Convert XUL to HTML

Mozilla

Project Name: Convert XUL to HTML

Project Mentor: Philipp Kewisch (Fallen)

Personal Details

Name: Arshad Khan

Email: arshdkhn1@gmail.com

Phone: +91-8090907372, +91-7065199979

Time Zone: UTC+05:30

IRC nickname: arshad

Github: https://github.com/arshdkhn1

FreeCodeCamp: https://www.freecodecamp.com/arshdkhn1

LinkedIn: https://in.linkedin.com/in/arshdkhn1

Website: http://iamarshad.com

Projects: http://iamarshad.com/projects

Open Source Contributions: Jenkins, Quran, Green Navigation, React Slingshot etc

About me: I am majoring in Information Technology from Guru Gobind Singh Indraprastha University, New Delhi and will be graduating next year. I have great interest in web development and web design. I have made nearly 30 web projects which demonstrate my skills and expertise in front-end, data-visualization, back-end and full-stack web development. I have also completed CS106A(Programming Methodology), CS106B(Programming Abstractions) and CS103(Mathematical Foundations of Computing) which are courses taught by Stanford University to Computer Science majors. I have completed them under the Stanford Engineering Everywhere scheme. In my free time, I also mentor web development newbies at FreeCodeCamp.

Certifications:

- Frontend Web Development
- Data-visualization
- MongoDB for NodeJS Developers

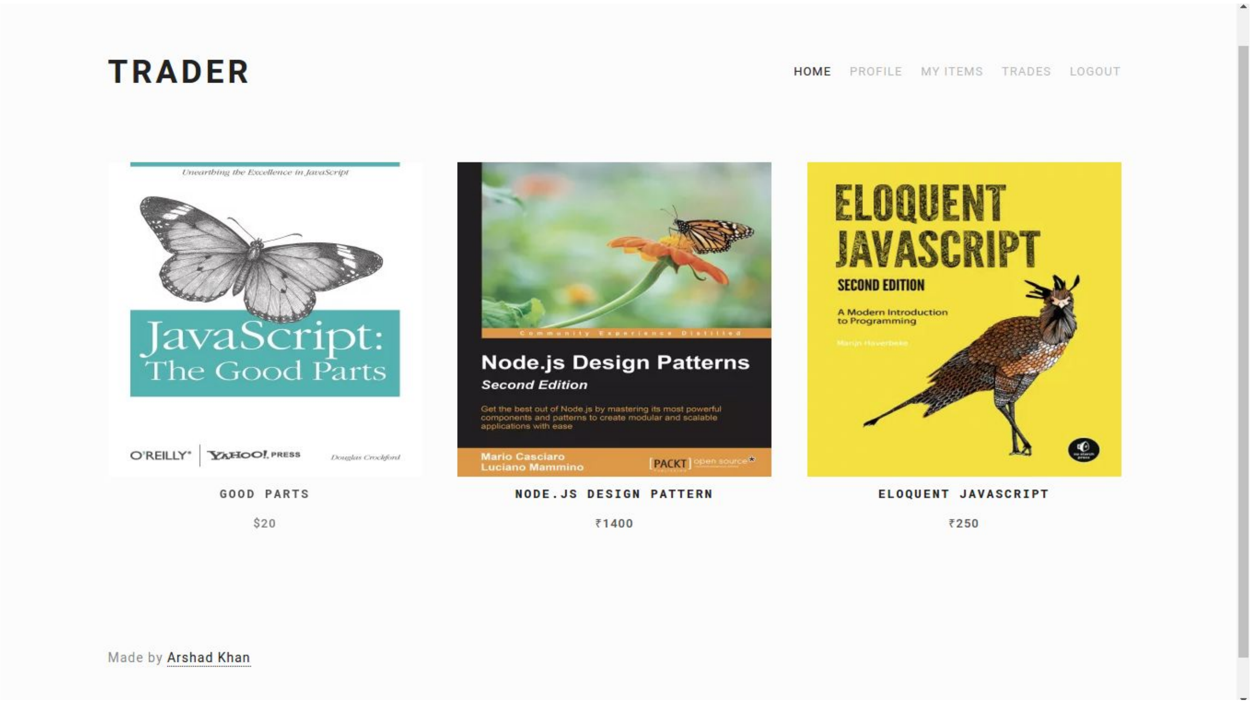
I am working on the last project after which I will get backend certification as well.

