TRAVEL-IN

Capstone Design

April 30th, 2019

Fatima Ezzahra El Aidos

Supervised by Dr. Sedki Samadi
Capstone Report

Student Statement:
I, Fatima Ezzahra El Aidos, affirm that I have applied ethics to the design process and in the selection of the final proposed design. And that, the designer has held the safety of the public to be paramount and has addressed this in the presented design wherever may be applicable.

_____________________________________________________
Fatima Ezzahra El Aidos

Approved by the Supervisor(s)

________________________________
Dr. Sedki Samadi
ACKNOWLEDGEMENTS

First and foremost, I would like to express my deepest gratitude to my supervisor Dr. Sedki Samadi, for kindly accepting to supervise me, and for his support and dedicated involvement throughout the accomplishment of this capstone.

This accomplishment would not have been possible without my family's support. I, therefore, seize this opportunity to express my sincere and deepest gratitude to my parents and brother for their love, sacrifices, encouragements, moral and financial support throughout my journey at AUI.

To Jamal S., thank you for your patience, understanding and support over the past four years. Finally, to all my friends thank you for your help, advice and support. My stay at AUI would not have been memorable without you.
Contents

1. INTRODUCTION  1

2. PROJECT SCOPE  2

2.1. Project Overview  2

2.2. Market Information  2

2.2.1. Target Audience  2

2.2.2. Similar Existing Applications / Competitors  2

3. PROJECT METHODOLOGY  4

3.1. Feasibility Study  4

3.1.1. Technical Feasibility  4

3.1.2. Schedule Feasibility  4

3.1.3. Economic Feasibility  4

3.1.4. Legal Feasibility  4

3.2. Software Engineering Process  5

3.3. Project Management Plan  5

4. SOFTWARE REQUIREMENT SPECIFICATION  6

4.1. Functional Requirements  6

4.2. Non-Functional Requirements  11

4.2.1. Product Requirements  11

4.2.2. Organizational Requirements  11

4.2.3. External Requirements  12

4.3. Use Case Diagram  12

5. SOFTWARE DESIGN  14

5.1. Collections Diagram  14

5.2. Activity Diagram  15

5.3. Technology Enablers  15

5.3.1. Back End  16

5.3.2. Front End  17

5.4. System Architecture  17

6. SOFTWARE IMPLEMENTATION  18
6.1. Back End 18
6.2. Font End 20

7. SOFTWARE TESTING AND VALIDATION 20

8. STEEPLE ANALYSIS 21
8.1. Social Analysis 21
8.2. Technical Analysis 21
8.3. Environmental Analysis 21
8.4. Ethical Analysis 21
8.5. Political Analysis 21
8.6. Legal Analysis 21
8.7. Economic Analysis 22

9. CONCLUSION 22

10. FUTURE WORK 22

11. REFERENCES 25
Table of Figures

**FIGURE 1** INCREMENTAL MODEL 5
**FIGURE 2** TRAVEL-IN USE CASE DIAGRAM 13
**FIGURE 3** COLLECTIONS DIAGRAM FOR TRAVEL-IN 14
**FIGURE 4** ACTIVITY DIAGRAM FOR TRAVEL-IN 15
**FIGURE 5** TRAVEL-IN SYSTEM ARCHITECTURE 17
**FIGURE 6** TRAVEL-IN ROUTES 19
ABSTRACT

Doting the tourism sector, which is one of Morocco’s prospect pillars in its economy, with web and / or mobile applications to further promote it would be of great added value. Web and mobile applications are now frequently used for planning travels, for considerations such as authenticity or competitive prices. They provide users with information about lodging, security, and nearby addresses to check. These applications, however, tend to omit un-promoted, hidden regions that carry beautiful spots to discover. Taking that into consideration, the main aim of this capstone project is to build a web application which enables travelers to share information and feedback about unique and less known places to visit around Morocco. The name of this application is a concatenation of the words "travel" and "information". The present document provides an overview of the different phases I went through, in order to build a working prototype for this web application.
RÉSUMÉ

Doter le secteur touristique d’applications web et / ou mobiles afin de le promouvoir serait d’une grande valeur ajoutée à l’économie marocaine. De nos jours, les applications Web et mobiles sont fréquemment utilisées pour la planification des voyages. Ils fournissent aux utilisateurs des informations sur l’hébergement, la sécurité et les activités à proximité. Cependant, ces applications ont tendance à omettre les endroits secrets les plus envoûtants à visiter. Tenant compte de cela, le principal objectif de ce projet est de développer une application Web qui permettrait aux voyageurs de partager des informations sur des lieux uniques à visiter au Maroc. Le nom de cette application est une concaténation des mots "travel" et "information". Le présent document fournit une vue d’ensemble des différentes phases, qui ont permis la concrétisation cette idée.
1. **Introduction**

