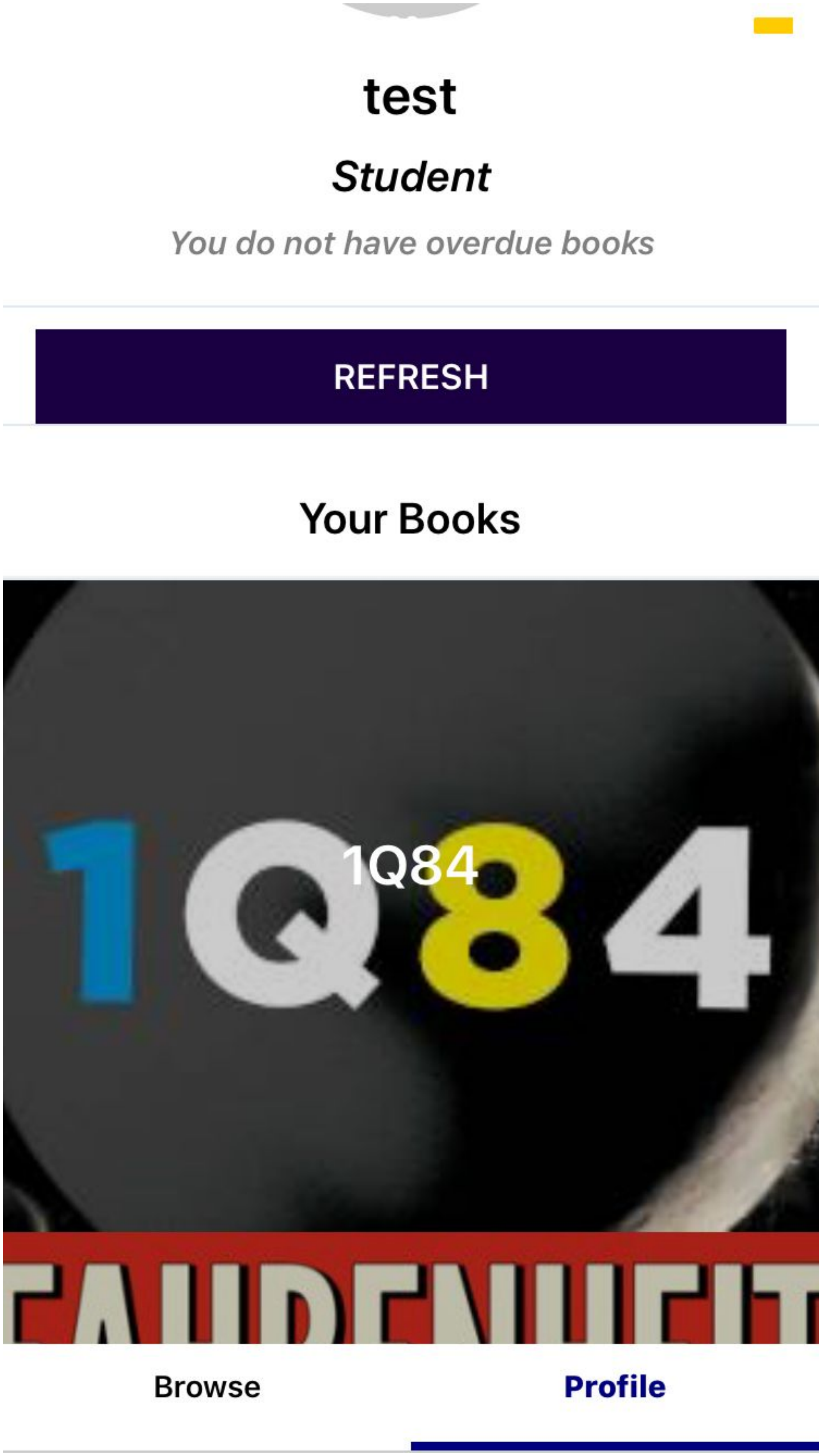
test

Student

You do not have overdue books

Your Books

Your Reservations



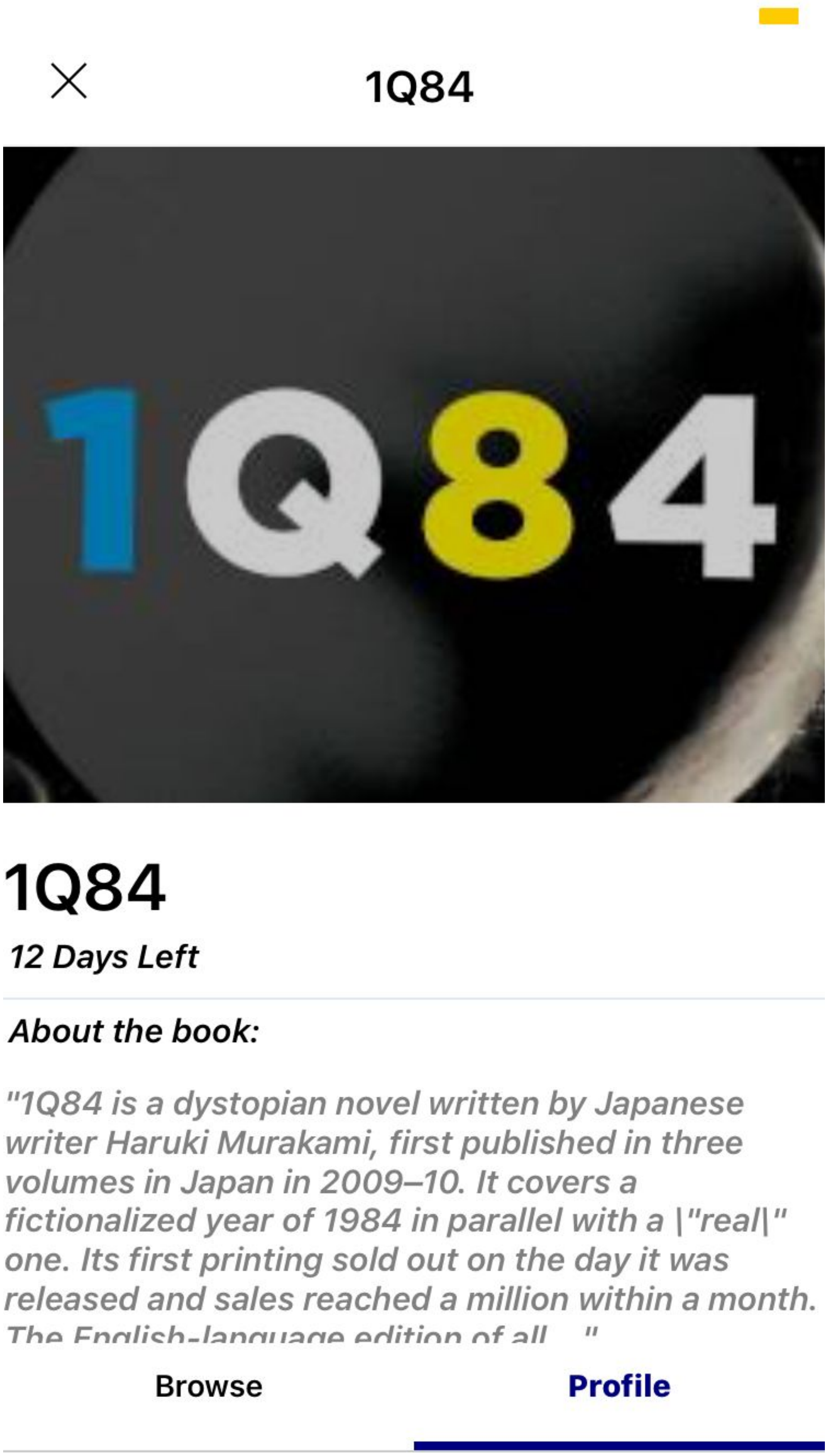
test

Student

You do not have overdue books

Your Books

1Q84



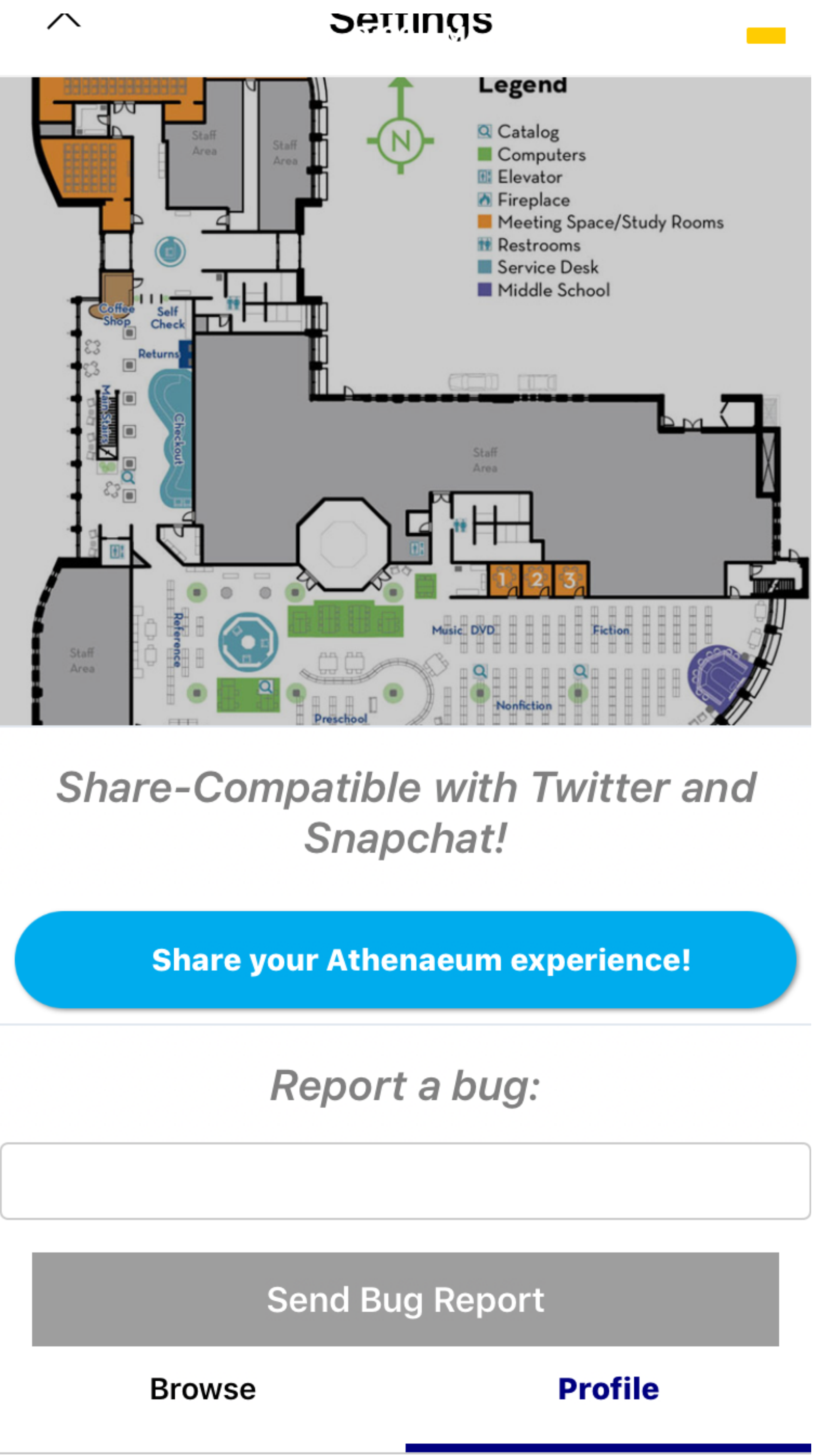
1Q84

1Q84

12 Days Left

About the book:

"1Q84 is a dystopian novel written by Japanese writer Haruki Murakami, first published in three volumes in Japan in 2009–10. It covers a fictionalized year of 1984 in parallel with a \"real\" one. Its first printing sold out on the day it was released and sales reached a million within a month. The English-language edition of all..."



Share-Compatible with Twitter and Snapchat!

Report a bug:

Source Code

```
import React from 'react';
import { View, Text } from 'react-native';
import Tabs from './tests/Nav'
import Main from './tests/Login'
import UserProfile from './tests/User'
import Page from './tests/Page'
import Settings from './tests/Settings'

export default class App extends React.Component {
 render() {
   return (
     <Main />
   );
 }
}
```

