

CS 320 Course Project Final Report

for

PDF Share & Schedule

Prepared by

Group Name: R&B

|  |  |  |
| --- | --- | --- |
| Ben Mikalenko | 11589806 | Ben.mikailenko@wsu.edu |
| Roman Stolyarov | 11689068 | Roman.stolyarov@wsu.edu |

|  |  |
| --- | --- |
|  |  |
| Date: | Friday, October 25th 2019 |

Contents

Contents ii

1 Introduction 1

1.1 Project Overview 1

1.2 Definitions, Acronyms and Abbreviations 1

1.3 References and Acknowledgments 1

2 Design 2

2.1 System Modeling 2

2.2 Interface Design 2

3 Implementation 3

3.1 Development Environment 3

3.2 Task Distribution 3

3.3 Challenges 3

4 Testing 4

4.1 Testing Plan 4

4.2 Tests for Functional Requirements 4

4.3 Tests for Non-functional Requirements 4

4.4 Hardware and Software Requirements 4

5 Analysis 5

6 Conclusion 6

Appendix A - Group Log 7

# 

# Introduction

*<TO DO: Please provide a brief introduction to your project.>*

## Project Overview

PDF Share and Schedule is an online application that allows users to upload PDF documents to a centralized location. PDF documents are grouped, shown on the home pag and organized by user inputted date. Through the upper menu GUIs, the user can add, delete, and edit the groups of PDF documents.

## Definitions, Acronyms and Abbreviations

Chord Chart – A form of musical document that lists a songs lyrics along with the chords played at each section

Date (in regards to the webpage) – An entry to the main webpage that has a real calendar date as a name and has zero or more PDFs attached and avalible for download.

GUI – General User Interface

PDF – Portable Document Format

Main Page – The div with the id of MainPage in the index.html where all the user inputted content is stored

## References and Acknowledgments

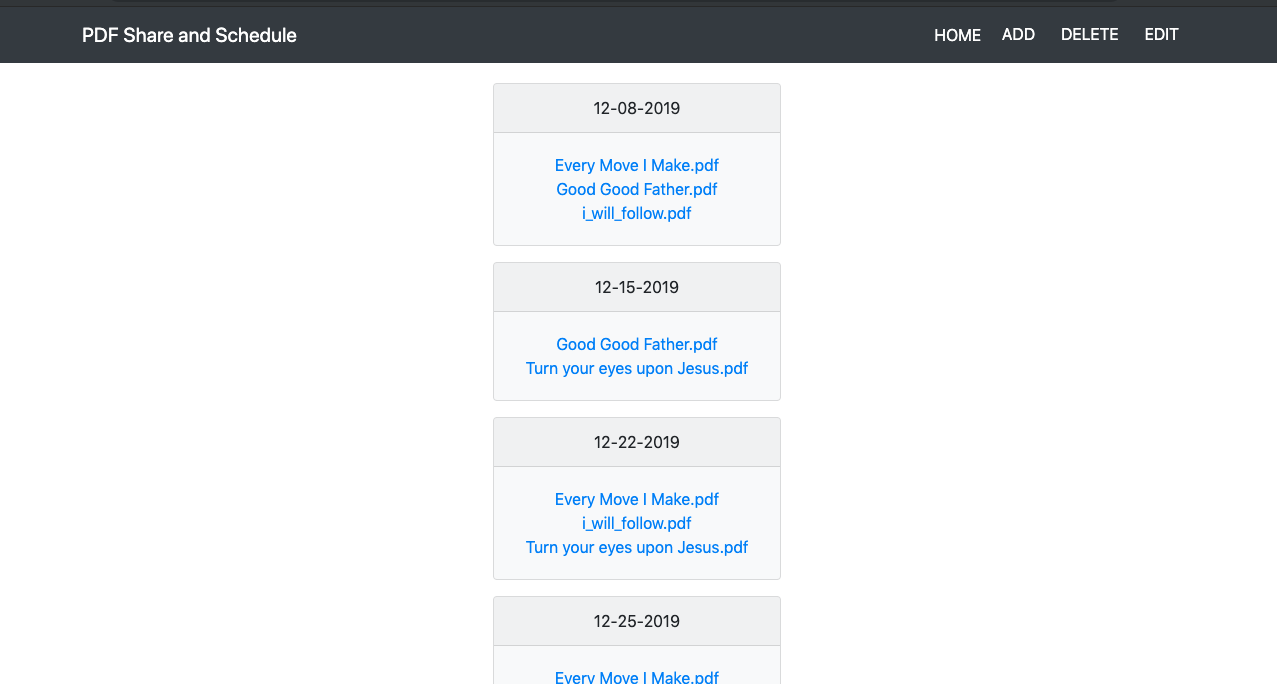
NO REFERENCES YET (USE IEEE CITATION IF WRITING A REFERENCE)

