AsyncStorageFX.js

```
import AsyncStorage from '@react-native-async-storage/async-storage';

function setData(key, value) {
  AsyncStorage.setItem(`@${key}`, value)
  .then(() => {
  })
  .catch(e => console.log(e));
}

async function getData(key) {
  try {
    const data = await AsyncStorage.getItem(`@${key}`);
    return JSON.parse(data);
  } catch (e) {
    console.log(e)
    return [];
  }
}

async function checkInKeys(key) {
  try {
    const keys = await AsyncStorage.getAllKeys()
    return keys.includes(`@${key}`);
  } catch (e) {
    console.log(e);
    return false;
  }
}

export {setData, getData, checkInKeys};
```

Classroom.js

```
import {ScrollView, StyleSheet, Text, ToastAndroid, View} from "react-native";
import {Dropdown} from "react-native-element-dropdown";
import {List} from "../Components/List";
import MagnifierButton from "../Components/SearchButton";

export function Classroom() {

  return (
      <View style={styles.container}>
        {/*{isLoading ? <LoadingPopup visible={isLoading}/> : null}*/}
        {/*{searching ? <LoadingPopup visible={searching}/> : null}*/}
        <View style={styles.selectorContainer}>
          <Dropdown style={styles.selectorView} containerStyle={styles.selectorList} data={timeslots} labelField="label"
                    valueField="value"
                    onChange={(item) => {
                      setSelectedTimeSlot(item.value)
                    }}
                    value={selectedTimeSlot}
                    mode={"modal"}
                    autoScroll={false}
          />
          <Dropdown style={styles.selectorView} containerStyle={styles.selectorList} data={rooms} labelField="label"
                    valueField="value"
                    onChange={(item) => {
                      setSelectedRoom(item.value)
                    }}
                    value={selectedRoom}
                    search={true}
                    mode={"modal"}
                    autoScroll={false}
          />
          <MagnifierButton onPress={searchClassRoom}/>
        </View>
        <Text style={styles.label}>Classes</Text>
        <ScrollView style={styles.scrollView}>
          {resultingData.length === 0 ? <Text style={{fontSize: 100, alignSelf: 'center'}}>🙅🙅</Text> :
           <List data={resultingData} type={"Classroom"}/>}
        </ScrollView>
      </View>);
}

const styles = StyleSheet.create({
  container: {
    flex: 1, backgroundColor: '#fff', alignItems: 'center', width: '100%',
  }, scrollView: {
    width: '80%', margin: 20,
  }, label: {
    fontSize: 18, fontWeight: 'bold', marginVertical: 10, alignSelf: 'flex-start', marginLeft: '6%',
  },
  selectorView: {
    width: '40%',
    padding: 10,
    marginVertical: 10,
    borderWidth: 0.3,
    borderColor: '#000',
    borderRadius: 5,
  },
  selectorList: {
    width: '200%',
    padding: 10,
    marginVertical: 10,
    borderWidth: 0.3,
    borderColor: '#000',
    borderRadius: 5,

  },
  selectorContainer: {
    flexDirection: "row",
    justifyContent: "space-around",
    alignItems: "center",
    width: "100%",
  }
});
```