Browse.js

```
import React from 'react';
import { Text, View, ScrollView, StyleSheet, Button, Alert, RefreshControl } from
'react-native';
import { Card, Icon, Divider, Tile, Header } from 'react-native-elements';
import TopBar from './Header';
import { Manage } from './data/Manager';
var data = require('./data/TestData');

//Import all of the external files needed.

var manager = new Manage(data.data);
manager.updateItemsList();
//setup the data manager to enable data modifications

class BrowseCard extends React.Component{
 constructor(props){
   //inherit arguments that are passed into the card component
   super(props);
   this.state = {
     buttonConfirmed: false
   }
 }
 alert = (msg) => {
   console.log(msg)
 }
 sendAddData(){
   //this data sends functions to the manager from within the card itself.
   //the data that is stored in the card as props.
   manager.addItem(this.props.username, this.props.title)
 }
 sendReserveData(){
   manager.reserveItem(this.props.username, this.props.title)
 }
 render(){
   return(
     <View>
       <Card image={{uri: this.props.img}}>
         <View style={styles.border}>
           <Text style={styles.title}>{this.props.title}</Text>
         </View>
           <Text style={styles.description}>{this.props.isAvailable ? 'Currently
Available': 'Unavailable'}</Text>
           <Text style={styles.description}>{this.props.description}</Text>
           <View style={styles.bottomRow}>
             <View style={styles.buttonGroup}>
               <Icon
                 name={this.props.isAvailable ? 'plus': 'archive'}
                 type='evilicon'
                 color='black'
                 onPress={() => Alert.alert(
                   'Are you sure?',
                   'This book will be put in your '+ (this.props.isAvailable ?
'currently owned book list.' : 'reservations') + ' Refresh your profile to see
changes.',
                   [
                     {text: 'Cancel', onPress: ()=>console.log('Press cancelled')},
                     {text: 'Confirm', onPress: ()=>(
                       this.props.isAvailable ? this.sendAddData():
this.sendReserveData(),
                       console.log('check!')
                     )}
                   ]
                   //prompt the user to confirm or cancel the add request. if it is
accepted, the book will be added
                   //to the user's books or reservations depending on whether or not
the book is available
                 )}
                 size={30}
               />

             </View>
             <View style={styles.textGroup}>
               <Text style={styles.text}>{this.props.isAvailable ? 'Borrow' :
'Reserve'}</Text>
               <Text style={styles.text}>· {this.props.author}</Text>
               <Text style={styles.text}>· {this.props.days} day rental</Text>
             </View>
           </View>
       </Card>
     </View>
   )
 }
}

export default class Browse extends React.Component{
 constructor(props){
   super(props);
   this.state = {
     bookList: manager.renderBooks,
     refreshing: false
   }
 }
 createCards(){
   //this renders a card for every book available in the library list
data(testdata.js)
   return this.state.bookList.map((book, key)=>{
     //for every book in the library's data, it generates a visible card component
display the book info
     return(
       <View>
         {/*This creates a card object and transfers the data of the book item in the
data base
           into the card.*/}
         <BrowseCard username = {this.props.username} title={book.title} author =
{book.author} days={book.days} isAvailable={book.isAvailable}
description={book.description} img={book.img}/>
       </View>
     )
   })
 }
 fetchData(){
   this.setState({bookList: manager.renderBooks});
   //when this function is called, it updates the data if there are changes made
 }
 _onRefresh() {
   this.setState({refreshing: true});
   this.fetchData()
   this.setState({refreshing: false});
   //call the update data function when the refresh button is pressed
 }
 render(){
   //this sets up the main view that the user sees. The Top Bar is created for the
view
   return(
     <View style={styles.container}>
       <ScrollView refreshControl={
         <RefreshControl
           refreshing={this.state.refreshing}
           onRefresh={this._onRefresh.bind(this)}
         />
       }>
         <Tile
           imageSrc={{uri:
('https://i.pinimg.com/736x/67/01/a2/6701a2dd27613398be205baba6eb823c--homework-aesthe
tic-aesthetic-studying.jpg')}}
           featured
           title='Centennial Library'
           caption='Powered by Athenaeum'
         />
         <Divider />
         {
           this.createCards()
           //call the create cards function to show the full collection of books
         }
       </ScrollView>
     </View>
   )
 }
}

const styles = StyleSheet.create({
 container: {
   paddingVertical: 20
 },
 heading: {
   paddingTop: 15,
   fontSize: 20,
   fontStyle: 'italic',
   color: 'grey',
   alignSelf: 'flex-start',
   paddingLeft: 10
 },
 buttonText: {
   color: 'grey',
   fontSize: 12,
   fontStyle: 'italic',
   paddingTop: 6.5
 },
 textGroup: {
   alignSelf: 'flex-end',
   flexDirection: 'row'
 },
 buttonGroup: {
   alignItems: 'flex-start',
   flexDirection: 'row'
 },
 container: {
   backgroundColor: 'white'
 },
 buttons: {
   justifyContent: 'flex-end',
   flex: 10
 },
 bottomRow: {
   flexDirection: 'row'
 },
 title: {
   fontWeight: 'bold',
   fontSize: 30,
   paddingBottom: 8
 },
 description: {
   color: 'grey',
   fontSize: 15,
   paddingBottom: 8,
   fontStyle: 'italic'
 },
 text: {
   color: 'grey',
   fontSize: 12,
   fontStyle: 'italic',
   justifyContent: 'flex-end',
   paddingBottom: 5
 },
});
```

