DevOP by OpFlow

Christopher Lebovitz
Courtney Hill
Kacheef White
James Setliff
Charles Young

February 8, 2021

1.0 Scope

1.1 Identification

DevOP v0.1a, Productivity and Communication tool for development teams.

1.2 System Overview

DevOP is based on the Electron framework and as such will run on any modern operating system. The application brings the functionality of many popular development tools under a single umbrella and supports personal and team-based record keeping, visual task board to easily convey that information, and real-time chat with persistence. DevOP is targeted at teams that want modern planning and team-work functionality without the monthly subscription and cloud-based storage of information that most of these tools require.

The DevOP client will be deployed to each individual team member's computer but may also be hosted on a webserver for remote connections, thanks to the flexibility provided by Electron. The client can be pointed at any MySQL database for storage of information; however, it is designed with the idea of a team-owned database managed locally. Team members will be able to create and update records, manipulate task boards, and communicate with one another or groups -- all within a single application.

1.3 Document Overview

The document shall serve as the point of reference for all notable reference materials; including frameworks and packages the program relies on, and any other useful technical guides used during the development process. Additionally, all technical and design requirements, testing plans, and test results will be documented and maintained herein.

2.0 Referenced Documents

Frameworks:

Electron - https://www.electronjs.org/

React - https://reactjs.org/

Node.js - https://nodejs.org/en/

MySQL - https://www.mysql.com/

Packages:

react - https://www.npmjs.com/package/react

react-dom - https://www.npmjs.com/package/react-dom

Express - https://www.npmjs.com/package/express

socket.io - https://www.npmjs.com/package/socket.io

mysql - https://www.npmjs.com/package/mysql

References:

w3schools.com - https://www.w3schools.com/

Additional Tools:

Visual Studio Code - https://code.visualstudio.com/

MySQL Workbench - https://www.mysql.com/products/workbench/

3.0 Requirements

Rqmt ID	Rqmt Statement	Design Section #	Test Section #
1.0	Program shall be cross-platform and run on an “personal computer” type device with a modern operating system.		
1.1	Electron framework		
1.1.1	User views are rendered using React’s virtual DOM.		
1.1.2	Remote server provides real-time communication between clients and will resolve and handle database queries.		
1.1.3	Database will provide appropriate tables for storage of necessary information.		
2.0	Program shall provide, at a minimum, task tracking, visual task board, chat communication		
2.1	User shall be able to create a new record containing name of task, task number, task description, task assignment, date started, date ended		
2.2	User shall have access to a drag-and-drop interactable task card that can move between columns, providing tracking for task progress		
2.2.1	Task card shall consist of task name, task descript, and task assignee		
2.2.2	Task cards shall be re-orderable in the individual columns		
2.3	Users shall be able to communicate with one another via persistent, real-time chat		
2.3.1	Chat messages shall consist of user identifier, timestamp, and message and shall be stored indefinitely		
3.0	Program shall have a settings page with Username, password, database address, database port number		
4.0	Search Functionality		
4.1	Search bar shall return results including users, task title, task description, and chat message		
4.2	Search bar shall return a results page		

4.0 Design

How are you going to meet the requirements? This is technical requirements and what the user thinks it should do or behave. Examples of technical requirements are what OS will you use, tools need for development, device drivers, libraries, etc… A user requirement example would be a configuration file that is ASCII rather than binary or always put the OK button in the lower right-hand corner.

- 1.0 - Program shall be cross-platform and run on any "personal computer" type device with a modern operating system.
  - 1.1 Use Electron framework with React and NodeJS, supported by MySQL for database storage needs
    - 1.1.1 All client-side configuration will be provided through React to provide a seamless experience. Client to be configured as described in Design 2.0-4.0.
    - 1.1.2 NodeJS remote server shall be configured to support real-time communication via socket.io with clients, and handle database queries.
    - 1.1.3 MySQL database to be configured using industry standard best-practices for schema and table design.
  - 1.2 Compile using Electron-Forge to create executable for Windows, Linux, or Mac
- 2.0 - Program shall provide, at a minimum, task tracking, visual task board, chat communication
  - 2.1 Task Tracking
    - 2.1.1 Task record shall contain name of task, task number, task description, task assignee (who is working on it), date started, date ended.
    - 2.1.2 Task records shall be assigned a unique ID.
    - 2.1.3 Task record should be displayed as the main content when opened.
    - 2.1.4 Task record title and description shall be indexed.
  - 2.2 Visual Task Board
    - 2.2.1 Visual Task Board shall be comprised of columns that contain task cards.
    - 2.2.2 Visual Task Board columns shall be renamable.
    - 2.2.3 Task cards shall be moveable between columns.
    - 2.2.4 Columns and cards shall be removeable.
    - 2.2.5 Visual Task Board shall display as the main content when opened.
  - 2.3 Chat Communications
    - 2.3.1 Chat shall support communication with one another via persistent, real-time chat.
    - 2.3.2 Chat messages shall be stored in a database table and returnable via query.

2.3.3 Chat message shall be indexed.

3.0 Setting Page

3.1 Settings page shall include fields for Username, password, database address, database port number.

3.1.1 Task record should be displayed as the main content when opened.

3.1.2 Settings page shall write a local file (.json) with the information provided and read the same file on open to populate the fields.

3.1.3 Values entered on settings page shall be used as variables to make connections to remote server.

4.0 Search Functionality

4.1 Search bar shall query against Full-text indexes in MySQL database and return relevant results.

4.2 Results shall be displayed in drop-down form below search bar and should link to the relevant main-content view

5.0 Test Plan

How are you going to prove that you met the requirement by testing the design? Stay at the functional testing level. That is test functional areas. Unit testing is to be done by the developer and is normally not recorded.

Requirement #	Description
R1.0	User can launch program from any supported platform.
R1.1	User sees all applicable fields on Task form.
R1.2	User can create, update, and delete Task records.
R1.3	User can view Visual Task Board.
R1.4	User can create, rename, and delete Visual Task Board columns.
R1.5	User can move Task Cards between columns on Visual Task Board
R1.6	User can participate in real-time chat with other team members.
R1.7	User verifies that chat messages persist through repeating launches of the application.
R1.8	User can navigate to Settings page.
R1.9	User can edit values on the Settings page form.
R1.10	User verifies that Settings persist after saving.
R1.11	User can enter search queries in the search bar located at the top of the application.
R.12	User verifies that search results take them to the applicable view and/or record.

Appendix A. Test Results

The results of your step by step test plan. Record pass/fail.