FBFunctions.js

```
import {db} from "../Config/FirebaseConfig";
import {doc, collection, getDocs, getDoc} from "firebase/firestore";
import {setData} from "./AsyncStorageFX";

async function getAllTeachersData() {
  const teacherNames = [];
  await getDocs(collection(db, "TeacherSchedule")).then((querySnapshot) => {
    querySnapshot.forEach((doc) => {
      teacherNames.push({label: doc.id, value: doc.id});
      setData(doc.id, JSON.stringify(doc.data().data));
    })
  });
  setData("teacherNames", JSON.stringify(teacherNames));
  return teacherNames;
}

async function getTimeslots() {
  const timeslots = [];
  const docRef = doc(db, "Timetable", "BSE 6A");
  const docSnap = await getDoc(docRef);
  if (docSnap.exists()) {
    const data = docSnap.data();
    const oneDay = Object.keys(data)[0];
    Object.keys(data[oneDay]).forEach((key) => {
      timeslots.push({label: key, value: key});
    })
  } else {
    console.log("No such document!");
  }
  timeslots.sort(function (a, b) {
    const timeA = parseInt(a.label.split(":")[0]);
    const timeB = parseInt(b.label.split(":")[0]);
    return timeA - timeB;
  });
  setData("timeslots", JSON.stringify(timeslots));
  return timeslots;
}

export {getAllTeachersData, getTimeslots};
```

```
function FindClassRoomDetails(fullSchedule,roomNo,slot) {
  let data = [];
  fullSchedule=fullSchedule[0];
  console.log(JSON.stringify(fullSchedule))
  if(roomNo === undefined || roomNo === null || roomNo === ""){
    return data;
  }
  for (const className in fullSchedule) {
    for (const classObj of fullSchedule[className]) {
      if (classObj[slot] &&
          classObj[slot]["classRoom"].includes(roomNo)) {
        data.push({className, day: classObj["Day"], time: slot, subject: classObj[slot]["subject"], teacher: classObj[slot]["teacher"], classRoom: classObj[slot]["classRoom"]})
      }
    }
  }
  data = [...new Set(data)]
  const weekdays = ['Sunday', 'Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday'];
  data.sort((a, b) => {
    return weekdays.indexOf(a.day) - weekdays.indexOf(b.day);
  });
  console.log(fullSchedule["BSE 6A"][2]["08:00 to 09:30"]["day"])
  return data;
}

function extractAllSubjects(data) {
  let subjects = [];
  Object.values(data).forEach((week) => {
    week.forEach((day) => {
      Object.values(day).forEach((slot) => {
        if (slot !== null && typeof slot === "object" && "subject" in slot) {
          subjects.push(slot.subject);
        }
      });
    });
  });
  subjects = [...new Set(subjects)]
  return subjects;
}

function extractAllTeachers(data) {
  let teachers = [];
  Object.values(data).forEach((week) => {
    week.forEach((day) => {
      Object.values(day).forEach((slot) => {
        if (slot !== null && typeof slot === "object" && "teacher" in slot) {
          teachers.push(slot.teacher);
        }
      });
    });
  });
  teachers = [...new Set(teachers)]
  return teachers;
}

function noOfClasses(JsonData, reqDay) {
  let no = 0;
  Object.keys(JsonData).map((key, index) => {
    JsonData[key].forEach((day) => {
      if (day.Day === reqDay) {
        Object.values(day).forEach((slot) => {
          if (slot !== null && typeof slot === "object" && "teacher" in slot)
            no++;
        });
      }
    })
  })
  console.log(no)
}

function extractAllClassRooms(data) {
  data=data[0];
  let classRooms = [];
  Object.keys(data).map((ClassNameSemesterSection) => {
    for (const daySchedule of data[ClassNameSemesterSection]) {
        for (const timeSlot in daySchedule) {
        if (daySchedule[timeSlot] !== null && typeof daySchedule[timeSlot] === 'object') {
          const room = daySchedule[timeSlot].classRoom;
          if (room && !classRooms.includes(room)) {
            classRooms.push(room);
          }
        }
      }
    }
  });
  classRooms.sort((a, b) => {
    if (a > b) {
      return 1;
    } else if (a < b) {
      return -1;
    } else {
      return 0;
    }
  });

  classRooms= classRooms.map((classRoom) => {
    return {label: classRoom, value: classRoom};
  });
    return classRooms;
}

export {
  FindClassRoomDetails,
  extractAllSubjects,
  extractAllTeachers,
  noOfClasses,
  extractAllClassRooms,
};
```

