DAFTAR LAMPIRAN

Lampiran-1. *App.js*

```
import React, { useState, useEffect } from 'react';
import { StyleSheet, ScrollView, View, TextInput, Text } from 'react-native';
import { NavigationContainer, useIsFocused } from '@react-navigation/native';
import { createNativeStackNavigator } from '@react-navigation/native-stack';
import MenuScreen from './components/MenuScreen';
import LoginScreen from './components/LoginScreen';
import KategoriScreen from './components/KategoriScreen';
import KategoriAddScreen from './components/KategoriAddScreen';
import KategoriDetailScreen from './components/KategoriDetailScreen';
import KategoriEditScreen from './components/KategoriEditScreen';
import KataTidakBakuScreen from './components/KataTidakBakuScreen';
import KataTidakBakuDetailsScreen from './components/KataTidakBakuDetailsScreen';
import KataTidakBakuAddScreen from './components/KataTidakBakuAddScreen';
import KataTidakBakuEditScreen from './components/KataTidakBakuEditScreen';
import Database from './Database';
import { Button, Card } from 'react-native-elements';
import Icon from 'react-native-vector-icons/FontAwesome';
const db = new Database();
function HomeScreen({ route, navigation }) {
  let [kalimat, setKalimat] = useState('');
  let [kalimatHasil, setKalimatHasil] = useState([]);
  let [hasil, setHasil] = useState([]);
  let [hasilKataBaku, setHasilKataBaku] = useState([]);
  let [hasilKataTidakTerdaftar, setHasilKataTidakTerdaftar] = useState([]);
  navigation.setOptions({
    headerRight: () => (
      <Button
        title="*"
        buttonStyle={{ padding: 0, backgroundColor: 'transparent' }}
        icon={
          <Icon
            name="sign-in"
            size={15}
          // color="white"
          />
        }
        onPress={() => {
          navigation.navigate('Login');
        }}
      />
    )
  });
  let cekKataTidakBaku = () => {
    if (kalimat !== '') {
      setKalimat('');
      setKalimatHasil([]);
      setHasil([]);
      let kata = kalimat.split(/[ .:;?!~,`"&|()<>{}\[\]\r\n/\\]+/);
```

```
    //clear data pada state
    let arrKalimat = [];
    let arrKataTidakTerdaftar = [];
    let arrHasil = [];
    let arrHasilKataBaku = [];
    let id = 0;
    let id2 = 0;
    let id3 = 0;
    let kataRemove = [];
    kata.forEach((v, i) => {
      db.findKataTidakBaku(v).then((data) => {
        if (data == '') {
          arrKalimat.push(v + " ")
          arrKataTidakTerdaftar.push(<Text>{id2 = id2 + 1}. <Text style={{ fontWeight:
'bold', color: 'blue' }}>{v} </Text></Text>)
        } else {
          //menampilkan style pada kalimat yang
          arrKalimat.push(<Text style={styles.highLight}>{v} </Text>)
          arrHasil.push(
            <View>
              <Text>{id = id + 1}. <Text style={{ fontWeight: 'bold', color: 'red'
}}>{data.ktb}</Text> = {data.kb} [{data.Kategori}]</Text>
            </View>
          )
          arrHasilKataBaku.push(
            <View>
              <Text>{id3 = id3 + 1}. <Text style={{ fontWeight: 'bold', color: 'green'
}}>{data.kb}</Text> [{data.Kategori}]</Text>
            </View>
          )
        }
      })
      db.findKataTidakBaku2(kata[i], kata[i + 1]).then((data) => {
        if (data !== '') {
          //menampilkan style pada kalimat yang
          arrKalimat.push(<Text style={styles.highLight}>{kata[i] + " " + kata[i + 1]} </Text>)
          arrHasil.push(
            <View>
              <Text>{id = id + 1}. <Text style={{ fontWeight: 'bold', color: 'red'
}}>{data.ktb}</Text> = {data.kb} [{data.Kategori}]</Text>
            </View>
          )
          arrHasilKataBaku.push(
            <View>
              <Text>{id3 = id3 + 1}. <Text style={{ fontWeight: 'bold', color: 'green'
}}>{data.kb}</Text> [{data.Kategori}]</Text>
            </View>
          )
          // kirim index kata yang akan di hapus
          kataRemove.push(i);
        }
      })
    })
    let time = kata.length / 5;
    (time <= 0) ? time = 1000 : time = time * 1000;
```

```
    setTimeout(() => {
      //hapus 2 kata dalam kalimat
      kataRemove.forEach(v => {
        arrKalimat.splice(v, 1);
        arrKalimat.splice(v + 1, 1);
      })
      setKalimatHasil(arrKalimat);
      setHasilKataTidakTerdaftar(arrKataTidakTerdaftar);
      setHasil(arrHasil);
      setHasilKataBaku(arrHasilKataBaku);
    }, time);
  } else {
    alert('Masukkan Kalimat terlebih dahulu');
  }
};
const isFocused = useIsFocused();
if (isFocused) {
  if (route.params) {
    const { loginStatus } = route.params;
    if (loginStatus) {
      navigation.setOptions({
        headerRight: () => (
          <Button
            title="*"
            buttonStyle={{ padding: 0, backgroundColor: 'transparent' }}
            icon={
              <Icon
                name="database"
                size={15}
              // color="white"
              />
            }
            onPress={() => {
              navigation.navigate('Menu');
            }}
          />
        )
      });
    }
  }
}
return (
  <ScrollView style={styles.container}>
    <View style={{marginBottom: 50,}}>
      <View style={styles.subContainer}>
        <TextInput
          multiline={true}
          numberOfLines={1}
          placeholder={'Masukkan kalimat disini'}
          onChangeText={(text) => setKalimat(text)}
          value={kalimat}
        />
      </View>
      <View>
        <Button
```

```
          title="Cek"
          buttonStyle={{ backgroundColor: 'green' }}
          onPress={() => cekKataTidakBaku()}
        />
        {
          (kalimatHasil == '') ? null :
            <Card containerStyle={{ marginTop: 15 }}>
              <Card.Title style={{ textAlign: 'left' }}>Kalimat</Card.Title>
              <Card.Divider />
              <Text style={styles.fonts}>
                {kalimatHasil}
              </Text>
            </Card>
        }

        {
          (hasilKataBaku == '') ? null :
            <Card containerStyle={{ marginTop: 15 }}>
              <Card.Title style={{ textAlign: 'left' }}>Hasil Kata Baku</Card.Title>
              <Card.Divider />
              {hasilKataBaku}
            </Card>
        }
        {
          (hasil == '') ? null :
            <Card containerStyle={{ marginTop: 15 }}>
              <Card.Title style={{ textAlign: 'left' }}>Hasil Kata Tidak Baku</Card.Title>
              <Card.Divider />