Links to projects category wise:

- Frontend projects
- Data-visualization Projects
- Backend and Api Projects
- CS106A
- All projects

Languages and Skills: C, Java, HTML, CSS, Pug(Jade), Sass, JavaScript, JQuery, ReactJS, Redux, D3.js, Node.js, Express, MongoDB, Webpack, Git, Passport

Top Web Projects



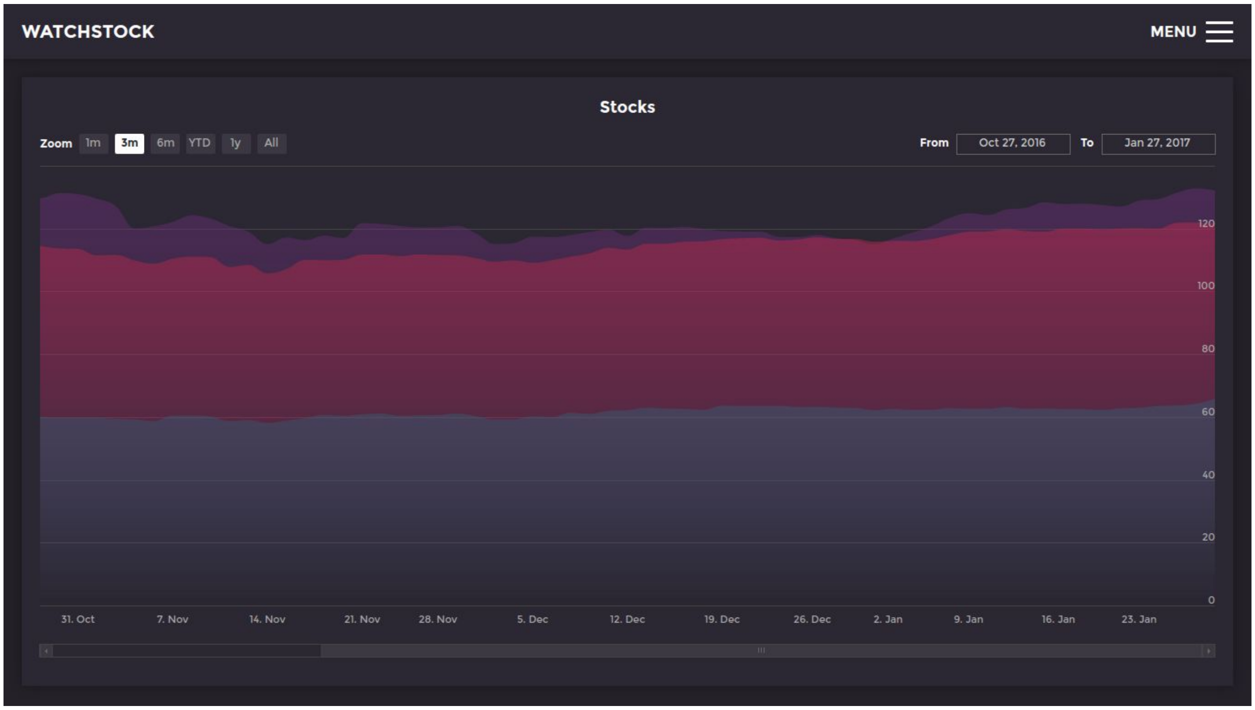
Trader

A beautiful app for trading items. User can add items, make proposal and accept/decline trade request for their own items.