Login.js

```
import React from 'react';
import { View, Text, ScollView, StyleSheet } from 'react-native';
import t from 'tcomb-form-native';
import { Button, Icon, Tile, Divider } from 'react-native-elements'
import { StackNavigator } from 'react-navigation';
import Tabs from './Nav'

import { Manage } from './data/Manager';
var data = require('./data/TestData');

//Import all of the external files needed.
var manager = new Manage(data.data);
manager.updateItemsList();

var options = {
 auto: 'placeholders'
```

```
};

const Form = t.form.Form;

var Type = t.enums({
 Student: 'Student',
 Teacher: 'Teacher'
});

const SignInUser = t.struct({
 username: t.String,
 password: t.String,
});

const SignUpUser = t.struct({
 username: t.String,
 password: t.String,
 type: Type
});

class SignIn extends React.Component{
 handleSubmit = () => {
   const value = this._form.getValue(); // use that ref to get the form value
   //verify if the user is in the database and if the password matches that of the user in the databse
   let username = value.username
   username = username.replace(/\s/g, '')
   let verification = manager.verifyUser(username, value.password.replace(/\s/g, ''))
   console.log('login' + username)
   verification ? this.props.navigation.navigate('Nav', {username: manager.findUsername(username.toLowerCase())}): null
 }
 render(){
   return(
     <View style={styles.container}>
       <Tile
         imageSrc={{uri: ('https://i.imgur.com/yGrP2nX.jpg')}}
       />
       <Text style={styles.key}>For viewing purposes, the username is test, and the password is 'password'</Text>
       <Form
         ref={c => this._form = c} // assign a ref
         type={SignInUser}
         options={options}
       />
       <Button
         onPress={this.handleSubmit}//when the button is pressed, it calls the handleSubmit function
         title='Sign In'
         buttonStyle={styles.button}
       />
       <Divider />
       <Button
         onPress={()=>this.props.navigation.navigate('SignUp')}
         title='Need an account? Sign up here!'
       />
       <Divider />
     </View>
   )
 }

}
class SignUp extends React.Component{
 handleSubmit = () => {

   const value = this._form.getValue(); // use that ref to get the form value
   //verify if the user is in the database and if the password matches that of the
user in the databse
   let username = value.username.toLowerCase().replace(/\s/g, '')
   let password = value.password.toLowerCase().replace(/\s/g, '')
   let type = value.type
   console.log(username)
   manager.createUser(username, password, type)
   this.props.navigation.navigate('Nav', {username: username})

 }
 render(){
   return(
     <View style={styles.container}>
       <Tile
         imageSrc={{uri: ('https://i.imgur.com/yGrP2nX.jpg')}}
       />
       <Text style={styles.key}>Welcome!</Text>
       <Form
         ref={c => this._form = c} // assign a ref
         type={SignUpUser}
         options={options}
       />
       <Button
         onPress={this.handleSubmit}//when the button is pressed, it calls the
handleSubmit function
         title='Create Account!'
         buttonStyle={styles.button}
       />
       <Divider />
       <Button
         onPress={()=>this.props.navigation.navigate('SignIn')}
         title='Have an account? Sign in here!'s
       />
     </View>
   )
 }

}


const Main = StackNavigator (
 {
   SignUp: {screen: SignUp},
   SignIn: {screen: SignIn},
   Nav: {screen: Tabs}
 },
 {
   headerMode: 'none'
});
const styles = StyleSheet.create({
 container: {
   backgroundColor: 'white'
 },
 div: {
   paddingBottom: 20
 },
 key: {
   color: 'grey',
   fontStyle: 'italic',
 }
})
export default Main
```

