**SEIS752 Advanced Web Design**

**Spring 2015**

**Project Report**

**‘Vacation Planner’**

<http://www.dreamer42.com/vacation/index.php>

<https://github.com/dreamer42/SEIS752-Vacation>

**Mike Palmer & John Stark**



**Table of Contents**

# [Introduction](#introduction)

# [Description](#description)

## Purpose

## List of web pages

## Business Logic

# [Technologies employed](#technologies)

## Languages (Html, javascript, php, JSON…).

## External API’s (Google maps)

## Database (MySQL, tables created)

## IDE

## Site Host (dreamer42 … , XAMPP – LocalHost, )

## Audio and Video

# [Scope](#scope)

## In-scope

## Out-of-scope

## If We Had More Time We Would …

# [Software Development Process](#swprocess)

## Agile

## Start small, code a little- test a lot

## Repository

## UI testing

## Load testing

# [Technical challenges encountered](#techChallenges)

# Appendices

## [Appendix A](#appendixA) – Project Proposal

## [Appendix B](#appendixB) – Screen shots of user accessible pages

# Introduction

This project provided an opportunity to apply the lessons learned during the course and to develop an application to a greater extent that was possible in the weekly homework assignments. The application addresses the need to store, organize and access important information related to a vacation. Saving reservations, confirmations, phone numbers, etc. in this secure website makes them accessible by several means. They cannot left behind, blow out the car window or be eaten by the dog as a paper list might. Items needing further attention can be highlighted to provide a visual reminder.

This report assumes the reader has visited the website to gain some familiarity with it. The site is at: <http://www.dreamer42.com/vacation/index.php>

# Description

## Purpose

This project developed and deployed a website for recording vacation plans. The user begins by registering or logging in, creating a vacation (which establishes a unique ID for it in the database) and then entering their daily itinerary. While planning the vacation the user can enter, change or delete information such as reservations, confirmation numbers, phone numbers and email addresses for lodging, sites to visit, daily destinations, etc.

The user can then reference and update their itinerary during their travels. In addition, the website’s business logic can provide some analysis of the travel plans. A google map is conveniently accessed by the click of a button with no need to re-enter location information.

## List of webpages.

The user interacts with some 9 webpages.

* [http://www.dreamer42.com/vacation/**index.php**](http://www.dreamer42.com/vacation/index.php)
  + Login via a drop down box then 🡪
  + Register [http://www.dreamer42.com/vacation/**register.php**](http://www.dreamer42.com/vacation/register.php)
* [http://www.dreamer42.com/vacation/**welcome.php**](http://www.dreamer42.com/vacation/welcome.php) Chose to create a new vacation or work with an existing one
* [http://www.dreamer42.com/vacation/**createNewVacation.php**](http://www.dreamer42.com/vacation/createNewVacation.php)
  + Enter vacation name and start date 🡪 enterDayDetails
  + [http://www.dreamer42.com/vacation/**enterDayDetails.php**](http://www.dreamer42.com/vacation/enterDayDetails.php)
* [http://www.dreamer42.com/vacation/**listOfVacations.php**](http://www.dreamer42.com/vacation/listOfVacations.php) choose a vacation to work with
* [http://www.dreamer42.com/vacation/**vacationSummary.php**](http://www.dreamer42.com/vacation/vacationSummary.php) edit, delete or add a row
  + Edit via <http://www.dreamer42.com/vacation/enterDayDetails.php> )
  + View Map [http://www.dreamer42.com/vacation/**displayMap.php**](http://www.dreamer42.com/vacation/displayMap.php)
  + Analyze travel plans via [http://www.dreamer42.com/vacation/**displayAnalysis.php**](http://www.dreamer42.com/vacation/displayAnalysis.php)

## Business logic.

The SQL queries provide some business logic by selecting only the appropriate information. The displayAnalysis.php page is backed by logic in getAnalysis.php page that compute and analyze the daily travel distances based on the starting locations and destinations entered. The results of the analysis is provide to the user textually and visually. The distances traveled determines which .jpeg image is loaded (hammock on the beach, packed car on the road, Oregon Trail graphic).

Although it’s not business logic, a random number is generated when the user clicks on the car image on the Index.php page. That random number then indexes into an array of audio clips to select one of them to play.

# Technologies employed

## Languages (Html, javascript, php, JSON…).

We used PHP, JavaScipt, and HTML as we did in other homework assignments. In addition we used:

* + - Google Map API (see below)
    - Bootstrap
      * Has built in resizing, formatting, colored buttons, …
    - Bootbox
      * Modal message boxes built on bootstrap and jQuery
    - Query parameters ….
      * better security and special characters
    - JSON
    - Login using encryption and SALT for storing password.
    - SQL

## External API’s (Google maps)

We used the Google Map API in our website. Our integration provides the user with a map showing their travel route, and computes their travel distances. The map informs the user of their route and may highlight any gross errors in the entered starting points and destinations.