Technology used: React, Redux, React-router, SASS, JavaScript, Node.js, Express, MongoDB, Webpack, React Slingshot, Flexbox, Social Authentication

Live App

Code



From Oct 27, 2016 To Jan 27, 2017

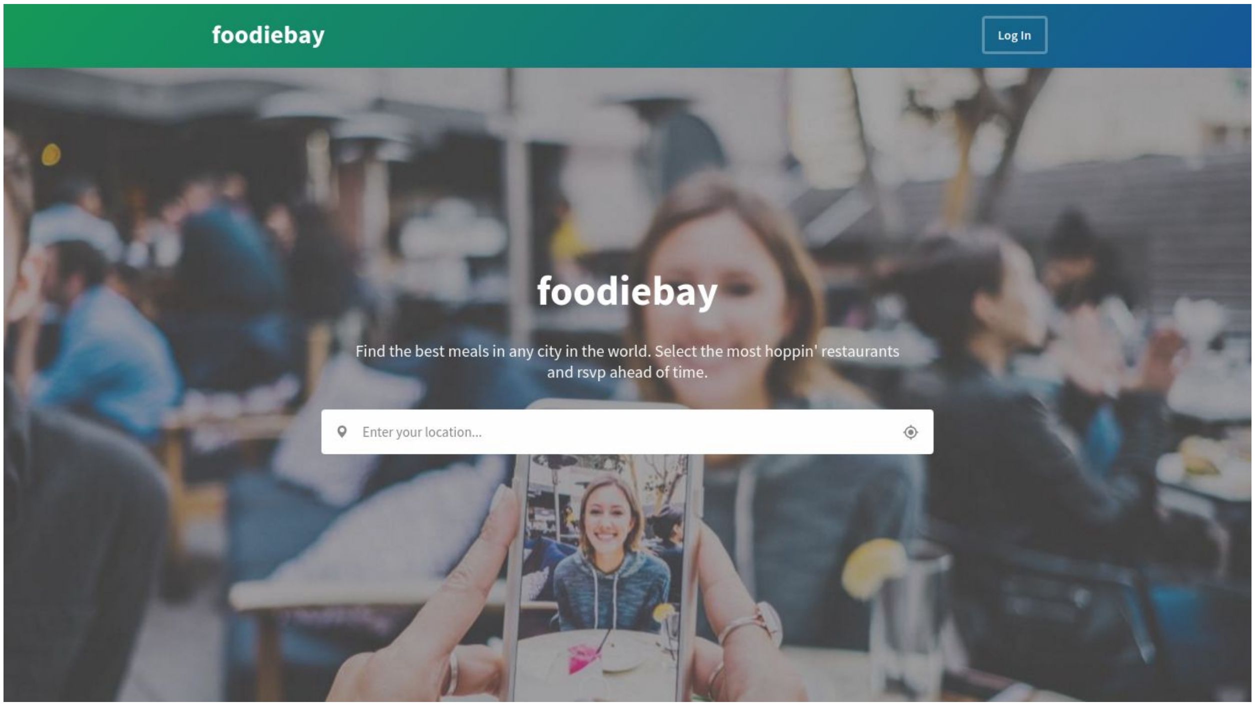
WatchStock

WatchStock is full stack MERN web app where user can view a graph displaying the recent trend lines for each added stock. User can add new stocks by their symbol name and can remove stocks as well. User will see changes and get notifications in real-time when any other user adds or removes a stock.

Technology used: React, CSS, JavaScript, Node.js, Express, MongoDB, Webpack, Socket.io

Live App

Code



Find the best meals in any city in the world. Select the most hoppin' restaurants and rsvp ahead of time.