Nav.js

```
import React from 'react';
import { Text, View } from 'react-native';
import { TabNavigator } from 'react-navigation';
var ScrollableTabView = require('react-native-scrollable-tab-view');
import TopBar from './Header';
import Browse from './Browse';
import UserProfile from './User';

//manages navigation between the profile and browse pages
export default class Tabs extends React.Component{
 render() {
   const { params } = this.props.navigation.state;
   const username = params ? params.username : null;
   //inherit the username string from the main component. this is passed down
   //to each page and component during navigation to allow data modifications
   return (
       <ScrollableTabView tabBarPosition='bottom' tabBarBackgroundColor='white'>
```

```
         <Browse tabLabel="Browse" username={username} /*pass the username into each
of the pages*//>
         <UserProfile tabLabel="Profile" username={username}/>
       </ScrollableTabView>
   );
 }
}
```

Page.js

```
import React from 'react';
import { Text, View, StyleSheet, ScrollView, RefreshControl, Alert, Share } from
'react-native';
import { Card, Avatar, Button , Icon, Divider, Tile, Header, SocialIcon } from
'react-native-elements';
var ScrollableTabView = require('react-native-scrollable-tab-view');
import PTRView from 'react-native-pull-to-refresh';
import { StackNavigator } from 'react-navigation';
import { Manage } from './data/Manager';
var data = require('./data/TestData');

var user = 'test';
var manager = new Manage(data.data);
manager.updateItemsList();

class BackButton extends React.Component{
 render(){
   return(
     <Icon
       name='close'
       type='evilicon'
       color='black'
       onPress={() => this.props.navigation.goBack()}
       //when this button is pressed it goes back to the previous page
       size={30}
     />
   )
 }
}


export default class Page extends React.Component{
 onClick(title) {
   Share.share({
     message: 'Currently Reading ' + title + '!',
     url: manager.data.books[title].img,
     title: 'Books by Athenaeum'
   }, {
```

```
   // Android only:
     dialogTitle: 'with love, from Centennial High School, Corona',
   // iOS only:

   })
 }
 sendReturnData(username, title, type){
   //check what type of list the book belongs in.
   //if the page is rendered for a book in the user's book list, it uses the
removeItem function
   //if it is a reservation, the component's function accesses the remove reservation
function
   if(type==='books'){
     manager.removeItem(username, title)
   }
   if(type==='reservations'){
     manager.removeReservation(username, title)
   }
   //creates local function connected to the manager to send the card's prop data to
the Manager
   //in order to  modify the book and user's data.
   this.props.navigation.goBack()
 }
 //write a return JSON function

 render(){
   //import parameters that are passed into the component
   const { params } = this.props.navigation.state;
   const title = params ? params.title : null;
   const description = params ? params.description : null;
   const img = params ? params.img : null;
   const username = params ? params.username : null;
   const type = params ? params.type : null;
   //inherits all of the data passed into the page
   return(
     <ScrollView style={styles.container}>
       <Header
         statusBarProps={{ barStyle: 'light-content' }}
         leftComponent={<BackButton navigation = {this.props.navigation}/>}
         centerComponent={{ text: JSON.stringify(title).replace(/(^"|"$)/g, ''),
style: { color: 'black', fontSize: 20} }}
         backgroundColor = 'white'
       />
       <Tile
         imageSrc={{uri: (JSON.stringify(img).replace(/(^"|"$)/g, ''))}}
         featured
       />
       <Text style={styles.title}>{JSON.stringify(title).replace(/(^"|"$)/g,
'')}</Text>
       <Text style={styles.days}>12 Days Left</Text>
       <Divider />
       <Text style={styles.days}>About the book:</Text>
       <Text style={styles.description}>{JSON.stringify(description)}</Text>

       <SocialIcon
         title='Share your Athenaeum experience!'
         button
         type='twitter'
         onPress={()=>this.onClick(title)}
       />

       <Button
         icon={
           <Icon
             name='arrow-right'
             size={15}
             color='white'

           />
         }
         title='Return this book'
         onPress={() => Alert.alert(
           'Return this?',
           'Are you sure you want to return this book?.',
           [
             {text: 'Cancel', onPress: ()=>console.log('Press cancelled')},
             {text: 'Confirm', onPress:
()=>this.sendReturnData(JSON.stringify(username).replace(/(^"|"$)/g, ''),
JSON.stringify(title).replace(/(^"|"$)/g, ''),
JSON.stringify(type).replace(/(^"|"$)/g, ''))}
           ]
         )}
       />
     </ScrollView>
   )
 }
}

const styles = StyleSheet.create({
 container: {
   backgroundColor: 'white'
 },
 title: {
   paddingTop: 20,
   fontSize: 30,
 },
 days: {
   paddingTop: 5,
   paddingBottom: 10,
   paddingLeft: 2,
   fontSize: 15,
   fontStyle: 'italic'
 },
 description: {
   paddingTop: 5,
   color: 'grey',
   fontSize: 15,
   paddingBottom: 20,
   fontStyle: 'italic'
 },
 button: {
   color: 'green'
 }

})
```