## Database (MySQL, tables created)

We used MySQL which is part of the XAMPP stack, and is available on the remote host. The MySQL interface was very user friendly with an intuitive GUI. We usually used this interface to develop the queries we needed in our PHP code. This reduced the potential of introducing errors in the code.

We maintained a set of .sql files that could drop the data tables and recreate them. This facilitated a reset to the database after a messing coding/debugging session, and by resetting, we could keep our local database copies in sync. It also provided some disaster recovery.

## IDE

John had a fair amount of experience with Eclipse and chose NetBeans to gain some experience with another IDE. The only problem he experienced was that NetBeans did not catch as many JavaScript errors as Chrome’s Inspector did.

Mike used IntelliJ.

## Site Host (dreamer42 … , XAMPP – LocalHost, )

For local website hosting we used XAMPP. We did our development locally and had no problems with XAMPP.

Our remote host was at <http://www.dreamer42.com/vacation/index.php>. We used FileZilla to upload our files to the remote host.

# Scope

## In-scope

We implemented all of the key features in our proposal.

* Create User Account
* User Login
* Create New Vacation
* Display Vacation
* Update Existing Vacation
* Delete Vacation
* Store Core Information about Each Day

And we implemented several or the bonus features in our proposal.

* Calculate Mileage Traveled Each Day
* Provide a Map Showing Route
* Provide Reports about Trips
* Color Code Vacation to Alert to Areas of Concern

In addition we partially implemented the remaining three bonus feature in our proposal.

* Store Addition Advanced Information
* Security Measures Beyond Basic Login
* Polished UI

## Out-of-scope

There are many things that would still need to be done for a commercial website

Out of Scope items included

* + Security audit / refactoring
  + Refactor to add accessibility
  + Review fonts, CSS, for consistency page-to-page
  + More UI testing
  + More performance and scale testing

## If We Had More Time We Would …

Do more

* Security testing including use of an automated tool like Vercaode.
* More load and scale testing.
* More CSS work to improve the appearance.
* More mobile friendly design and then some mobile testing and rating.
* Captcha could be added to the register screen.
* Twilio could be added to send alert messages

On the registration screen the next feature to add is a captcha. This would improve security by reducing bots being successful in creating accounts. It would also provide an opportunity to incorporate another external API into the application.

The other feature that could really take the user experience to the next level is Twilio notification. This notification can come in a variety of ways. Automated messages (if you opt in via a configuration page) saying X number of days until vacation starts. Another way that would be very helpful would be user requested reminder. Some places like National Parks have reservations 6 months in advance. It would sure be nice to get a text reminder saying “reservations for day X of your vacation at Yellowstone are starting at 8:00 a.m. tomorrow”. This could even be a prompt when you put status of a day to “yellow” asking if you want follow-up reminder. Once again, an opportunity to leverage an external API to make the site more robust.

# Software Development Process

## Agile

We strove to set a realistic scope for our project, and applied Agile programming methodologies.

We had weekly iterations with a brief planning meeting after class on Wednesdays. Daily scrums (in person, or via email) provided coordination on task assignments and a chance to collaborate on solving problems. This worked well with our philosophy of advancing by small incremental changes. Since we both work in the same building at Thomson Reuters, stand-up meetings were easy to arrange and attend.

## Start small, code a little, test a lot

Our coding practice was to start small, make it work, and then expand on it. In the long run, it was faster and easier to make small changes, test them, and check them in. Any introduced bugs were restricted to a small section of code, which greatly simplified debugging. And each refactor focused on solving one problem; that of enhancing working code with one new feature.

For example, we needed to save *status* values somewhere on the enterDayDetails page, so that a javaScript function could pass them to another PHP page for use with queries. But displaying these numeric values to the user would be confusing. We solved the first problem by adding input fields to the html form.

<input id="morning\_status" name="morning\_status" value="testValue" style="color:blue">

And then after we had the queries working. We refactored and added a hidden attribute

<input type="hidden" id="morning\_status" …. style="color:blue">

Mike’s early insistence on this approach was key to our success.

## Repository

GIT worked great! We created separate sub directories for image (.jpg), audio (.mp3), document (.docx & .pptx), SQL, and library (.ccs & .js) files. We made a practice of frequently syncing with the remote repository to keep our local working copies up-to-date.

Creating an MKLink allowed us to point GitHub and our IDE to the same set of local files. It took a little research and a couple tries to get things set up properly. Here is an example of that activity that resulted in great time savings and less errors (for forgetting to move files) when updating code:

C:\xampp\htdocs>mklink /J c:\xampp\htdocs\vacation "C:\Education\St. Thomas\SEIS752\SEIS752-Vacation"

We did discover that GIT was unable to merge the binary files created by Microsoft Word. Next time we would probably use Google Doc for the report.