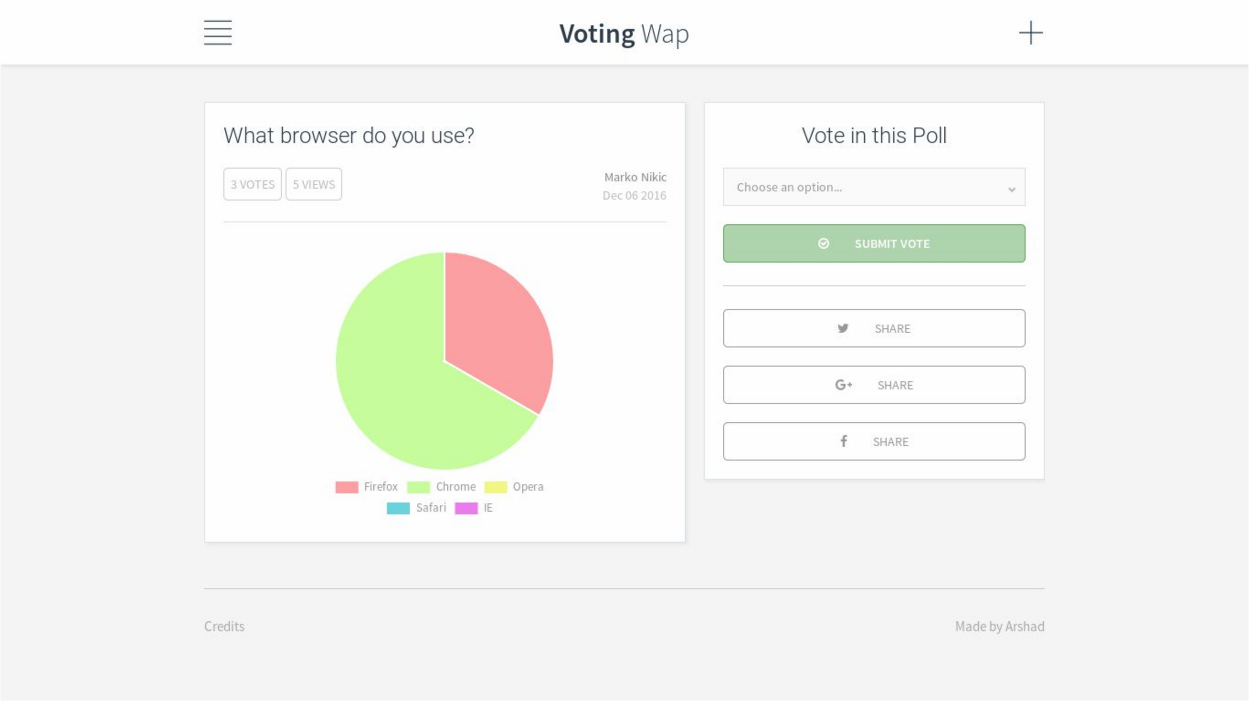
Foodiebay

Foodiebay is full stack javascript web app that finds restaurants in any city of the world with reviews from local experts. User can add himself to going-list of any restaurant and share it online with friends.

Technology used: Pug(Jade), Sass, JavaScript, Node.js, Express, MongoDB, Social Authentication

Live App

Code



VotingWap

Voting Wap is full stack javascript web app that enables user to create, delete, vote and share poll and view the results.

Technology used: EJS, CSS, JavaScript, Node.js, Express, MongoDB, Social Authentication

Live App

Code

Personal Availability

I have reviewed the GSoC 2017 Timeline and I am agreeing that I will be fully available without any constraints during the time mentioned there except in the first two weeks I will be partially off because of my final semester exams. I will surely cover that up by working on weekends.

Project Abstract

Lightning has traditionally been using XUL for its user interface. To modernize, I will be converting User Interface of some dialogs into React. The new components will be using web standards as much as possible, avoiding extensive use of third party libraries. I will be using React.js for developing HTML. React.js is very good for making reusable web components and it is quite efficient as well. Unlike XUL, debugging is very easy with react.js. The components that I have picked to convert into react.js are **Print dialog box, Calendar Alarm dialog box and Event dialog box.**