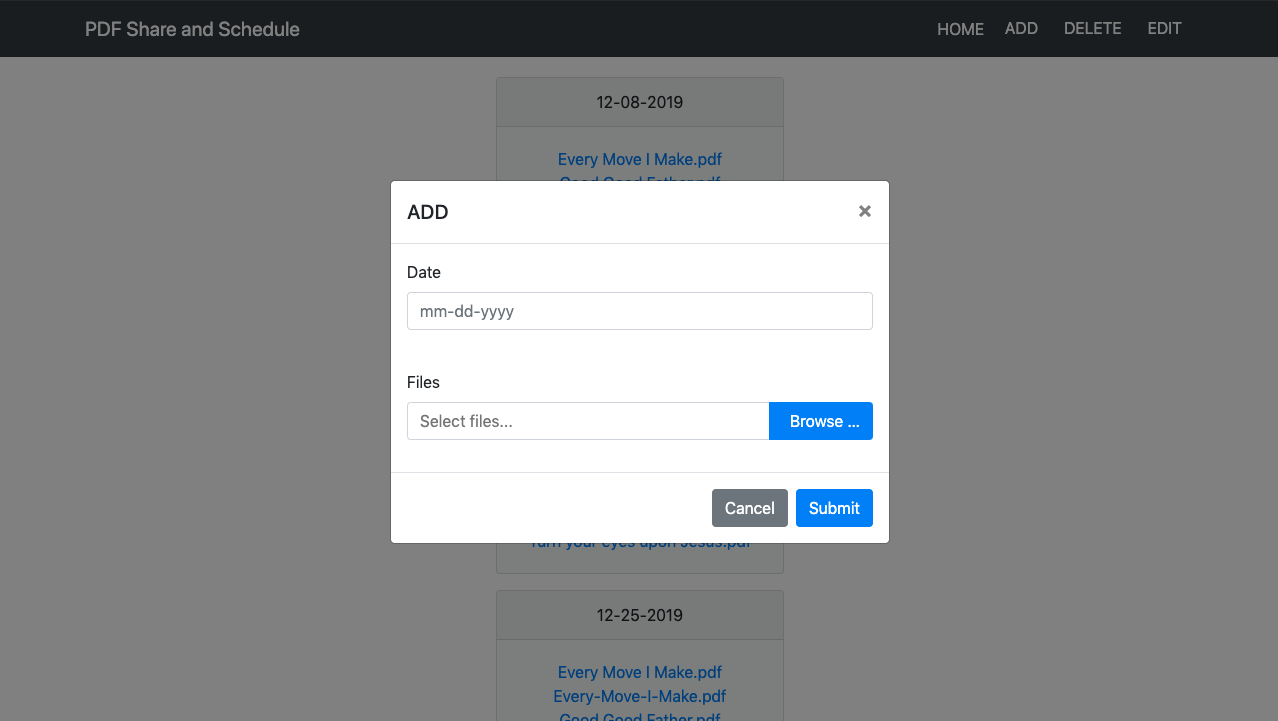
# Design

## System Modeling

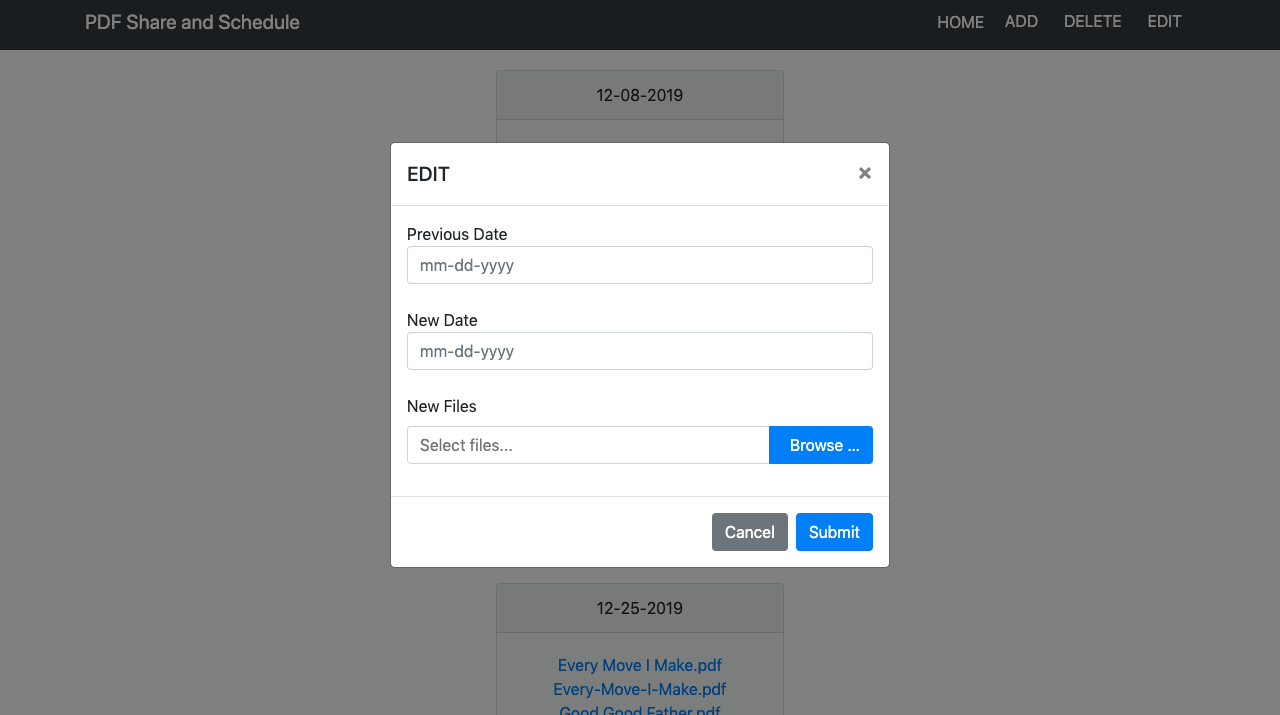
Our inplementaion strictly follows the design document in milestone 2. But note, in the design documents “structural modeling” section, “entry” and “main page” are reffered to the divs with id’s of dates and the div of id “MainPage” accordingly.