              {hasil}
            </Card>
        }
        {
          (hasilKataTidakTerdaftar == '') ? null :
            <Card containerStyle={{ marginTop: 15 }}>
              <Card.Title style={{ textAlign: 'left' }}>Hasil Kata yang tidak ada dalam
Database</Card.Title>
              <Card.Divider />
              {hasilKataTidakTerdaftar}
            </Card>
        }
      </View>
    </View>
  </ScrollView >
 );
}
const Stack = createNativeStackNavigator();
function App() {
 return (
  <NavigationContainer>
    <Stack.Navigator>
      <Stack.Screen name="Home" component={HomeScreen} />
      <Stack.Screen name="Menu" component={MenuScreen} />
      <Stack.Screen name="Login" component={LoginScreen} />
      <Stack.Screen name="Kategori" component={KategoriScreen} />
      <Stack.Screen name="KategoriAdd" component={KategoriAddScreen} />
```

```
        <Stack.Screen name="KategoriDetail" component={KategoriDetailScreen} />
        <Stack.Screen name="KategoriEdit" component={KategoriEditScreen} />
        <Stack.Screen name="KataTidakBaku" component={KataTidakBakuScreen} />
        <Stack.Screen name="KataTidakBakuDetail"
component={KataTidakBakuDetailsScreen} />
        <Stack.Screen name="KataTidakBakuAdd" component={KataTidakBakuAddScreen}
/>
        <Stack.Screen name="KataTidakBakuEdit" component={KataTidakBakuEditScreen}
/>
      </Stack.Navigator>
    </NavigationContainer>
  );
}
const styles = StyleSheet.create({
  container: {
    flex: 1,
    padding: 20
  },
  subContainer: {
    flex: 1,
    marginBottom: 20,
    padding: 5,
    borderBottomWidth: 2,
    borderBottomColor: '#CCCCCC',
  },
  highLight: {
    fontWeight: 'bold',
    color: 'red',
  },
  fonts: {
    marginBottom: 8,
  },
});
export default App;
```

Lampiran-2. *Databases.js*

```
import SQLite from "react-native-sqlite-storage";
SQLite.DEBUG(true);
SQLite.enablePromise(true);

export default class Database {
    initDB() {
        let db;
        return new Promise((resolve) => {
            // console.log("Plugin integrity check ...");
            SQLite.echoTest().then(() => {
                    // console.log("Integrity check passed ...");
                    // console.log("Opening database ...");
                    SQLite.openDatabase({name : "Kata.db", createFromLocation : "~Kata.db"})
                        .then(DB => {
                            db = DB;
                            console.log("Database OPEN");
                            db.executeSql('SELECT 1 FROM KataTidakBaku LIMIT 1').then(() => {
```

```
                        console.log("Database KataTidakBaku is ready ... executing query
...");
                    })
                    db.executeSql('SELECT 1 FROM Kategori LIMIT 1').then(() => {
                        console.log("Database Kategori is ready ... executing query ...");
                    })
                    // .catch((error) => {
                    //     console.log("Received error: ", error);
                    //     console.log("Database not yet ready ... populating data");
                    //     db.transaction((tx) => {
                    //         tx.executeSql('CREATE TABLE IF NOT EXISTS KataTidakBaku
(kbId, ktb, kb)');
                    //     }).then(() => {
                    //         console.log("Table created successfully");
                    //     }).catch(error => {
                    //         console.log(error);
                    //     });
                    // });
                    resolve(db);
                })
                .catch(error => {
                    console.log(error);
                });
            })
            .catch(error => {
                console.log("echoTest failed - plugin not functional");
            });
    });
};
closeDatabase(db) {
    if (db) {
        console.log("Closing DB");
        db.close()
            .then(status => {
                console.log("Database CLOSED");
            })
            .catch(error => {
                this.errorCB(error);
            });
    } else {
        console.log("Database was not OPENED");
    }
};
listKataTidakBaku() {
    return new Promise((resolve) => {
        const ListKata = [];
        this.initDB().then((db) => {
            db.transaction((tx) => {
                tx.executeSql('SELECT * FROM KataTidakBaku', []).then(([tx, results]) => {
                    // console.log("Query completed");
                    var len = results.rows.length;
                    for (let i = 0; i < len; i++) {
                        let row = results.rows.item(i);
                        // console.log(`Kata ID: ${row.kbId}, Kata Tidak Baku: ${row.ktb}, Kata
Baku: ${row.kb}`)
```

```
                        const { kbId, ktb, kb, kb_kategori } = row;
                        ListKata.push({
                            kbId,
                            ktb,
                            kb,
                            kb_kategori
                        });
                    }
                    console.log(ListKata);
                    resolve(ListKata);
                });
            }).then((result) => {
                this.closeDatabase(db);
            }).catch((err) => {
                console.log(err);
            });
        }).catch((err) => {
            console.log(err);
        });
    });
}
findKataTidakBakuById(id) {
    // console.log(id);
    return new Promise((resolve) => {
        this.initDB().then((db) => {
            db.transaction((tx) => {
                tx.executeSql('SELECT * FROM KataTidakBaku WHERE kbId = ?',
[id]).then(([tx, results]) => {
                    console.log(results);
                    if (results.rows.length > 0) {
                        let row = results.rows.item(0);
                        resolve(row);
                    }
                });
            }).then((result) => {
                this.closeDatabase(db);
            }).catch((err) => {
                console.log(err);
            });
        }).catch((err) => {
            console.log(err);
        });
    });
}
findKataTidakBaku(val) {
    // console.log(val);
    return new Promise((resolve) => {
        this.initDB().then((db) => {
            db.transaction((tx) => {
                tx.executeSql('SELECT * FROM KataTidakBaku LEFT JOIN Kategori ON
