

CS 320 Course Project Final Report

for

Expense Tracker

Prepared by

Group Name: KVC Inc.

|  |  |  |
| --- | --- | --- |
| Chayse Jorgenson | 11678623 | Chayse.jorgenson@wsu.edu |
| Kevin Tong | 11686700 | Kevin.tong@wsu.edu |
| Vitaliy Kudrik | 11603449 | Vitaliy.kudrik@wsu.edu |
|  |  |  |
|  |  |  |

|  |  |
| --- | --- |
|  |  |
| Date: | <place the date of submission here> |
|  |  |
|  |  |
|  |  |

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.>*

The Expense Tracker is a web application that helps users analyze their financial information through spending habits and aids users to formulate financial saving goals.

## Project Overview

< A brief description of the project.

TO DO: Write 1-2 paragraphs describing the project. >

The Expense tracker is a useful tool to offer users insight into managing their finances more efficiently by displaying user input financial information in an easy to read format. The information displayed contains a graph of different expenses,

Once the expenses have been totaled and displayed to the user, Expense Tracker also displays tips or recommendations on where the user may need to adjust their finances or show the user where they have been doing well with their finances.

## Definitions, Acronyms and Abbreviations

<Define all the terms necessary to properly interpret the report, including acronyms and abbreviations.

There are no acronyms or abbreviations in this document.

## References and Acknowledgments

<List any other documents or Web addresses to which this document refers. These may include user interface style guides, contracts, standards, system requirements specifications, use case documents, or a vision and scope document.

TO DO: Use the standard IEEE citation guide for this section.>

# Design

## System Modeling

< Update your UML diagrams in milestone 2, to reflect the real implementation of this software.

TO DO: Provide an updated version of the UML diagrams, including use case diagrams, sequence (or state) diagrams, activities diagrams, and class diagrams. If you don’t have an updated version, just mention: “our implementation strictly follows the design document (milestone 2)”. >

## 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

<Describe the development environment you were using for the project.

TO DO: List the programming lanagues, IDEs, tools, etc.>

IntelliJ IDE, Notepad ++, Javascript, HTML, CSS, Semantic UI

## 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).” >*

Chayse:  
  
Kevin:

Vitaliy: Html and CSS, also writing on the SRS and some of the final report.

## 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.>*

Vitaliy: Just trying to get everything lined up and working properly was a challenge in the beginning but it got better later on. The biggest challenges was writing the SRS and the final report, because I just don’t know what to expect and if what we actually wrote is up to standard or not.

# 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. >

Vitaliy: The SRS took a few hours to write up(Just my parts alone), because it was the initial document without any real tangible thing to work with/base on. The html and css also took a few days, mostly because there were lots of iterations of the final look. There were also lots of changes in how we presented the data which required a few extra hours of work. Also getting all the html elements everything to actually go where you want it to go was kind of a challenge. I believe the SRS and the Final report were the hardest because they’re the only things, I personally can’t verify if we are doing correctly or not. Mostly because we can see the website functioning properly and the testing showed that there were no errors, but we don’t know if the actual technical documents are up to par, because we haven’t had them graded yet.

# Conclusion

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

Vitaliy: Through this project, I learned the intricities of teamwork, and I also got a glimse of the actual work a computer scientist has to do, outside of just normal coding. I learned that writing SRS documents was not the most fun thing to do, and also learned that html does not like beginners. I tried out many different frameworks and styling packages that all were very interesting and showed me the possibilites of JS and frontend coding. Frameworks like Meteor JS, and frontend tools like Semantic UI. Actually writing a website and having a use for it, has opened up my eyes to the world of html and javascript. I’ve gotten a lot of ideas that would work extremely easily with a webpage and some javascript, that I can know implement, due to the knowledge gained from this 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.>