Settings.js

```
import React from 'react';
import { View, Text, Share, StyleSheet, ScrollView } from 'react-native';
import { Header, Tile, Divider, SocialIcon, Button, Icon } from
'react-native-elements'
var t = require('tcomb-form-native');

var Form = t.form.Form;

var Report = t.struct({
 Bug: t.String,              // a required string        // a boolean
});

var options = {
 auto: 'none'
};

class BackButton extends React.Component{
 render(){
   return(
     <Icon
       name='close'
       type='evilicon'
       color='black'
       onPress={() => this.props.navigation.goBack()}
       size={30}
     />
   )
 }
}
```

```
export default class Settings extends React.Component{

 onClick() {
   Share.share({
     message: 'Sent from the Athenaeum App by Centennial FBLA. See you at SLC!',
     url: 'http://www.cafbla.org/pages/CAFBLA',
     title: 'Wow, did you see that?'
   }, {
   // Android only:
     dialogTitle: 'with love, from Centennial High School, Corona',
   // iOS only:

   })
 }
 handleSubmit = () => {
   const value = this._form.getValue(); // use that ref to get the form value
   console.log(value)
 }
 render(){
   return(
     <ScrollView style={styles.container}>
       <Header
         statusBarProps={{ barStyle: 'light-content' }}
         leftComponent={<BackButton navigation = {this.props.navigation}/>}
         centerComponent={{ text: 'Settings', style: { color: 'black', fontSize: 20}
}}
         backgroundColor = 'white'
       />
       <Tile
         imageSrc={{uri:
('http://elmhurstpubliclibrary.org/lib/wp-content/uploads/First-Floor-Map.jpg')}}
         featured
       />
       <Divider />
       <Text style={styles.subtitle}>Share-Compatible with Twitter and
Snapchat!</Text>
       <SocialIcon
         title='Share your Athenaeum experience!'
         button
         type='twitter'
         onPress={this.onClick.bind(this)}
       />
       <Divider />
       <Text style={styles.subtitle}>Report a bug:</Text>
       <Form
         options={options}
         ref={c => this._form = c}
         type={Report}
       />
       <Button title='Send Bug Report' onPress={this.handleSubmit}/>
     </ScrollView>
   )
 }
}
const styles = StyleSheet.create({
 container: {
   backgroundColor: 'white'
 },
 subtitle: {
   paddingTop: 15,
   paddingBottom: 15,
   textAlign: 'center',
   fontSize: 20,
   fontStyle: 'italic',
   color: 'grey'

 }
})
```