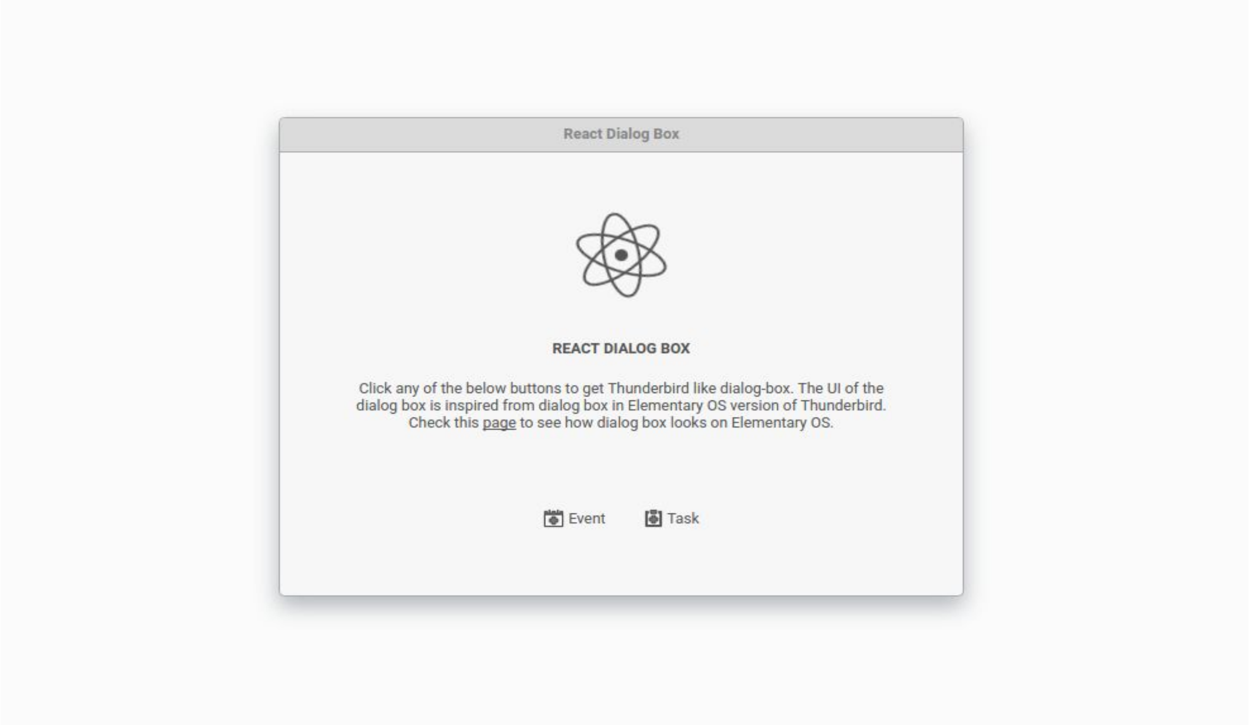
Short Description of Timeline

In the starting weeks of GSoC, I will be working on **Print dialog box** and then **Calendar Alarm dialog box** and then finally before one week or two weeks for mid-term evaluations, I will start my work on **Event dialog box**.

Many components like custom dropdowns, calendar dropdowns that are used in **Print dialog box** can be reused later in **Event dialog box** so I will utilise this time to make the layout of event dialog box like Mockup N.

The motive of this project will also be to make these **dialog responsive enough to utilise the space and adjust layout accordingly in both large and small screen width.** Therefore there will be **two breakpoints for the dialog box especially the Event dialog box** as the rest of dialog box that I have chosen doesn't show much components.