## UI testing

We had two sets of UI testing. In the first set Mike had his middle school children try out the website; they had no problems navigating, registering, entering data, etc. This was an informal review and no logs were kept. In the second set, we asked class members to ‘test drive’ the website as part of our project presentation. We asked them to enter some feedback in the last row of ‘vacation’ data they entered. There were no requests for assistance in using the website, and all class members provided feedback (shown below). There were no surprises in the feedback; we were working on adjusting the red shading to make it lighter but not pink. The suggestion to add collapsible areas for phone numbers, links, etc. was something we considered when scoping the project - we decided to start with simple text areas. Overall, the fact that everyone could use the site unassisted speaks for itself, and the comments seemed appropriate for a term project completed in a short time.

Good Features:

* Nice work, the integration of Bootstrap works well
* I can see myself using this app a lot!
* I love planning everything and also having a lot of options.
* It's nice to have the app calculate some things for you. Nice to have the map
* Workflow through application is straightforward. Mapping is effective.
* Great usage of storing previous data for starting location and user experience
* Liked Google Maps use --Multi-point trips-- good for road trips

Room for Improvement:

* It would be great to add more information that can be collapsible. ex: add notes, phone numbers, links to the hotels, confirmation pages images.
* I am very picky about presentation and appearance. I would work a little bit on the layout. But(!) I understand
* Functional, but maybe larger text entry fields and a more grid-like layout would make data entry a bit easier. Defaulting the ending location to the same value as start location would be nice.
* User Interface can be improved by positioning elements in different ways.
* Don't like the red background on text field -- hard to read/see

## Load testing

The class ‘test drive’ during our presentation demonstrated that the site was able to handle at least 8-10 concurrent users. We understand that this is a very light load, but we did include it in our development process.

# Technical challenges encountered

Neither of us had much experience developing websites, so finding the technologies and then learning how to use them was our first technical challenge. We applied the PHP, Javascript, etc. we learned in class. Google searches for solutions led us to Bootstrap, Bootbox, parameterized queries, etc. which were new to us and required some research to learn how to use them.

Our research led us to some icons we used instead of the basic grey rectangular buttons. This improved the readability and appearance of our pages. We also found some built-in color coded buttons in bootstrap that we employed to provide a means of setting the status for activities.

NetBeans IDE did not always show errors, Chrome Inspector was better at this.

Bootstrap versions were not forward/backward compatible.

Javascript is very picky and not informative about errors

**APPENDIX A.**

**SEIS 752 – Final Project Proposal**

**Team Members:**

* Mike Palmer
* John Stark

**Project Description:**

The website resulting from this project will provide a traveler with a way to organize their travel plans. Users will be able to create vacation itineraries. This will help them during the planning process as they assemble the pieces. It will also be a helpful tool on the vacation to reference reservation numbers and chart the progress of their trip.

**Key Features:**

Create User Account

User Login

Create New Vacation

Display Vacation

Update Existing Vacation

Delete Vacation

Store Core Information about Each Day

**Bonus Features:**

Store Addition Advanced Information

Calculate Mileage Traveled Each Day

Provide a Map Showing Route

Provide Reports about Trips

Color Code Vacation to Alert to Areas of Concern

Security Measures Beyond Basic Login

Polished UI

**User Stories:**

#1 - planning

User wants to plan a vacation. User enters each day into the system. Fills in starting city, ending city, main activity for the day, and lodging info.

#2 – vacationing

User references the app each day. Retrieves their hotel reservation info. Reviews the activities for the upcoming day.

**Mock-up of Possible Screen**



**Challenges**

Using an external API to calculate travel time between two cites by driving.

Ensuring data is validated.

Preventing security breaches.

**Project Timeline**

|  |  |  |
| --- | --- | --- |
| Event | Start Date | End Date |
| Prepare Proposal | 2/18/2015 | 3/4/2015 |
| Gather Additional Requirements | 2/25/2015 | 3/11/2015 |
| Database Table Created | 3/11/2015 | 3/18/2015 |
| Login Screen | 3/11/2015 | 3/18/2015 |
| Display Single Vacation Day | 3/11/2015 | 3/25/2015 |
| Display Multiple Vacation Days | 3/11/2015 | 4/1/2015 |
| Entry of Single Vacation Day | 4/1/2015 | 4/8/2015 |
| Entry of Multiple Vacation Days | 4/1/2015 | 4/15/2015 |
| Add, Update, Delete | 4/8/2015 | 4/22/2015 |
| Bonus Features | 4/15/2015 | 5/6/2015 |
| Testing & Fixing | 4/15/2015 | 5/12/2015 |
| Launch! |  | 5/13/2015 |

**Tentative Technology Plan**

LAMP stack

GIT

MySQL

HTML

PHP

Javascript

**APPENDIX B. Screen shots of pages**

