List.js

```
import {Text, View} from "react-native";

export function List({data, type}) {
  if (data.length === 0) {
    return (<Text>No Record</Text>)
  }
  function renderItemClassroomBased(item) {
    return (
        <View style={{
          marginVertical: 10,
          borderColor: 'black',
          borderRadius: 10,
          borderWidth: 1,
          overflow: 'hidden',
        }}>
          <View style={{
            flexDirection: 'row', justifyContent: 'space-evenly', backgroundColor: 'rgb(2, 201, 208)',
            padding: 10
          }}>
            <Text>{item.className}</Text>
            <Text>{item.day}</Text>
          </View>
          <View style={{padding: 10}}>
            <Text>{item.subject}</Text>
            <Text>{item.teacher.trim()}</Text>
            <Text>{item.classRoom}</Text>
            <Text>{item.time}</Text>
          </View>
        </View>
    );
  }

  function renderItemTeacherBased(item) {
    return (
        <View style={{
          marginVertical: 10,
          borderColor: 'black',
          borderRadius: 10,
          borderWidth: 1,
          overflow: 'hidden',
        }}>
          <View style={{
            flexDirection: 'row', justifyContent: 'space-evenly', backgroundColor: 'rgb(2, 201, 208)',
            padding: 10
          }}>
            <Text>{item.className}</Text>
            <Text>{item.day}</Text>
          </View>
          <View style={{padding: 10}}>
            <Text>{item.subject}</Text>
            <Text>{item.classRoom}</Text>
            <Text>{item.timeSlot}</Text>
          </View>
        </View>
    );
  }

  if (type === "Classroom") {
    return data.map((item) => renderItemClassroomBased(item))
  }
  else if (type === "Teacher") {
    return data.map((item) => renderItemTeacherBased(item))
  }else if (type === "Timetable") {
    return <Text>Not Yet</Text>
  }
}
```

Loading.js

```
import React from 'react';
import { View, Modal, ActivityIndicator, StyleSheet } from 'react-native';

const LoadingPopup = ({ visible }) => {
  return (
      <Modal
          animationType="fade"
          transparent={true}
          visible={visible}
      >
        <View style={styles.container}>
          <View style={styles.popup}>
            <ActivityIndicator size="large" color="#0000ff" />
          </View>
        </View>
      </Modal>
  );
};

const styles = StyleSheet.create({
  container: {
    flex: 1,
    backgroundColor: 'rgba(0, 0, 0, 0.5)',
    justifyContent: 'center',
    alignItems: 'center',
  },
  popup: {
    backgroundColor: '#fff',
    borderRadius: 10,
    padding: 20,
    alignItems: 'center',
    justifyContent: 'center',
  },
});

export default LoadingPopup;
```

SearchButton.js

```
import React from 'react';
import { TouchableOpacity } from 'react-native';
import { MaterialIcons } from '@expo/vector-icons';

const MagnifierButton = ({ onPress }) => {
  return (
      <TouchableOpacity style={{alignSelf:'center'}} onPress={onPress}>
        <MaterialIcons name="search" size={35} color="black" />
      </TouchableOpacity>
  );
};


export default MagnifierButton;
```

SqliteFX.js

```
import {collection, getDocs} from "firebase/firestore";
import * as SQLite from 'expo-sqlite';
import {db} from "../Config/FirebaseConfig";
import {ToastAndroid} from "react-native";
import {setData} from "./AsyncStorageFX";

const sqlDatabase = SQLite.openDatabase('timetable.db');

async function fetchAndSaveDataFromFB() {
  sqlDatabase.transaction((tx) => {
    tx.executeSql('CREATE TABLE IF NOT EXISTS Timetable (id INTEGER PRIMARY KEY,className TEXT, day TEXT, startTime TEXT, classRoom TEXT, subject TEXT, teacher TEXT);', [],
        () => ToastAndroid.show("Timetable table created", ToastAndroid.SHORT));
  });
  await getDocs(collection(db, "Timetable")).then((querySnapshot) => {
    sqlDatabase.transaction((tx) => {
      querySnapshot.forEach((docSnapshot) => {
        const docData = docSnapshot.data();
        Object.entries(docData).forEach(([day, timeSlots]) => {
          Object.entries(timeSlots).forEach(([startTime, timeSlot]) => {
            if (timeSlot === null || typeof timeSlot !== 'object') return;
            const {classRoom, subject, teacher} = timeSlot;
            tx.executeSql(
                'INSERT INTO Timetable (className, day, startTime, classRoom, subject, teacher) VALUES (?, ?, ?, ?, ?, ?)',
                [docSnapshot.id, day, startTime, classRoom, subject, teacher]);
          });
        });
      });
    });
  }).catch((e) => {
    console.log(e);
  });
}

async function checkIfTableExists() {
  let tableExists = false;
  try {
    await sqlDatabase.transaction((tx) => {
      tx.executeSql('SELECT * FROM Timetable', [], (tx, results) => {
        if (results.rows.length > 0) {
          tableExists = true;
        }
      });
    });
  } catch (e) {
    console.log(e);
  }
  return tableExists;
}

async function getClassRoomNamesFromSQL() {
  if (!(await checkIfTableExists())) {
    await fetchAndSaveDataFromFB();
  }
  try {
    let classRoomNames = [];
    await sqlDatabase.transaction((tx) => {
      tx.executeSql('SELECT DISTINCT classRoom FROM Timetable', [], (tx, results) => {
        for (let i = 0; i < results.rows.length; i++) {
          classRoomNames.push(results.rows.item(i).classRoom);
        }
        classRoomNames.sort();
        classRoomNames = classRoomNames.map((classRoomName) => {
          return {label: classRoomName, value: classRoomName};
        });
        setData("classRoomNames", classRoomNames);
        return classRoomNames;
      });
    });
  } catch (e) {
    return [];
  }
}

export {fetchAndSaveDataFromFB, getClassRoomNamesFromSQL};
```