React Prototype



React Dialog Box

REACT DIALOG BOX

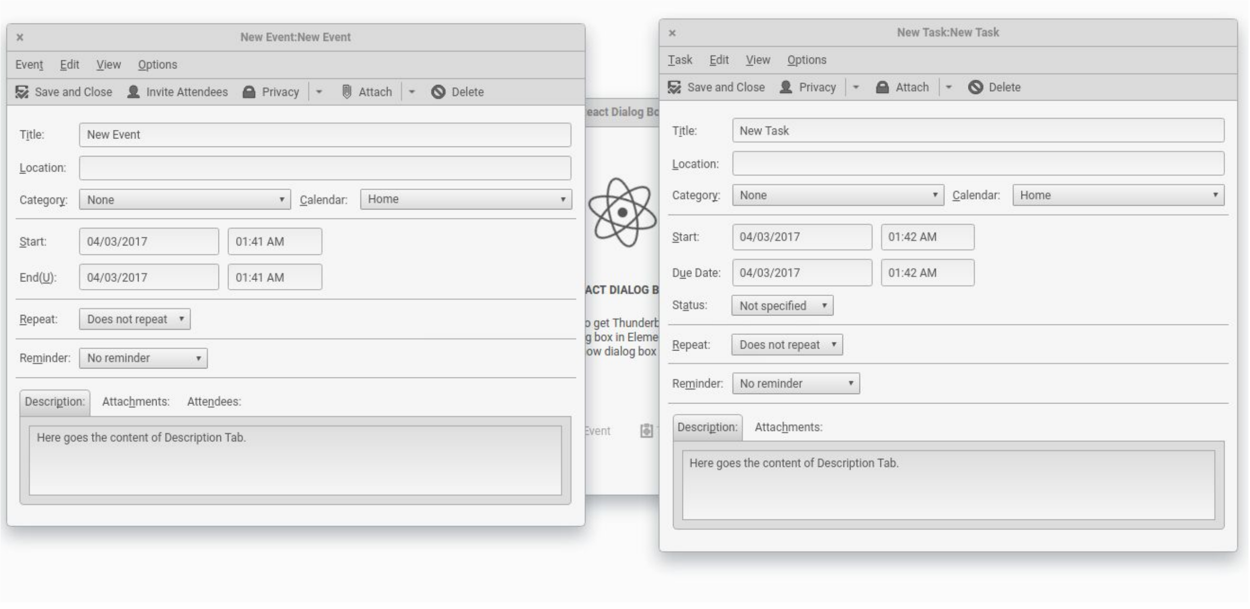
Click any of the below buttons to get Thunderbird like dialog-box. The UI of the dialog box is inspired from dialog box in Elementary OS version of Thunderbird. Check this page to see how dialog box looks on Elementary OS.

Live Prototype Demo

Code

I have made a react prototype for the event and task dialog box of calendar. It won’t be right to say this prototype is for any particular dialog box as this prototype uses config files and parse it into React UI. This whole prototype doesn’t not uses any third party libraries, it only uses **vanilla javascript, react.js and css.** I have used modal instead of new window for dialog box as removing location bar in window is not possible at browser level in all browsers.

This prototype have draggable modal and it mimics the dialog box of Elementary OS version of Thunderbird. Both of the dialog-box can be opened at same time and have separate states. Below are the few images of dialog boxes in action.



Title: New Event
Location:
Category: None
Calendar: Home
Start: 04/03/2017 01:41 AM
End(U): 04/03/2017 01:41 AM
Repeat: Does not repeat
Reminder: No reminder
Description: Attachments: Attendees:
Here goes the content of Description Tab.

Title: New Task
Location:
Category: None
Calendar: Home
Start: 04/03/2017 01:42 AM
Due Date: 04/03/2017 01:42 AM
Status: Not specified
Repeat: Does not repeat
Reminder: No reminder
Description: Attachments:
Here goes the content of Description Tab.

Event and Task Dialog Box

Proposed Detailed Timeline

Pre GSoC

- Learn XUL, XBL and XML.
- Understand the workflow of the calendar app
- Understand how the html is used in iframe and how xul communicates with it.
- If possible try to contribute to the Thunderbird/Lightning project.

Community Bonding

- Seek guidance from mentor about the workflow and how things work at Mozilla.
- Enrich knowledge about Calendar and integrate redux setup into the calendar app.
- If possible start coding in this phase, to get a head-start.

Week 1

Day	**Tasks**
Day 1	● Create XUL window which will wrap all the components of Print dialog box. ● Create basic UI of all the **components of dialog.**
Day 2	● Hook up controlled elements of with state to make them interactive. ● Improve the UI of components. Currently options in Print dialog box are a bit padded left. Utilise space to make them look better.
Day 3	● Implement the custom UI of **select box**. Default UI of the select box doesn't allow much styling for options. ● Implement custom UI of **radio buttons and checkboxes.**
Day 4	● Start creating custom UI of **calendar input.** Hook up a demo dropdown with it. ● Create UI of the top control bar of the custom calendar dropdown.
Day 5	● Create the rest UI of **Calendar body.** ● Hook up **month button in top control bar** with a dropdown having list of months.