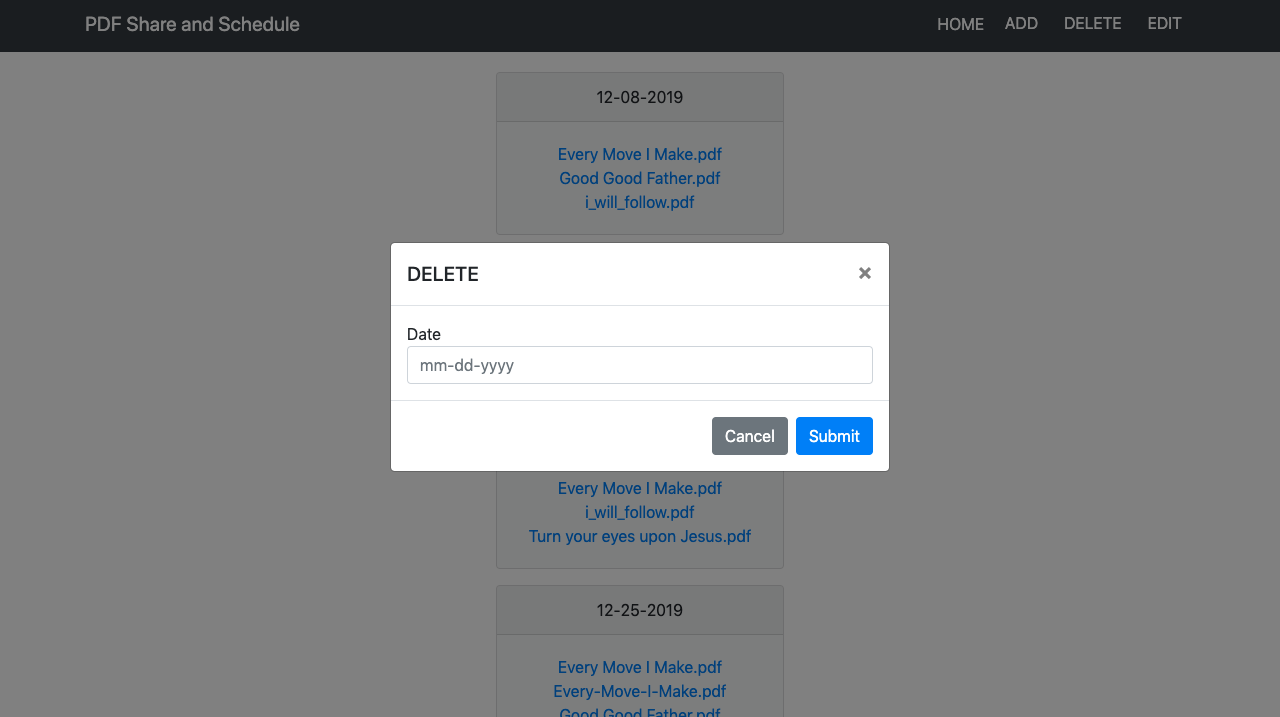
## Interface Design

Main page

Add GUI

Edit GUI



Delete GUI

# Implementation

## Development Environment

HTML and Javascript was used for the implementation. Plenty of UI frameworks like bootstrap, bootstrap-datepicker, and bootstrap-fileinput were used inside the of the HTML. Node js + express was the main web application framework used for the backend.

UI Framworks used:

Bootstrap – Main layout and look of the page, buttons, and main page entries (bootstrap cards)

Bootstrap-datepicker – Calendar pop-pop in the GUIs date input field

Bootstrap-fileinput – Allowed for multiple file input from only one input field

Javascript framworks used:

Node js – Runtime enviornment

Express js – Application framework

Express-fileupload – Allowed express to easily put uploaded files into the req.body.files

Mv – Moved uploaded files to local directory

Fs – Allowed the use of the local directory

JsDom/JQuery – Allowed the manipulation of the HTML files

Del – Deleted entire folders quickly

Multer – Allowed us to use multipart form data

## Task Distribution

*<Describ how the implementation tasks are distributed among team members.*

*TO DO: For each team member, describe his/her main implementation tasks in this project.*

*If this is a one-person project, mention: “all the work presented here is done by \*\*\* (your name).” >*

## Challenges

The most challenging part was to actually upload the files to the application directory along with the date (text) input. Multipart data was being inputted from the forms, but express-fileupload only received the files and not the date input. We used multer to receive the text input and express-fileupload to receive the files.

# Testing

## Testing Plan

Testing will begin once the backend implementation is complete. Estimated time of testing is the last week of November 2019. Things to be tested:

1. The application redirects with no errors or crashes to the home page when the submit button is hit and there are no files inputted.
2. The application deletes all the selected files from the applications “files” directory from the delete GUI.
3. All entries to the main page are organized by date at all times.
4. 1+ files can be inputted and uploaded successfully.

## Tests for Functional Requirements

<Describe your test results for the functional requirements.

TODO: Provide a list of use cases or functions you have tested, as well as the testing results (whether or not the system passed the tests).>

## Tests for Non-functional Requirements

<Similar to the Section 4.2, but this section is for the non-functional requirements. >

## Hardware and Software Requirements

<Describe the hardware and software requirements for performing the tests. >

# Analysis

<In this Section you need to analyze the effort that has been put on this project.

TODO: Describe how many hours (approximately) each team member spent on the project, for each milestone, which milestone takes the most effort and why. >

# Conclusion

<Conclude the document with what you have learned through working on the project.>

Appendix A - Group Log

< Describe how frequently the group meembers meet during the semester, and how effective the communication is. This is optional for one-person projects.>