Invicta: Project Report

Leesa Marie Price, Matthew Minish, & Grayson Martin

Capstone Project Fall 2021

The University of West Florida

November 29, 2021

CIS 4595 & Capstone

Dr. OK

Table of Contents

1 Executive Summary

During this past summer, many students traveled around the United States and outside the States. Upon discussing with these individuals, it became apparent that the rapidly changing status of different locations was hard to keep up with. People wished that there was some sort of system they could subscribe to that would notify them of the different steps they must take to continue traveling instead of searching through many news articles and press releases. Thus came the idea of a COVID19 travel web application.

2 Final Requirements vs Initial Requirements

The requirements stayed mostly the same with some minor changes.

2.1 Initial Requirements

We will have an Angular-based web application that is connected to the TravelAPI as well as a database that will store user information and trip information. Our web application will have an API key to the API and will only exchange information about different countries and their travel restrictions without linking it to a user. On the database side, we will have different datasets on a SQL database. One will have a user ID and trip IDs. One will have the trip ID and trip information.

User Stories:
As a user, I would like to be able to add my travel plans.
As a user, I would like to be able to opt in for text message updates.
As a user, I would like to be able to opt out of text messages.
As a user, I would like to be able to modify my travel plans.
As a user, I would like to be able to delete my travel plans.
As a user, I would like to be able to receive email updates.
As a user, I would like to be able to modify my account information.

As a user, I would like to be able to create an account.

As a user, I would like to be able to visit the website.

As a user, I would like to be able to log into my existing account.

As a developer, I would like to be able to store user information in a database.

As a developer, I would like to use TravelAPI to receive current information regarding restrictions in most countries.

2.2 Final Requirements

We will have an Angular-based web application that is connected to the TravelAPI as well as a database that will store user information and trip information. Our web application will have an API key to the API and will only exchange information about different countries and their travel restrictions without linking it to a user. On the database side, we will have different datasets on a firebase database. One will have a user ID and trip IDs. One will have the trip ID and trip information.

User Stories:

As a user, I would like to be able to add my travel plans.

As a user, I would like to be able to modify my travel plans.

As a user, I would like to be able to delete my travel plans.

As a user, I would like to be able to receive email updates.

As a user, I would like to be able to create an account.

As a user, I would like to be able to visit the website.

As a user, I would like to be able to log into my existing account.

As a developer, I would like to be able to store user information in a database.

As a developer, I would like to use TravelAPI to receive current information regarding restrictions in most countries.

3 Final Timeline vs Initial Timeline

The timelines stayed very similar, as we were able to stay mostly on track. We did leave a bit of wiggle room just in case we needed it, so that was very beneficial to our project.

3.1 Initial Timeline

Wk	Date	Class	What to Accomplish	Physically Due
5	09/20	Presentation 1	(Sprint 1)User stories, Project Plan, Find source of data. Set up **basic** Website. Search, Account, Login, and Home pages.	**Friday:** Project Plan
6	09/27	Group Meetings 1A		**Friday:** Individual Plan
7	10/04		(Sprint 2)Set up connection to a DB. Update Account page. Create travel plans page.	**Friday:** Technical Document
8	10/11			
9	10/18	Presentation 2	(Sprint 3) Set up text message and email connection.	**Friday:** Group Report 1
10	10/25			**Friday:** Group Report 2
11	11/01		(Sprint 4) Pick a host platform. Start UI update. Set up automatic sending of messages to users.	**Friday:** Group Report 3
12	11/08	Group Meetings 2B		**Friday:** Group Report 4
13	11/15		(Sprint 5) Finish UI update. Finish remaining tasks.	**Friday**: Individual Presentation
14	11/22			**Friday**: Individual Report
15	11/29	Final Presentations	Finish up final deliverables.	**Friday**: Final Deliverables
16	12/06	**Finals Week**		**Friday:** Department Questionnaire

3.2 Final Timeline

Wk	Date	Class	What to Accomplish	Physically Due
5	09/20	Presentation 1	(Sprint 1)User stories, Project Plan, Find source of data.	**Friday:** Project Plan
6	09/27	Group Meetings 1A	Set up **basic** Website. Search, Account, Login, and Home pages.	**Friday:** Individual Plan
7	10/04		(Sprint 2)Set up connection to a DB. Update Account page. Create travel plans page.	**Friday:** Technical Document
8	10/11			
9	10/18	Presentation 2	(Sprint 3) Set up email connection.	**Friday:** Group Report 1
10	10/25			**Friday:** Group Report 2
11	11/01		(Sprint 4)Working on email connection	**Friday:** Group Report 3
12	11/08	Group Meetings 2B		**Friday:** Group Report 4
13	11/15		(Sprint 5) Start UI update. Set up automatic sending of messages to users.	**Friday**: Individual Presentation
14	11/22			**Friday**: Individual Report
15	11/29	Final Presentations	Finish up final deliverables.	**Friday**: Final Deliverables
16	12/06	**Finals Week**		**Friday:** Department Questionnaire

4 Project Results vs Expectations

At the beginning of our project, we expected to make a travel web application that users would be able to use to keep track of covid restrictions based on different locations. This application would be able to send users emails and text updates. At the end of our project, we instead have a travel web application that users can use to keep track of covid restrictions for only two countries. This application is able to send users email updates.

5 Software Evaluation

After completion of this project, we went back and looked at our original Project Plan and we discussed a few things that we did not implement. For example, we didn't use merge

requests, which were meant to be used to share context throughout the project. Instead of doing this, we would code on our own and then discuss the changes we made during our weekly meetings. In addition to this, we would communicate over Discord if we had any questions regarding the code another person wrote. This helped us to maintain our knowledge of the growing codebase.

Cypress was mentioned in the Project Plan as our front-end testing framework, and we followed through with that. It was used to simulate clicks on the website to make sure our program was working as expected. However, we were unable to test the Google Login portion of the website because Cypress doesn't support iFrames, which were required with the Firebase Google Authenticator.

In addition, we added Firebase tests to verify our database was working properly. We followed the Principle of Least Privileged for the design of the database to make sure users only had access to the collections they were permitted to use.

6 Work to Be Done

Our project mostly met all of its requirements initially planned out, but a few of the things we didn't have time to complete were: the ability to receive text message updates and the ability to send dynamic emails. We decided to not send text messages because it took longer than expected to implement email updates. Additionally, the email updates do work, but they do not send dynamic information-- it's always the same email being sent to one of our personal accounts, so the user won't actually receive anything.

Another thing that we weren't expecting is purchasing the database for official use. Right now, we are just using the free version, which only allows us to use Germany and Spain as valid travel destinations. If we were to develop this program further, we would most likely end up purchasing full access to the API.