Week 2

Day	Tasks
Day 1	● Hook up **UI of year button** in **top control bar** with a dropdown having list of years and other controls. ● Hook up the **days/week sections** of **calendar dropdown** with real calendar data as the no of days in months differs and.
Day 2	● Add styling to grey out the days of **next/previous month** to distinct from current month and highlight the selected date and current date. ● Hook up the **month changing control of the top control bar** with the days/week section.
Day 3	● Bind correct format value when date is selected from calendar dropdown. ● Create custom UI for the preview section for the **Monthly Grid.**
Day 4	● Create custom UI for the preview section for the **Weekly Planner** option of the Layout select box. ● Create custom UI for the preview section for the **List** option of the Layout select box.
Day 5	● Hook up the **What to Print** section's **Event and Task checkboxes** with the backend data. ● Hook up the **print preview** with backend data for the **List option of Layout select box.**

Week 3

Day	**Tasks**
Day 1	● Hook up the print preview with backend data for the **Monthly Grid** option of Layout select box. ● Hook up the print preview with backend data for the **Weekly Planner** option of Layout select box.
Day 2	● Hook up the Tasks checkbox of What to Print section with bottom section of Options like **Task with no due date**. ● Hook up the Tasks checkbox of What to Print section with bottom section of Options like Task with **Completed Tasks**.
Day 3	● Bind the **print button** and make sure it is working well. ● Bind the **cancel button** and make sure it is working well and all the selection of the dialog box are refreshed whenever print dialog is opened again.
Day 4	● Separate out components into **smart and dumb components.** ● Integrate the redux state into print options if needed.
Day 5	● Make sure all the components using redux state works well. ● Make sure **print preview section** works well with redux.

Week 4 (Evaluation Week)

Day	Tasks
Day 1	Buffer day for in order to accommodate any pending/overdue tasks from previous weeks. Head towards next day work as soon as done.
Day 2	Buffer day for in order to accommodate any pending/overdue tasks from previous weeks. Head towards next day work as soon as done.
Day 3	Buffer day for in order to accommodate any pending/overdue tasks from previous weeks. Head towards next day work as soon as done.
Day 4	● Create XUL window which will wrap all the components of **Alarm/Reminder dialog box.** ● **Create UI of components** of the Reminder/Alarm dialog box.
Day 5	● Improve the UI of components and buttons of currently selected tasks or events. They doesn’t stand out and blend with the background. ● Create custom UI for **Snooze Buttons with dropdowns.** ● **Submit Evaluations.**

Week 5

Day	Tasks
Day 1	- Hook up **Snooze-for and Snooze All buttons** temporarily with react state which will be later replaced by redux state if needed. - Hook up **Dismiss and Dismiss All buttons** temporarily with react state which will be later replaced by redux state if needed.
Day 2	- Make sure everything works fine. - Find bugs and fix them and then head towards the next day work.
Day 3	- Add redux store state into the components. - Hook up everything with backend data.
Day 4	Buffer day for in order to accommodate any pending/overdue tasks from previous weeks. Head towards next day work as soon as done.
Day 5	Buffer day for in order to accommodate any pending/overdue tasks from previous weeks. Head towards next day work as soon as done.

Week 6

Day	**Tasks**
Day 1	● Create XUL window which will wrap all the components of **Event dialog box.** ● Create **UI of the simple components** of the Event dialog box.
Day 2	● Create **UI for the bottom tabbed and other information section.** ● Start making custom dropdown for **Category dropdown** and hook it up with a demo dropdown.
Day 3	● Add **new category input component** to add other category options. Implement UI taking Mockup N as reference. ● Add **checkbox into each option** of dropdown.
Day 4	● Hookup dropdown temporarily with react state which will be later replaced by redux state. ● Start creating the **custom UI for dropdown of Time Input** and a demo dropdown.
Day 5	● Add **hours section** into the custom time input dropdown. ● Add **minute section** into the custom time input dropdown.