KataTidakBaku.kb_kategori = Kategori.idKat WHERE ktb like ?', [val]).then(([tx, results])
=> {
                    // console.warn(results);
                    if (results.rows.length > 0) {
                        let row = results.rows.item(0);
```

```
                    resolve(row);
                }else{
                    resolve('');
                }
            });
        }).then((result) => {
            this.closeDatabase(db);
        }).catch((err) => {
            console.warn(err);
        });
    }).catch((err) => {
        console.warn(err);
    });
});
}
findKataTidakBaku2(val1,val2) {
    // console.log(val);
    return new Promise((resolve) => {
        this.initDB().then((db) => {
            db.transaction((tx) => {
                tx.executeSql('SELECT * FROM KataTidakBaku LEFT JOIN Kategori ON
KataTidakBaku.kb_kategori = Kategori.idKat WHERE ktb like ?', [val1+" "+val2]).then(([tx,
results]) => {
                    // console.warn(results);
                    if (results.rows.length > 0) {
                        let row = results.rows.item(0);
                        resolve(row);
                    }else{
                        resolve('');
                    }
                });
            }).then((result) => {
                this.closeDatabase(db);
            }).catch((err) => {
                console.warn(err);
            });
        }).catch((err) => {
            console.warn(err);
        });
    });
}
addKataTidakBaku(k) {
    return new Promise((resolve) => {
        this.initDB().then((db) => {
            db.transaction((tx) => {
                tx.executeSql('INSERT INTO KataTidakBaku VALUES (?, ?, ?, ?)', [null,
k.ktb.toLowerCase(), k.kb.toLowerCase(), k.kb_kategori.toLowerCase()]).then(([tx,
results]) => {
                    resolve(results);
                });
            }).then((result) => {
                this.closeDatabase(db);
            }).catch((err) => {
                console.log(err);
            });
```

```
            }).catch((err) => {
                console.log(err);
            });
        });
    }
    updateKataTidakBaku(id, k) {
        return new Promise((resolve) => {
            this.initDB().then((db) => {
                db.transaction((tx) => {
                    tx.executeSql('UPDATE KataTidakBaku SET ktb = ?, kb = ?, kb_kategori = ?
WHERE kbId = ?', [k.ktb.toLowerCase(), k.kb.toLowerCase(),
k.kb_kategori.toLowerCase(), id]).then(([tx, results]) => {
                        resolve(results);
                    });
                }).then((result) => {
                    this.closeDatabase(db);
                }).catch((err) => {
                    console.log(err);
                });
            }).catch((err) => {
                console.log(err);
            });
        });
    }
    deleteKataTidakBaku(id) {
        return new Promise((resolve) => {
            this.initDB().then((db) => {
                db.transaction((tx) => {
                    tx.executeSql('DELETE FROM KataTidakBaku WHERE kbId = ?',
[id]).then(([tx, results]) => {
                        console.log(results);
                        resolve(results);
                    });
                }).then((result) => {
                    this.closeDatabase(db);
                }).catch((err) => {
                    console.log(err);
                });
            }).catch((err) => {
                console.log(err);
            });
        });
    }
    listKategori() {
        return new Promise((resolve) => {
            const ListKategori = [];
            this.initDB().then((db) => {
                db.transaction((tx) => {
                    tx.executeSql('SELECT * FROM Kategori', []).then(([tx, results]) => {
                        console.log("Query completed");
                        var len = results.rows.length;
                        for (let i = 0; i < len; i++) {
                            let row = results.rows.item(i);
                            console.log(`Kata ID: ${row.idKat}, Kategori: ${row.Kategori}`)
                            const { idKat, Kategori } = row;
```

```
                        ListKategori.push({
                            idKat,
                            Kategori
                        });
                    }
                    console.log(ListKategori);
                    resolve(ListKategori);
                });
            }).then((result) => {
                this.closeDatabase(db);
            }).catch((err) => {
                console.log(err);
            });
        }).catch((err) => {
            console.log(err);
        });
    });
}
findKategoriById(id) {
    console.log(id);
    return new Promise((resolve) => {
        this.initDB().then((db) => {
            db.transaction((tx) => {
                tx.executeSql('SELECT * FROM Kategori WHERE idKat = ?', [id]).then(([tx,
results]) => {
                    console.log(results);
                    if (results.rows.length > 0) {
                        let row = results.rows.item(0);
                        resolve(row);
                    }
                });
            }).then((result) => {
                this.closeDatabase(db);
            }).catch((err) => {
                console.log(err);
            });
        }).catch((err) => {
            console.log(err);
        });
    });
}
addKategori(v) {
    return new Promise((resolve) => {
        this.initDB().then((db) => {
            db.transaction((tx) => {
                tx.executeSql('INSERT INTO Kategori VALUES (?, ?)', [null,
v.Kategori.toLowerCase()]).then(([tx, results]) => {
                    resolve(results);
                });
            }).then((result) => {
                this.closeDatabase(db);
            }).catch((err) => {
                console.log(err);
            });
        }).catch((err) => {
```

```
            console.log(err);
        });
    });
  }
  updateKategori(id, v) {
    return new Promise((resolve) => {
      this.initDB().then((db) => {
        db.transaction((tx) => {
          tx.executeSql('UPDATE Kategori SET Kategori = ? WHERE idKat = ?',
[v.Kategori.toLowerCase(), id]).then(([tx, results]) => {
            resolve(results);
          });
        }).then((result) => {
          this.closeDatabase(db);
        }).catch((err) => {
          console.log(err);
        });
