

CS 320 Course Project Final Report

for

SubScruple

Prepared by Kevin Black

Group Name: BBS teemMaets

|  |  |  |
| --- | --- | --- |
| Shane Bellika | 11639152 | shane.bellika@wsu.edu |
| Kevin Black | 11480710 | kevin.black@wsu.edu |
| Alex Sloan | 11651363 | alex.sloan@wsu.edu |

December 5, 2018

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

**SubScruple** is a free, web-based application focused on letting users manage their personal subscriptions for a variety of services and was developed and managed by BBS teemMaets. The user is able to enact different actions while using this application, such as: view a multitude of services listed by the application, adding subscriptions to their own profile, viewing all their subscriptions and a running total for how much they spend monthly/yearly, editing their profile, and checking notifications which, depending on the user’s respective location, they would receive notifications about newly available services or different deals their services are currently offering.

## Project Overview

While the project has functionally changed since the initial ideas stated in the respective software requirements specification document, the sole purpose of this project stayed pretty much the same. The latter being that we wanted to create an application that could give users the possibility to manage their subscriptions in one place and being able to keep track of how much they spend as a result. In this brief section, you were introduced to a summary of our finalized version ofour application thus far, as well as any terms that need defining and references/acknowledgements utilized for this document. The remaining sections of this document will go indepth into the various steps of the development process, such as: the seemingly constant changing design of the application throughout its early stages, interface designs, certain development environments/techniques, how we went about testing our application, and discussing the challenges and future associated with this project.

## Definitions, Acronyms and Abbreviations

|  |  |
| --- | --- |
| 1. CS 320 | Fundamentals of Software Engineering, of which this document and application were developed at Washington State University under the tutelage of Professor Xinghui Zhao. |
| 2. IDE | Integrated Development Environment, where the programming is performed. |
| Meteor | Web application framework written using Node.js, with one of its key features being that web applications don’t need manual refreshes to display changes to in application. |
| MEAN | A development technique composed of MongoDB, Express.js, AngularJS, and Node.js. |

## References and Acknowledgments

* Standard IEEE Citation Guide:

<https://learn.wsu.edu/bbcswebdav/pid-2795651-dt-content-rid-91333015_1/courses/2018-FALL-VANCO-CS-320-7661-LEC/IEEE-Citation-StyleGuide.pdf>

* Slide reference material provided by Professor Xinghui Zhao of WSUV:

<https://learn.wsu.edu/webapps/blackboard/content/listContent.jsp?course_id=_189425_1&content_id=_2750588_1&mode=reset>

* Application coding standard (for JavaScript): <https://eslint.org/>
* Application development environments: <https://www.jetbrains.com/idea/?fromMenu>

# Design

## System Modeling

Despite quite a bit of changes in some functionalities in our original take of the application, the general system modeling diagrams created for this project in the software design document have stayed pretty much the same. As much as our team wanted to implement the application following the goals we set in place in our second milestone, some features were just not feasible for our time frame and our experience. The few exceptions for the system modeling diagrams that changed can be found below.

## Interface Design

<Provide several screenshots to illustrate your interface design.

TO DO:

For each subsystem, pick one or two representative screenshots and paste here.>

# Implementation

## Development Environment

**SubScruple** was developed using two certain assignments from CS 320 as a foundation for the application, since a lot of the elements contained in those assignments were very similar to what we desired in our application, saving on time and utilizing reusability. The application was programmed in three development languages: JavaScript, HTML, and CSS. The IDE used for the entirety of this development was Intellij, as being able to enforce the coding standard provided by professor Xinghui Zhao was easy to install and follow in realtime in this IDE. Distributed version control was of the utmost importance for communication of progress between group members and was done by using a public repository containing all the files for the application on GitHub. While not an absolutely necessary web application framework to work with for this project, we used Meteor due to its use of a MEAN stack, no HTTP request response cycle, and access to a local, client-side database. Meteor was ran on Windows 10 through the administrator version of Windows PowerShell.

## 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

*<This section is optional. Describe the challenges in the implementation, if there are any, and how you dealt with them.*

*TO DO: If you don’t have anything to fill in, just leave this section blank.>*

# Testing

## <*This section is a summary of your testing report>*

## Testing Plan

<Describe your testing plan for the project.

TODO: Give a list of items or functions you want to test, and also a schedule for performing the testing. >

## 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.>