Week 7

Day	Tasks
Day 1	● Add **Expanded minute section** and **hook it up with the expand button to toggle it.** ● Make sure the format of time is right and correct time input is selected using the custom time input dropdown.
Day 2	● **Bound the Start Date with End/Due time input.** ● Disable the custom time inputs when all day events are not marked.
Day 3	● For the **repeat dropdown,** extend the custom dropdown with a new feature of adding a custom item. ● Hook it up with the **Edit recurrence** dialog box. ● The UI of the repeat dropdown will be improved by taking Mockup N as reference.
Day 4	● Reuse the dropdown with custom input option in **Reminder dropdown.** ● Hook it up with **Setup Reminders** dialog box.
Day 5	● Implement transitions between the different tabbed section like **Attendees, Attachment, Description** etc. ● Integrate **webpage attachment** into **attachment tab.** ● **Improve UI of the Attachment tab and items** by taking Mockup N as reference.

Week 8 (Evaluation Week)

Day	**Tasks**
Day 1	- Hook up the state with Attachment and add **extra items like explicit button to remove attachment** etc. - Integrate **Invite attendees dialog box** into **attendees tab.** - **Improve UI of the Attendees tab and items** by taking Mockup N as reference.
Day 2	- **Implement bottom checkboxes** of attendees tab. - Makes sure all the tabs are working well. - Add **priority section in the bottom section** as well using Mockup N as reference guide.
Day 3	- Add **privacy section i**n the bottom section and hook it up with state. - Add **priority section i**n the bottom section and hook it up with state. - Make sure everything is working well.
Day 4	- Add redux to handle state of custom category and calendar input elements and other items in the first section of event dialog. - Add redux to handle state of **custom repeat and reminder input elements** and other items in the first section of event dialog.
Day 5	- Hook up the **redux state into attachment tab.** - Hook up the **connection between the xul menu bar and react-redux attachment section**. - **Submit Evaluations.**

Week 9

Day	**Tasks**
Day 1	- Hook up the **redux state into attendees tab.** - Hook up the **connection between the xul menu bar and react-redux attendees section**.
Day 2	- Add the extra items that are present in menu bar but not directly in dialog using Mockup N as reference. - Hook up **privacy and priority options with xul menubar** and redux state.
Day 3	- Hook up **Show timezone as with bottom status element** in dialogbox. - Add redux state into these element and make sure they work well.
Day 4	- Improve the **UI of bottom options that are present in event dialog box to show status of some options in xul menubar e.g., Show time as , priority etc.** - Make the **upper section responsive on the** basis of endpoints discussed earlier and add some design change to improve the layout.
Day 5	- Make the **middle section responsive** on the basis of endpoints discussed earlier and add some design change to improve the layout.

Week 10

Day	Tasks
Day 1	- Work on the overall design of event tab to make UI better. - Make it as responsive as possible.
Day 2	- Test the Upper section and make sure it works well. - Fix bugs that are found while testing.
Day 3	- Test the Middle section and make sure it works well. - Fix bugs that are found while testing.
Day 4	- Test the Tabs section and make sure it works well. - Fix bugs that are found while testing.
Day 5	- Test the Tabular section and newly added options that were hidden in xul menu bar. Make sure it works well. - Fix bugs that are found while testing.

Week 11 and Week 12

Buffer week in order to accommodate any pending/overdue tasks from previous weeks. After completing previous work and fixing bugs, try to convert some other dialog box in the rest of the time.

Post GSoC

Work on the other issues that were left for the making the GSoC project more consistent and manageable.

Why am I the right person for this project ?

- By taking look at the projects that I have done over the last year, it will be evident that I have an eye for good design and I haven't exaggerated the skills that I have mentioned.
- React projects like Trader, WatchStock, other react projects and the react dialog box prototype shows my proficiency in **React.js** and **Redux** the skills that are required to complete this project successfully.
- I can completely work with javascript and css without using any third party library to complete this project and again the dialog box prototype shows it very well.
- I am keen learner and I have learned nearly all of the web development technologies that I use by self study so it won't be hard for me to learn anything new quickly.

Contributions

- Make bootstrap compatible with Elementary OS
- Other Contributions