      }).catch((err) => {
        console.log(err);
      });
    });
  }
  deleteKategori(id) {
    return new Promise((resolve) => {
      this.initDB().then((db) => {
        db.transaction((tx) => {
          tx.executeSql('DELETE FROM Kategori WHERE idKat = ?', [id]).then(([tx,
results]) => {
            console.log(results);
            resolve(results);
          });
        }).then((result) => {
          this.closeDatabase(db);
        }).catch((err) => {
          console.log(err);
        });
      }).catch((err) => {
        console.log(err);
      });
    });
  }
}
```

Lampiran-3. *MenuScreen.js*

```
import React, { Component } from 'react';
import { RefreshControl, StyleSheet, FlatList, ActivityIndicator, View, Text } from 'react-
native';
import { Button, ListItem } from 'react-native-elements';
import Database from '../Database';
import Icon from 'react-native-vector-icons/FontAwesome';

const db = new Database();
export default class MenuScreen extends Component {
```

```
    constructor() {
        super();
        this.state = {
            isLoading: false,
            isFetching: false,
        };
    }
    componentDidMount() {
        this.props.navigation.setOptions({
            title: 'Database'
        })
    }
    onRefresh() {
    }
    keyExtractor = (item, index) => index.toString()
    renderItem = ({ item }) => (
        <ListItem bottomDivider onPress={() => {
            this.props.navigation.navigate(item.Link, {idKat: `${item.idKat}`});
        }}>
            <ListItem.Content>
                <ListItem.Title>{item.Judul}</ListItem.Title>
            </ListItem.Content>
            <ListItem.Chevron />
        </ListItem>
    )
    render() {
        if (this.state.isLoading) {
            return (
                <View style={styles.activity}>
                    <ActivityIndicator size="large" color="#0000ff" />
                </View>
            )
        }
        return (
            <View>

                <FlatList
                    keyExtractor={this.keyExtractor}
                    data={[{ "Id": 1, "Judul": "Data Kategori", "Link": "Kategori" }, { "Id": 2, "Judul":
"Data Kata Baku", "Link": "KataTidakBaku" }]}
                    renderItem={this.renderItem}
                    refreshControl={<RefreshControl onRefresh={() => this.onRefresh()}
                        refreshing={this.state.isFetching} />}
                />
                <Button title="Logout" onPress={() => this.props.navigation.navigate('Home', {
                loginStatus: false})} buttonStyle={{ backgroundColor: 'red', width: '50%',
alignSelf: 'center', marginTop: 15 }}/>
            </View>
        );
    }
}
const styles = StyleSheet.create({
    container: {
        flex: 1,
        paddingBottom: 22
```

```
    },
    item: {
        padding: 10,
        fontSize: 18,
        height: 44,
    },
    activity: {
        position: 'absolute',
        left: 0,
        right: 0,
        top: 0,
        bottom: 0,
        alignItems: 'center',
        justifyContent: 'center'
    },
    message: {
        padding: 16,
        fontSize: 18,
        color: 'red'
    }
});
```

Lampiran - 4. *LoginScreen.js*

```
import React, { Component } from 'react';
import { RefreshControl, StyleSheet, FlatList, ActivityIndicator, View, Text } from 'react-
native';
import { Button, ListItem } from 'react-native-elements';
import Database from '../Database';
import Icon from 'react-native-vector-icons/FontAwesome';

const db = new Database();
export default class MenuScreen extends Component {
    constructor() {
        super();
        this.state = {
            isLoading: false,
            isFetching: false,
        };
    }
    componentDidMount() {
        this.props.navigation.setOptions({
            title: 'Database'
        })
    }
    onRefresh() {
    }
    keyExtractor = (item, index) => index.toString()
    renderItem = ({ item }) => (
        <ListItem bottomDivider onPress={() => {
            this.props.navigation.navigate(item.Link, {idKat: `${item.idKat}`});
        }}>
            <ListItem.Content>
                <ListItem.Title>{item.Judul}</ListItem.Title>
```

```
            </ListItem.Content>
            <ListItem.Chevron />
        </ListItem>
    )
    render() {
        if (this.state.isLoading) {
            return (
                <View style={styles.activity}>
                    <ActivityIndicator size="large" color="#0000ff" />
                </View>
            )
        }
        return (
            <View>

                <FlatList
                    keyExtractor={this.keyExtractor}
                    data={[{ "Id": 1, "Judul": "Data Kategori", "Link": "Kategori" }, { "Id": 2, "Judul":
"Data Kata Baku", "Link": "KataTidakBaku" }]}
                    renderItem={this.renderItem}
                    refreshControl={<RefreshControl onRefresh={() => this.onRefresh()}
                        refreshing={this.state.isFetching} />}
                />
                <Button title="Logout" onPress={() => this.props.navigation.navigate('Home', {
                loginStatus: false})} buttonStyle={{ backgroundColor: 'red', width: '50%',
alignSelf: 'center', marginTop: 15 }}/>
            </View>
        );
    }
}
const styles = StyleSheet.create({
    container: {
        flex: 1,
        paddingBottom: 22
    },
    item: {
        padding: 10,
        fontSize: 18,
        height: 44,
    },
    activity: {
        position: 'absolute',
        left: 0,
        right: 0,
        top: 0,
        bottom: 0,
        alignItems: 'center',
        justifyContent: 'center'
    },
    message: {
        padding: 16,
        fontSize: 18,
        color: 'red'
    }
});
```

Lampiran – 5. *KataTidakBakuScreen.js*

```
import React, { Component } from 'react';
import { RefreshControl, StyleSheet, FlatList, ActivityIndicator, View, Text } from 'react-
native';
import { ListItem, Button } from 'react-native-elements';
import Database from '../Database';
import Icon from 'react-native-vector-icons/FontAwesome';

const db = new Database();
export default class KataTidakBakuScreen extends Component {
  constructor() {
    super();
    this.state = {
      isLoading: true,
      isFetching: false,
      KataTidakBaku: [],
      arrKategori: [],
      notFound: 'Database kosong.\nTekan tombol (+) untuk menambah data.'