Morocco is a mesmerizing destination with unique, hidden, unconventional places that are worth visiting. Whether they are Moroccan citizens or foreigners discovering the country for the first time, all travelers that you might stumble upon across the country tend to live a unique experience during their journey rather than visiting the famous landmarks in each city. However, to find such unique places and/or itineraries, most if not all people go through a very complex process of thinking and planning since information about such places is usually difficult to find.

The idea behind *Travel-in* is to build a platform that would contain a selection of truly worth crossing off your bucket list destinations all over Morocco. Moreover, this selection of destinations would be provided through *Travel-in* in the form of suggestions shared by travelers, with other travelers, from all over the world, that are seeking inspiration and/or information regarding unconventional touristic gems in Morocco.

The platform would not only ease the process of finding the right place, among a variety of available suggestions, to spend their next weekend or holidays but also share their feedback on previous experiences but also promote tourism and indirectly contribute to the economic revival of some villages, and deteriorating areas.
2. Project Scope

2.1. Project Overview

*Travel-in* is a web application that will offer to travelers the opportunity to share information about unique places to visit in Morocco and share their journeys and experiences through comments. With regard to the schedule, this web application is to be delivered by the end of Spring 2019 as a fulfillment of the capstone design requirements.

2.2. Market Information

Conducting a small market research at the early stage of my capstone project was an opportunity for me to know more about my competitors and narrow down my target audience.

2.2.1. Target Audience

The *Travel-in* web application will mainly target foreigners who would like to visit the hidden gems of Morocco. Yet, since the number of Moroccan citizens who are interested on visiting unique spots around the country is continuously growing, and due to the lack of applications and/or websites that offer such information, Moroccan travelers are also among my target audience. The platform will be designed to grant access to its users at any time and from anywhere, as long as they have an internet access and a laptop or a phone.

2.2.2. Similar Existing Applications / Competitors

*General Description*

*Google Places:* As the search metrics website defines it, Google Places is “Google’s listing for local business search results” [1]. This web listing, that is quite similar to yellow pages, provides users with relevant locations of businesses who have registered with Google Places. In addition to the users’ ability to leave reviews, information that is displayed to the user next to the map that shows the location’s address, pictures of the location, opening and closing times, website address and phone number of the concerned business [1].

*Visitmorocco.com:* This website is the property of the Moroccan Office of Tourism(MOT). It has several sections that provide general information regarding
traveling around the country, ranging from history and geography to cultural immersion experiences and excursions. Unfortunately, in terms of content, this website does not provide any information about unique places to visit while in Morocco [2].

Moroccotourismoffice.com: This website is owned by Mr. Laurent Guinard. The platform also provides its visitors with a fair amount of information concerning all the regions of Morocco. Furthermore, and unlike the MOT's website, Mr. Guinard's platform comes with a section named "Unique Destinations" which includes similar content to the one I am willing to provide to my audience. However, the section still needs further development as so far it only contains 9 destinations [3].

Performance Tests

The table below summarizes the results of performance tests that were conducted through the “Dareboost” tool. The “first byte” column of the table refers to the amount of time elapsed between the first HTTP request sent from the browser and the reception of the first byte of the web page by the browser. The “start render” column, defines the delay between the first request from the browser and the first visual elements being displayed on the screen. Last but not least, the third column refers to the time it takes for the web page to be fully loaded with all resources being fetched, parsed and executed [4].

<table>
<thead>
<tr>
<th></th>
<th>First Byte</th>
<th>Start Render</th>
<th>Fully Loaded</th>
</tr>
</thead>
<tbody>
<tr>
<td>Google Places</td>
<td>0.315s</td>
<td>1.200s</td>
<td>2.676s</td>
</tr>
<tr>
<td>VisitMorocco</td>
<td>0.497s</td>
<td>1.600s</td>
<td>5.737s</td>
</tr>
<tr>
<td>MoroccoTourismOffice</td>
<td>0.623s</td>
<td>1.800s</td>
<td>11.802s</td>
</tr>
</tbody>
</table>

According to “Dareboost”, It is recommended that a server response takes less than 200 milliseconds, for the first visual elements to appear in less than a second, and for a page to be fully loaded within 4 seconds. In comparison with the recommendation, we could clearly see that Google places is better in terms of performance compared to the two other websites [4]. As the “MoroccoTourismOffice” seems to be the slowest one in terms of time it takes to be fully loaded with 11.802s in contrast to Google places which only requires 2.676s.
3. Project Methodology

3.1. Feasibility Study

Carrying out a feasibility study is an essential step in any software development process, as its aim is to find out whether an idea can be implemented based on several factors which include but are not limited to: technical, economical, schedule and legal factors.