Teachers.js

```
import {ActivityIndicator, ScrollView, StyleSheet, Text, ToastAndroid, View} from "react-native";
import {useEffect, useState} from "react";
import {getAllTeachersData} from "../../BackEnd/FBFunctions";
import {Dropdown} from "react-native-element-dropdown";
import {List} from "../Components/List";
import {checkInKeys, getData} from "../../BackEnd/AsyncStorageFX";

export function Teachers() {
  useEffect(() => {
    checkInKeys("teachers").then((res) => {
      if (!res) {
        let promise = getDataNotFoundInAsyncStorage();
      } else {
        let promise = getDataFoundInAsyncStorage();
      }
    }).catch(() => {
      let promise = getDataNotFoundInAsyncStorage();
    })
    return () => {
      console.log("Teachers Screen Unmounted")
    }
  }, []);

  async function getDataNotFoundInAsyncStorage() {
    try {
      const teacherNs = await getAllTeachersData()
      setTeachersNames(teacherNs);
      ToastAndroid.show("Teachers Data Updated", ToastAndroid.SHORT);
      setLoading(false);
    } catch (e) {
      setTeachersNames([{label: e, value: e}])
      setLoading(false);
    }
  }

  async function getDataFoundInAsyncStorage() {
    try {
      const teacherNs = await getData("teachers");
      setTeachersNames(teacherNs);
      ToastAndroid.show("Teachers Data Updated", ToastAndroid.SHORT);
      setLoading(false);
    } catch (e) {
      setTeachersNames([{label: e, value: e}])
      setLoading(false);
    }
  }

  const [teachersNames, setTeachersNames] = useState([]);
  const [loading, setLoading] = useState(true);
  const [slot, setSlot] = useState(null);
  const [selectedTeacherData, setSelectedTeacherData] = useState([]);
  return (
      <View style={styles.container}>
        <Dropdown style={styles.slotSelector} data={teachersNames} labelField="label" valueField="value"
                  onChange={(item) => {
                    setSlot(item);
                    getData(item.label).then((data) => {
                      setSelectedTeacherData(data);
                    }).catch((e) => {
                      console.log(e);
                    });
                  }}
                  placeholder={"Select a teacher"}
                  value={slot}
                  search={true}
                  searchPlaceholder="Teacher name"
                  autoScroll={false}
        />
        <Text style={styles.label}>Teacher's Schedule</Text>
        <ScrollView style={styles.scrollView}>
          {selectedTeacherData.length === 0 ? <Text style={{fontSize: 100, alignSelf: 'center'}}>👨‍👦</Text> :
           (<>
                 <List data={selectedTeacherData} type={"Teacher"}/>
               </>
           )
          }
        </ScrollView>
        {loading ? <ActivityIndicator size="large" color="#0000ff"/> : null}
      </View>
  )
```

```
}

const styles = StyleSheet.create({
  container: {
    flex: 1, backgroundColor: '#fff', alignItems: 'center', width: '100%',
  }, scrollView: {
    width: '80%', margin: 20,
  }, label: {
    fontSize: 18, fontWeight: 'bold', marginTop: 10, alignSelf: 'flex-start', marginLeft: '6%',
  },
  slotSelector: {
    width: '90%',
    padding: 10,
    marginTop: 10,
    borderWidth: 0.3,
    borderColor: '#000',
    borderRadius: 5,
  },
});
```