    };
  }
  componentDidMount() {
    this.forceUpdate();
    this.getKataTidakBaku();
    db.listKategori().then((res) => {
      res.forEach(element => {
        this.state.arrKategori.push([element.idKat,element.Kategori])
      })
    })
    this.props.navigation.setOptions({
      title: 'List Kata Tidak Baku',
      headerRight: () => (
        <Button
          buttonStyle={{ padding: 0, backgroundColor: 'transparent' }}
          icon={{ name: 'add', style: { marginRight: 0, fontSize: 28 } }}
          onPress={() => {
            this.props.navigation.navigate('KataTidakBakuAdd', {arrKat:
this.state.arrKategori});
          }}
        />
      )
    });
  }
  onRefresh() {
    this.setState({ isFetching: true }, () => { this.getKataTidakBaku(); });
    console.warn(this.state.KataTidakBaku);
  }
  getKataTidakBaku() {
    let KataTidakBaku = [];
    db.listKataTidakBaku().then((data) => {
      KataTidakBaku = data;
      this.setState({
        KataTidakBaku,
```

```
        isLoading: false,
        isFetching: false
      });
    }).catch((err) => {
      console.log(err);
      this.setState = {
        isLoading: false,
        isFetching: false
      }
    })
  }
  keyExtractor = (item, index) => index.toString()
  renderItem = ({ item }) => (
    <ListItem bottomDivider onPress={() => {
      this.props.navigation.navigate('KataTidakBakuDetail', {
        kbId: `${item.kbId}`, arrKat: this.state.arrKategori,
      });
    }}>
      {/* <Avatar title={item.kbId}/> */}
      <ListItem.Content>
        <ListItem.Title>{item.ktb}</ListItem.Title>
        <ListItem.Subtitle>{item.kb}</ListItem.Subtitle>
      </ListItem.Content>
      <ListItem.Chevron />
    </ListItem>
  )
  render() {
    if (this.state.isLoading) {
      return (
        <View style={styles.activity}>
          <ActivityIndicator size="large" color="#0000ff" />
        </View>
      )
    }
    if (this.state.KataTidakBaku.length === 0) {
      return (
        <View>
          <Text style={styles.message}>{this.state.notFound}</Text>
        </View>
      )
    }
    return (
      <FlatList
        keyExtractor={this.keyExtractor}
        data={this.state.KataTidakBaku}
        renderItem={this.renderItem}
        refreshControl={<RefreshControl onRefresh={() => this.onRefresh()}
        refreshing={this.state.isFetching} />}
      />
    );
  }
}

const styles = StyleSheet.create({
  container: {
```

```
    flex: 1,
    paddingBottom: 22
  },
  item: {
    padding: 10,
    fontSize: 18,
    height: 44,
  },
  activity: {
    position: 'absolute',
    left: 0,
    right: 0,
    top: 0,
    bottom: 0,
    alignItems: 'center',
    justifyContent: 'center'
  },
  message: {
    padding: 16,
    fontSize: 18,
    color: 'red'
  }
});
```

Lampiran – 6 . *KataTidakBakuAddScreen.js*

```
import React, { Component } from 'react';
import { StyleSheet, ScrollView, ActivityIndicator, View, TextInput } from 'react-native';
import { Button } from 'react-native-elements';
import Database from '../Database';
import { Picker } from '@react-native-picker/picker';

const db = new Database();

export default class KataTidakBakuAddScreen extends Component {
  constructor() {
    super();
    this.state = {
      // kbId: '',
      kb: '',
      ktb: '',
      kb_kategori: '',
      isLoading: false,
    };
  }
  componentDidMount() {
    this.props.navigation.setOptions({
      title: 'Tambah Kata Tidak Baku',
    });
    // this.kategoriList()
  }
  kategoriList = () => {
    // const params = this.props.route.params;
    return (this.props.route.params.arrKat.map((x, i) => {
```

```
    console.warn(x[1]);
    return (<Picker.Item label={x[1]} key={i} value={x[0].toString()} />)
  }));
}
updateTextInput = (text, field) => {
  const state = this.state
  state[field] = text;
  this.setState(state);
}
saveProduct() {
  this.setState({
    isLoading: true,
  });
  let data = {
    // kbId: this.state.kbId,
    ktb: this.state.ktb,
    kb: this.state.kb,
    kb_kategori: this.state.kb_kategori,
  }
  db.addKataTidakBaku(data).then((result) => {
    console.log(result);
    this.setState({
      isLoading: false,
    });
    this.props.navigation.goBack();
  }).catch((err) => {
    console.log(err);
    this.setState({
      isLoading: false,
    });
  })
}
render() {
  if (this.state.isLoading) {
    return (
      <View style={styles.activity}>
        <ActivityIndicator size="large" color="#0000ff" />
      </View>
    )
  }
  // this.tes()
  return (
    <ScrollView style={styles.container}>
      <View style={styles.subContainer}>
        <TextInput
          placeholder={'Kata Tidak Baku'}
          value={this.state.ktb}
          onChangeText={(text) => this.updateTextInput(text, 'ktb')}
        />
      </View>
      <View style={styles.subContainer}>
        <TextInput
          // multiline={true}
          // numberOfLines={4}
          placeholder={'Kata Baku'}
```

```
            value={this.state.kb}
            onChangeText={(text) => this.updateTextInput(text, 'kb')}
          />
        </View>
        <View>
          {/* <TextInput
            // multiline={true}
            // numberOfLines={4}
            placeholder={'Kategori'}
            value={this.state.kb_kategori}
            onChangeText={(text) => this.updateTextInput(text, 'kb_kategori')}
          /> */}
          <Picker
            selectedValue={this.state.kb_kategori}
            style={{ height: 50, width: 150 }}
            onValueChange={(itemValue, itemIndex) => this.updateTextInput(itemValue,
'kb_kategori')}
          // onValueChange={(itemValue, itemIndex) => setSelectedValue(itemValue)}
          >
            { this.kategoriList() }
          </Picker>
        </View>
        <View style={styles.button}>
          <Button
            large
            leftIcon={{ name: 'save' }}
            title='Save'
            onPress={() => this.saveProduct()} />
        </View>
      </ScrollView>
    );
  }
}
const styles = StyleSheet.create({
  container: {
    flex: 1,
    padding: 20
  },
  subContainer: {
    flex: 1,
    marginBottom: 20,
    padding: 5,
    borderBottomWidth: 2,
    borderBottomColor: '#CCCCCC',
  },
  activity: {
    position: 'absolute',
    left: 0,
    right: 0,
    top: 0,
    bottom: 0,
    alignItems: 'center',
    justifyContent: 'center'
  }
})
```