User.js

```
import React from 'react';
import { Text, View, StyleSheet, ScrollView, RefreshControl, Alert } from
'react-native';
import { Card, Avatar, Button , Icon, Divider, Tile, Overlay, Header } from
'react-native-elements';
var ScrollableTabView = require('react-native-scrollable-tab-view');
import Page from './Page'
import TopBar from './Header'
import Settings from './Settings'
import PTRView from 'react-native-pull-to-refresh';
import { StackNavigator } from 'react-navigation';
import { Manage } from './data/Manager';
var data = require('./data/TestData');
//import required packages
var user = 'test';
var manager = new Manage(data.data);
manager.updateItemsList();
//setup the data managers

class Tab extends React.Component{
 render(){
   return(
     <View>
       <Tile
         imageSrc={{uri: (this.props.img)}}
         title={this.props.title}
         featured
         onPress={() => this.props.navigation.navigate('Page', {
           title: this.props.title,
           img: this.props.img,
           description: this.props.description,
           type: this.props.type,
           username: this.props.username
           //pass all of the book's data and the active user's name into rendering
info page
         }))}
       />

     </View>
   )
 }
}
class BookView extends React.Component{
 //pass props
 constructor(props) {
     super(props);
     //state is used to manage changing data. the state of this Component
     //is used to access the books and reservations of the user
     this.state = {
       books: manager.returnUserBooks(this.props.username, 'books'),
       reservations: manager.returnUserBooks(this.props.username, 'reservations'),
       resfreshing: false
       //place the book data into the constructor. the state argument
       // of the component allows the data to be updated and changed
     };
   }
 createCards(type){
   return this.state[type].map((book)=>{
     //render a component for each book
     return(
       <Tab type={type==='books'?'books':'reservations'}
navigation={this.props.navigation} img={book.img} title={book.title}
description={book.description} username={this.props.username}/>
     )
     //pass the arguments into the tab, such as the book's image link, the book's
title, etc
   })
 }
 fetchData(){
   this.setState({books: manager.returnUserBooks(this.props.username, 'books')});
   this.setState({reservations: manager.returnUserBooks(this.props.username,
'reservations')});
   //when this function is called, it updates the data if there are changes made
 }
 _onRefresh() {
   this.setState({refreshing: true});
   this.fetchData()
   this.setState({refreshing: false});
   //call the update data function when the refresh button is pressed
 }
 render(){
   return(
     <View style={styles.bookView}>
       <Button
         title='REFRESH'
         onPress={this._onRefresh.bind(this)}
         buttonStyle={styles.button}
       />
       <Divider />
       <Header
         statusBarProps={{ barStyle: 'light-content' }}
         centerComponent={{ text: 'Your Books', style: { color: 'black', fontSize: 20}
}}
         backgroundColor = 'white'
       />
       <Divider />
       {this.createCards('books')}
       <Header
         statusBarProps={{ barStyle: 'light-content' }}
         centerComponent={{ text: 'Your Reservations', style: { color: 'black',
fontSize: 20} }}
         backgroundColor = 'white'
       />
       {this.createCards('reservations')}
     </View>
   )
 }
}
class SettingsButton extends React.Component{
 render(){
   return(
     <Icon
       name='navicon'
       type='evilicon'
       color='#517fa4'
       onPress={() => this.props.navigation.navigate('Settings', {
         navigation: this.props.navigation
       })}
       size={30}
     />
   )
 }
}
class Profile extends React.Component{
 render(){
   const { navigation, screenProps } = this.props
   let username = this.props.screenProps.user.username
   console.log('username'+username)
   return(
     <ScrollView style={styles.container}>
       <Header
         statusBarProps={{ barStyle: 'light-content' }}
         rightComponent={<SettingsButton navigation={this.props.navigation}/>}
         centerComponent={{ text: 'Profile', style: { color: 'black', fontSize: 20} }}
         backgroundColor = 'white'
       />
       <View style={styles.profileBar}>
         <Avatar
           xlarge
           rounded
           title="?"
           onPress={() => console.log("Works!")}
           activeOpacity={0.7}
         />
         <Text style={styles.profileUsername}>{username.charAt(0).toUpperCase() +
username.slice(1)}</Text>
         <Text
style={styles.profileSubtitle}>{manager.data.users[username].type}</Text>
         {
           //the overdue warning checks the user's object in the database
           //and checks to see overdue is set to false
         }
         <Text style={styles.overdueWarning}>{manager.data.users[username].overdue ?
'You have books overdue.' : 'You do not have overdue books'}</Text>
       </View>
       <Divider />
       <BookView navigation={this.props.navigation} username={username}/>

     </ScrollView>
   )
 }
}

const Stack = StackNavigator (
 {
   Profile: {
     screen: Profile
   },
   Page: {screen: Page},
   Settings: {screen: Settings}
 },
 {
   headerMode: 'none',
 }
);
export default class UserProfile extends React.Component{
 render(){
   const screenProps = {
     user: {
       username: this.props.username,
     },
   }
   return(
     <Stack screenProps={screenProps}/>
   )
 }
}

const styles = StyleSheet.create({
 container: {
   backgroundColor: 'white'
 },
 wrapper: {
   alignItems: 'center'
 },
 button: {
   backgroundColor: '#1b0042'
 },
 bottomText: {
   flexDirection: 'row',
   paddingTop: 20
 },
 bookView: {
   paddingTop: 10
 },
 profileBar: {
   paddingTop: 50,
   alignItems: 'center',
   paddingBottom: 20
 },
 profileUsername: {
   paddingTop: 20,
   fontSize: 25
 },
 profileSubtitle: {
   paddingTop: 10,
   fontSize: 20,
   fontStyle: 'italic'
 },
 overdueWarning: {
   paddingTop: 10,
   fontSize: 15,
   fontStyle: 'italic',
   color: 'grey'
 }

})
```