```
1
2       import {LogCache} from './modules/logcache.js'
3       import {OfflineLog} from './modules/log.js';
4       import {formatDateForDisplay} from './modules/helper.js';
5
6       // Global variables
7       let logCache = new LogCache();
8
9       let log;
10
11      window.onload = setup;
12
13      function setup() {
14
15          // Attempt to sync with the database
16          attemptDbSync();
17
18          // If there's an inprogress log that hasn't been saved prompt the user about how to handle it
19          if (localStorage.getItem('currentLog')) handleInProgressLog();
20
21          // Bind all the event listeners
22          addEventListeners();
23
24      }
25
26
27      function addEventListeners() {
28
29          // welcomeScreen View
30          document.getElementById('createNew').addEventListener('click', displayRecorder);
31
32          // inProgressLog view
33          document.getElementById('resumeInProgress').addEventListener('click', displayRecorder);
34          document.getElementById('saveInProgress').addEventListener('click', handleSave);
35          document.getElementById('deleteInProgress').addEventListener('click', handleDiscard);
36
37          // logRecorder view
38          document.getElementById('saveLog').addEventListener('click', handleSave);
39          document.getElementById('undo').addEventListener('click', handleUndo);
40          document.getElementById('redo').addEventListener('click', handleRedo);
41          let addButtons = document.getElementsByClassName('addBird');
42          for (let button of addButtons) {
43              button.addEventListener('click', addBird)
44          }
45
46          // saveFields view
47          document.getElementById('save').addEventListener('click', (e) => {
48              e.preventDefault();
49              log.weather = e.target.form.weather.value;
50              log.notes = e.target.form.notes.value;
51              saveLog();
52          });
53          document.getElementById('discard').addEventListener('click', handleDiscard);
54
55      }
56
57
58      function handleDiscard() {
59          let remove = window.confirm("Are you sure you want to discard this log? This cannot be undone.")
60          if (remove) {
61              localStorage.removeItem('currentLog')
```

```
62              log = undefined;
63              showView('welcomeScreen');
64          };
65      }
66
67
68
69      function handleInProgressLog() {
70
71          showView('inProgressLog');
72
73          // Create a new log from the localstorage data and save it to the global log variable;
74          let logData = JSON.parse(localStorage.getItem('currentLog'))
75          log = OfflineLog.fromData(logData);
76
77          document.getElementById('inProgressLogDate').innerHTML = formatDateForDisplay(log.startTime);
78          document.getElementById('inProgressLogCount').innerHTML = log.currentCount + (log.currentCount == 1 ? " bird"
79  : " birds");
80
81      }
82
83
84      function displayRecorder() {
85
86          log = log ? log : new OfflineLog('realTime', new Date());
87
88          showView('logRecorder');
89          updateUndoRedoState();
90
91          document.getElementById('currentTotal').innerHTML = log.currentCount;
92
93      }
94
95      function handleUndo() {
96          log.undo();
97
98          // Update undo / redo buttons
99          updateUndoRedoState();
100
101         // Display the new total
102         document.getElementById('currentTotal').innerHTML = log.currentCount;
103     }
104
105     function handleRedo() {
106         log.redo();
107
108         // Update undo / redo buttons
109         updateUndoRedoState();
110
111         // Display the new total
112         document.getElementById('currentTotal').innerHTML = log.currentCount;
113     }
114
115     function handleSave(e) {
116         if (log.data.length > 0) {
117
118             // Zero out the weather and notes fileds
119             let weather = document.getElementById('weather').value = "";
120             let notes = document.getElementById('notes').value = "";
121
122             // Show the save fields view
123             showView('saveFields');
```

```
124         }
125         else {
126             showView("welcomeScreen");
127         }
128     }
129 
130     function getStatusMessage() {
131         let message = "";
132         let logCount = logCache.logs.length;
133 
134         message = navigator.onLine ? "Online: " : "Offline: ";
135         if (logCount == 0) message += "All logs synced";
136         else message += `${logCount} ${logCount == 1 ? "log" : "logs"} waiting to be synced`;
137 
138         return message;
139     }
140 
141     function addBird(e) {
142         // Add the birds to the log
143         let num = Number(e.target.value);
144         log.addData(num);
145 
146         // Update the undo / redo buttons
147         updateUndoRedoState()
148 
149         // Display the new total
150         document.getElementById('currentTotal').innerHTML = log.currentCount;
151     }
152 
153     function saveLog() {
154         // Add the log to the log cache and remove the temporary currentLog storage
155 
156         if (log.data.length > 0) {
157             // Save the log to the log cache
158             logCache.addLog(log);
159             localStorage.removeItem('currentLog');
160             log = undefined;
161         }
162 
163         // Display the home view
164         showView("welcomeScreen")
165 
166         // Attempt to sync with the database
167         attemptDbSync();
168 
169     }
170 
171     function attemptDbSync() {
172         let statusOut = document.getElementById('status')
173 
174         if (navigator.onLine && logCache.hasLogs()) {
175             statusOut.innerHTML = "Syncing Logs...";
176 
177             console.log(logCache);
178 
179             logCache.addToDb()
180                 .then(() => {
181                     statusOut.innerHTML = "Online: All logs synced";
182                 })
183                 .catch((e) => {
184                     console.log(e);
185                     statusOut.innerHTML = "Error syncing logs. All data is still saved locally and will be resynced
```

```
the next time you launch the app.";
                })
        }

        else statusOut.innerHTML = getStatusMessage();
    }


    function showView(id) {
        let views = document.getElementsByClassName('view');
        for (let view of views) {
            if (view.id == id) view.classList.remove('hide');
            else view.classList.add('hide');
        }
    }

    function updateUndoRedoState() {
        let undo = document.getElementById('undo');
        let redo = document.getElementById('redo');

        undo.disabled = log.data.length > 0 ? false : true;
       redo.disabled = log.redoCache.length > 0 ? false : true;
}
```