Lampiran – 7. *kataTidakBakuDetailScreen.js*

```
import React, { Component } from 'react';
import { ScrollView, StyleSheet, Image, ActivityIndicator, View, Text } from 'react-native';
import { Card, Button } from 'react-native-elements';
import Database from '../Database';

const db = new Database();

export default class KataTidakBakuDetailsScreen extends Component {
  constructor() {
    super();
    this.state = {
      isLoading: true,
      data: {},
      id: '',
    };
  }
  componentDidMount() {
    this.props.navigation.setOptions({
      title: 'Detail Kata Tidak Baku',
    });
    const params = this.props.route.params;
    db.findKataTidakBakuById(params.kbId).then((data) => {
      data = data;
      this.setState({
        data: data,
        isLoading: false,
        id: data.kbId
      });
    }).catch((err) => {
      console.log(err);
      this.setState = {
        isLoading: false
      }
    })
  }
  deleteKataTidakBaku(id) {
    this.setState({
      isLoading: true
    });
    db.deleteKataTidakBaku(id).then((result) => {
      console.log(result);
      this.props.navigation.goBack();
    }).catch((err) => {
      console.log(err);
      this.setState = {
        isLoading: false
      }
    })
  }
  render() {
    if (this.state.isLoading) {
```

```
      return (
        <View style={styles.activity}>
          <ActivityIndicator size="large" color="#0000ff" />
        </View>
      )
    }
    return (
      <ScrollView>
        <Card style={styles.container}>
          <View style={styles.subContainer}>
            <View>
              <Text style={{ fontSize: 16 }}>ID: {this.state.data.kbId}</Text>
            </View>
            <View>
              <Text style={{ fontSize: 16 }}>Kata Tidak Baku: {this.state.data.ktb}</Text>
            </View>
            <View>
              <Text style={{ fontSize: 16 }}>Kata Baku: {this.state.data.kb}</Text>
            </View>
            <View>
              {/* <Text style={{ fontSize: 16 }}>Kategori: {this.state.data.kb_kategori}</Text> */}
              <Text style={{ fontSize: 16 }}>Kategori: {
                this.props.route.params.arrKat.filter((v) => {
                  return v[0] == this.state.data.kb_kategori
                })[0][1]
              }</Text>
            </View>
          </View>
          <View style={styles.detailButton}>
            <Button
              large
              backgroundColor={'#CCCCCC'}
              leftIcon={{ name: 'edit' }}
              title='Edit'
              onPress={() => {
                this.props.navigation.navigate('KataTidakBakuEdit', {
                  kbId: `${this.state.id}`, arrKat: this.props.route.params.arrKat,
kategori_selected: this.state.data.kb_kategori,
                });
              }} />
          </View>
          <View style={styles.detailButton}>
            <Button
              large
              backgroundColor={'#999999'}
              color={'#FFFFFF'}
              leftIcon={{ name: 'delete' }}
              title='Delete'
              onPress={() => this.deleteKataTidakBaku(this.state.id)} />
          </View>
        </Card>
      </ScrollView>
    );
  }
}
```

```
const styles = StyleSheet.create({
  container: {
    flex: 1,
    padding: 20
  },
  subContainer: {
    flex: 1,
    paddingBottom: 20,
    borderBottomWidth: 2,
    borderBottomColor: '#CCCCCC',
  },
  activity: {
    position: 'absolute',
    left: 0,
    right: 0,
    top: 0,
    bottom: 0,
    alignItems: 'center',
    justifyContent: 'center'
  },
  detailButton: {
    marginTop: 10
  }
})
```

Lampiran – 8. *KataTidakBakuEditScreen.js*

```
import React, { Component } from 'react';
import { StyleSheet, ScrollView, ActivityIndicator, View, TextInput } from 'react-native';
import { Button } from 'react-native-elements';
import { Picker } from '@react-native-picker/picker';
import Database from '../Database';

const db = new Database();
export default class KataTidakBakuEditScreen extends Component {
  constructor() {
    super();
    this.state = {
      kbId: '',
      ktb: '',
      kb: '',
      kb_kategori: '',
      isLoading: true,
    };
  }
  componentDidMount() {
    this.state.kb_kategori = this.props.route.params.kategori_selected
    console.warn(this.props.route.params.kategori_selected);
    this.props.navigation.setOptions({
      title: 'Edit Kata Tidak Baku',
    });
    const navigation = this.props.route.params;
    db.findKataTidakBakuById(navigation.kbId).then((data) => {
      console.log(data);
```

```
      const setData = data;
      this.setState({
        kbId: setData.kbId,
        ktb: setData.ktb,
        kb: setData.kb,
        kb_kategori: setData.kb_kategori,
        isLoading: false,
      });
    }).catch((err) => {
      console.log(err);
      this.setState = {
        isLoading: false
      }
    })
  }
  kategoriList = () => {
    // const params = this.props.route.params;
    return (this.props.route.params.arrKat.map((x, i) => {
      // console.warn(x[1]);
      return (<Picker.Item label={x[1]} key={i} value={x[0].toString()} />)
    }));
  }
  updateTextInput = (text, field) => {
    const state = this.state
    state[field] = text;
    this.setState(state);
  }
  updateProduct() {
    this.setState({
      isLoading: true,
    });
    let data = {
      kbId: this.state.kbId,
      ktb: this.state.ktb,
      kb: this.state.kb,
      kb_kategori: this.state.kb_kategori
    }
    db.updateKataTidakBaku(data.kbId, data).then((result) => {
      console.warn(result);
      this.setState({
        isLoading: false,
      });
      this.props.navigation.goBack();
    }).catch((err) => {
      console.log(err);
      this.setState({
        isLoading: false,
      });
    })
  }
  render() {
    if (this.state.isLoading) {
      return (
        <View style={styles.activity}>
          <ActivityIndicator size="large" color="#0000ff" />
```

```
        </View>
      )
    }
    return (
      <ScrollView style={styles.container}>
        <View style={styles.subContainer}>
          <TextInput
            editable={false}
            placeholder={'ID'}
            value={this.state.kbId.toString()}
            onChangeText={(text) => this.updateTextInput(text, 'kbId')}
          />
        </View>
        <View style={styles.subContainer}>
          <TextInput
            placeholder={'Kata Tidak Baku'}
            value={this.state.ktb}
            onChangeText={(text) => this.updateTextInput(text, 'ktb')}
          />
        </View>
        <View style={styles.subContainer}>
          <TextInput
            placeholder={'Kata Baku'}
            value={this.state.kb}
            onChangeText={(text) => this.updateTextInput(text, 'kb')}
          />
        </View>
        <View>
          {/* <TextInput
            // multiline={true}
            // numberOfLines={4}
            placeholder={'Kategori'}
            value={this.state.kb_kategori}
            onChangeText={(text) => this.updateTextInput(text, 'kb_kategori')}
          /> */}
          <Picker
            selectedValue={this.state.kb_kategori}
            style={{ height: 50, width: 150 }}
            onValueChange={(itemValue, itemIndex) => this.updateTextInput(itemValue,
'kb_kategori')}
          // onValueChange={(itemValue, itemIndex) => setSelectedValue(itemValue)}
          >
            { this.kategoriList() }
          </Picker>
        </View>
        <View style={styles.button}>
          <Button
            large
            leftIcon={{ name: 'save' }}
            title='Save'
            onPress={() => this.updateProduct()} />
        </View>
      </ScrollView>
    );
  }
```

```
}
const styles = StyleSheet.create({
  container: {
    flex: 1,
    padding: 20
  },
  subContainer: {
    flex: 1,
    marginBottom: 20,
    padding: 5,
    borderBottomWidth: 2,
    borderBottomColor: '#CCCCCC',
  },
  activity: {
    position: 'absolute',
    left: 0,
    right: 0,
    top: 0,
    bottom: 0,
    alignItems: 'center',
    justifyContent: 'center'
  }
})
```

Lampiran – 9. *KategoriScreen.js*

```
import React, { Component } from 'react';
import { RefreshControl, StyleSheet, FlatList, ActivityIndicator, View, Text } from 'react-
native';
import { ListItem, Button } from 'react-native-elements';
import Database from '../Database';
import Icon from 'react-native-vector-icons/FontAwesome';

const db = new Database();
export default class KategoriScreen extends Component {
  constructor() {
    super();
    this.state = {
      isLoading: true,
      isFetching: false,
      Kategori: [],
      notFound: 'Database kosong.\nTekan tombol (+) untuk menambah data.'
    };
  }
  componentDidMount() {
    this.forceUpdate();
    this.getKategori();

    this.props.navigation.setOptions({
      title: 'List Kategori',
      headerRight: () => (
        <Button
          buttonStyle={{ padding: 0, backgroundColor: 'transparent' }}
          icon={{ name: 'add', style: { marginRight: 0, fontSize: 28 } }}
```

```
        onPress={() => {
          this.props.navigation.navigate('KategoriAdd');
        }}
      />
    )
  });
}
onRefresh() {
  this.setState({ isFetching: true }, () => { this.getKategori(); });
  console.warn(this.state.Kategori);
}
getKategori() {
  let Kategori = [];
  db.listKategori().then((data) => {
    Kategori = data;
    this.setState({
      Kategori,
      isLoading: false,
      isFetching: false
    });
  }).catch((err) => {
    console.log(err);
    this.setState = {
      isLoading: false,
      isFetching: false
    }
  })
}
keyExtractor = (item, index) => index.toString()
renderItem = ({ item }) => (
  <ListItem bottomDivider onPress={() => {
    this.props.navigation.navigate('KategoriDetail', {
      idKat: `${item.idKat}`,
    });
  }}>
    {/* <Avatar title={item.kbId}/> */}
    <ListItem.Content>
      <ListItem.Title>{item.Kategori}</ListItem.Title>
    </ListItem.Content>
    <ListItem.Chevron />
  </ListItem>
)
render() {
  if (this.state.isLoading) {
    return (
      <View style={styles.activity}>
        <ActivityIndicator size="large" color="#0000ff" />
      </View>
    )
  }
  if (this.state.Kategori.length === 0) {
    return (
      <View>
        <Text style={styles.message}>{this.state.notFound}</Text>
      </View>
```

```
      )
    }
    return (
      <FlatList
        keyExtractor={this.keyExtractor}
        data={this.state.Kategori}
        renderItem={this.renderItem}
        refreshControl={<RefreshControl onRefresh={() => this.onRefresh()}
        refreshing={this.state.isFetching} />}
      />
    );
  }
}
const styles = StyleSheet.create({
  container: {
    flex: 1,
    paddingBottom: 22
  },
  item: {
    padding: 10,
    fontSize: 18,
    height: 44,
  },
  activity: {
    position: 'absolute',
    left: 0,
    right: 0,
    top: 0,
    bottom: 0,
    alignItems: 'center',
    justifyContent: 'center'
  },
  message: {
    padding: 16,
    fontSize: 18,
    color: 'red'
  }
});
```

Lampiran – 10. *KategoriAddScreen.js*

```
import React, { Component } from 'react';
import { StyleSheet, ScrollView, ActivityIndicator, View, TextInput } from 'react-native';
import { Button } from 'react-native-elements';
import Database from '../Database';
const db = new Database();
export default class KategoriAddScreen extends Component {
  constructor() {
    super();
    this.state = {
      Kategori: '',
      isLoading: false,
    };
  }
```

```
componentDidMount() {
  this.props.navigation.setOptions({
    title: 'Tambah Kategori',
  });
}
updateTextInput = (text, field) => {
  const state = this.state
  state[field] = text;
  this.setState(state);
}
saveProduct() {
  this.setState({
    isLoading: true,
  });
  let data = {
    // kbId: this.state.kbId,
    Kategori: this.state.Kategori,
  }
  db.addKategori(data).then((result) => {
    console.log(result);
    this.setState({
      isLoading: false,
    });
    this.props.navigation.goBack();
  }).catch((err) => {
    console.log(err);
    this.setState({
      isLoading: false,
    });
  })
}
render() {
  if (this.state.isLoading) {
    return (
      <View style={styles.activity}>
        <ActivityIndicator size="large" color="#0000ff" />
      </View>
    )
  }
  return (
    <ScrollView style={styles.container}>
      <View style={styles.subContainer}>
        <TextInput
          placeholder={'Kategori'}
          value={this.state.Kategori}
          onChangeText={(text) => this.updateTextInput(text, 'Kategori')}
        />
      </View>
      <View style={styles.button}>
        <Button
          large
          leftIcon={{ name: 'save' }}
          title='Save'
          onPress={() => this.saveProduct()} />
      </View>
```

```
      </ScrollView>
    );
  } }
const styles = StyleSheet.create({
  container: {
    flex: 1,
    padding: 20
  },
  subContainer: {
    flex: 1,
    marginBottom: 20,
    padding: 5,
    borderBottomWidth: 2,
    borderBottomColor: '#CCCCCC',
  },
  activity: {
    position: 'absolute',
    left: 0,
    right: 0,
    top: 0,
    bottom: 0,
    alignItems: 'center',
    justifyContent: 'center'
  }
})
```

Lampiran – 11. *KategoriDetailScreen.js*

```
import React, { Component } from 'react';
import { ScrollView, StyleSheet, Image, ActivityIndicator, View, Text } from 'react-native';
import { Card, Button } from 'react-native-elements';
import Database from '../Database';
const db = new Database();
export default class KategoriDetailsScreen extends Component {
  constructor() {
    super();
    this.state = {
      isLoading: true,
      data: {},
      id: '',
    };
  }
  componentDidMount() {
    this.props.navigation.setOptions({
      title: 'Detail Kategori',
    });
    const params = this.props.route.params;
    db.findKategoriById(params.idKat).then((data) => {
      data = data;
      this.setState({
        data: data,
        isLoading: false,
        id: data.idKat
      });
```

```
    }).catch((err) => {
      console.log(err);
      this.setState = {
        isLoading: false
      }
    }) }
  deleteKategori(id) {
    this.setState({
      isLoading: true
    });
    db.deleteKategori(id).then((result) => {
      console.log(result);
      this.props.navigation.goBack();
    }).catch((err) => {
      console.log(err);
      this.setState = {
        isLoading: false
      }
    })  }
  render() {
    if (this.state.isLoading) {
      return (
        <View style={styles.activity}>
          <ActivityIndicator size="large" color="#0000ff" />
        </View>
      ) }
    return (
      <ScrollView>
        <Card style={styles.container}>
          <View style={styles.subContainer}>
            <View>
              <Text style={{ fontSize: 16 }}>ID: {this.state.data.idKat}</Text>
            </View>
            <View>
              <Text style={{ fontSize: 16 }}>Kategori: {this.state.data.Kategori}</Text>
            </View>
          </View>
          <View style={styles.detailButton}>
            <Button
              large
              backgroundColor={'#CCCCCC'}
              leftIcon={{ name: 'edit' }}
              title='Edit'
              onPress={() => {
                this.props.navigation.navigate('KategoriEdit', {
                  idKat: `${this.state.id}`,
                });
              }} />
          </View>
          <View style={styles.detailButton}>
            <Button
              large
              backgroundColor={'#999999'}
              color={'#FFFFFF'}
              leftIcon={{ name: 'delete' }}
```

```
            title='Delete'
            onPress={() => this.deleteKategori(this.state.id)} />
          </View>
        </Card>
      </ScrollView>
    ); }
}
const styles = StyleSheet.create({
  container: {
    flex: 1,
    padding: 20
  },
  subContainer: {
    flex: 1,
    paddingBottom: 20,
    borderBottomWidth: 2,
    borderBottomColor: '#CCCCCC',
  },
  activity: {
    position: 'absolute',
    left: 0,
    right: 0,
    top: 0,
    bottom: 0,
    alignItems: 'center',
    justifyContent: 'center'
  },
  detailButton: {
    marginTop: 10
  }
})
```

Lampiran – 12. *KategoriEditScreen.js*

```
import React, { Component } from 'react';
import { StyleSheet, ScrollView, ActivityIndicator, View, TextInput } from 'react-native';
import { Button } from 'react-native-elements';
import { Picker } from '@react-native-picker/picker';
import Database from '../Database';
const db = new Database();
export default class KategoriEditScreen extends Component {
  constructor() {
    super();
    this.state = {
      idKat: '',
      Kategori: '',
      isLoading: true,
    }; }
  componentDidMount() {
    this.props.navigation.setOptions({
      title: 'Edit Kategori',
    });
    const navigation = this.props.route.params;
    console.warn(navigation.idKat);
```

```
    db.findKategoriById(navigation.idKat).then((data) => {
      // console.log(data);
      const setData = data;
      this.setState({
        idKat: setData.idKat,
        Kategori: setData.Kategori,
        isLoading: false,
      });
    }).catch((err) => {
      console.log(err);
      this.setState = {
        isLoading: false
      } }) }
  updateTextInput = (text, field) => {
    const state = this.state
    state[field] = text;
    this.setState(state);
  }
  updateProduct() {
    this.setState({
      isLoading: true,
    });
    let data = {
      idKat: this.state.idKat,
      Kategori: this.state.Kategori,
    }
    db.updateKategori(data.idKat, data).then((result) => {
      console.warn(result);
      this.setState({
        isLoading: false,
      });
      this.props.navigation.goBack();
    }).catch((err) => {
      console.log(err);
      this.setState({
        isLoading: false,
      }); }) }
  render() {
    if (this.state.isLoading) {
      return (
        <View style={styles.activity}>
          <ActivityIndicator size="large" color="#0000ff" />
        </View>
      ) }
    return (
      <ScrollView style={styles.container}>
        <View style={styles.subContainer}>
          <TextInput
            editable={false}
            placeholder={'ID'}
            value={this.state.idKat.toString()}
            onChangeText={(text) => this.updateTextInput(text, 'idKat')}
          />
        </View>
        <View style={styles.subContainer}>
```

```
          <TextInput
            placeholder={'Kategori'}
            value={this.state.Kategori}
            onChangeText={(text) => this.updateTextInput(text, 'Kategori')}
          />
        </View>
        <View style={styles.button}>
          <Button
            large
            leftIcon={{ name: 'save' }}
            title='Save'
            onPress={() => this.updateProduct()} />
        </View>
      </ScrollView>
    ); } }
const styles = StyleSheet.create({
  container: {
    flex: 1,
    padding: 20
  },
  subContainer: {
    flex: 1,
    marginBottom: 20,
    padding: 5,
    borderBottomWidth: 2,
    borderBottomColor: '#CCCCCC',
  },
  activity: {
    position: 'absolute',
    left: 0,
    right: 0,
    top: 0,
    bottom: 0,
    alignItems: 'center',
    justifyContent: 'center'
  } } )
```