3.1.1. Technical Feasibility

As I am willing to use some tools and technologies that I am not very familiar with, and which include but are not limited to: Node.js, Express.js, MongoDB, HTML, CSS, and REST, the product implementation will be challenging. Yet, the substantial amount of documentation and tutorials available online on how to use such technologies, added to diverse software developments skills I acquired through the Computer Science major courses, make this web application technically achievable.

3.1.2. Schedule Feasibility

Two weeks will be spent to finalize the requirements specification, designing the platform’s architecture and setting up the environment, in order to start building Travel-in. The implementation time frame of this product, is of about 7 weeks and the last week before the capstone defense will be spent on testing, which I will perform. This timeframe surely does not allow for a sophisticated version of the platform to be released. Nevertheless, it will allow me to build a prototype that matches the requirements that will be specified later on this document.

3.1.3. Economic Feasibility

The platform would not require any costs at its early beginning since I will take care of all the design and implementation required to build the software, and also because all the software resources that will be used are available for free.

3.1.4. Legal Feasibility

As for the time being, Travel-in will not use any e-commerce services. Furthermore, as the Moroccan Law states it, Travel-in users’ personal information will be treated under the terms of “La Loi n°09-08 relative à la protection des personnes physiques à l’égard
du traitement des données à caractère personnel”, and will not be sold to any third party.

3.2. Software Engineering Process

In every software engineering process model, there are four mandatory activities to go through: Software specification, software design and implementation, software testing, and software evolution [5]. Due to the time constraints, and the fact that I wanted to put into practice agile methodology while building this application, it seemed that the incremental model was the best fit for this software product as it allows us to first build a simple system on which only few features were implemented and delivered to the customer. Thereafter, successive versions are implemented based on the customer’s feedback and delivered again for further feedback until the desired application is realized.

![Figure 1 Incremental Model](image)

3.3. Project Management Plan

<table>
<thead>
<tr>
<th>Week</th>
<th>Agenda</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>• Project Selection</td>
</tr>
<tr>
<td>2</td>
<td>• Initial Specifications</td>
</tr>
<tr>
<td>3</td>
<td>• Feasibility Study and Analysis</td>
</tr>
<tr>
<td>4-7</td>
<td>• Requirement Specification &amp; Design</td>
</tr>
<tr>
<td></td>
<td>• Implementation begins</td>
</tr>
<tr>
<td></td>
<td>• Interim Report Submission</td>
</tr>
</tbody>
</table>
4. Software Requirement Specification

4.1. Functional Requirements

Defining functional requirements is concerned with clarifying features of a system and its components. This step can be regarded as the backbone of the system, since it describes services the system should provide [6].

Account Handling

This section's functional requirements will how the system and the system will interact to handle accounts creation requests, login requests.

Visitors:

- **User:** No privileges shall be granted to visitors. They are only allowed to view public information on the site.

- **System:** If a user is using the system without being registered and signed in, the system shall classify him as a visitor. As long as the system considers a user to be a visitor, he shall only be allowed to access data that has been categorized as public. Moreover, a link to the sign-up/sign-in pages should be displayed at the top of every page.

Creation of Accounts:

- **User:** The system should allow a new user to create an account in the platform by providing his with a form to fill. The user shall input a username and an email which have not already been assigned, and a password of his choice, alongside with other complementary information which include: the first name, the last name, a profile picture and a small introduction.

- **System:** When the user submits the registration form, the username and the email entered shall be checked against all existing usernames and emails in the database. If a match is found, the system shall display an error and prompt the
new user to re-fill the form. If the no match was found, the system shall redirect the user to the home page and display a successful registration message.

**Sign-In:**

- **User:** The system shall prompt the user to input his username and password to sign-in.

- **System:** The system shall verify that the credentials entered are correct (existing username, correct password). The system shall search the database to determine the correctness of the user's credentials. If a match is found for both the username and the password, the user shall be granted access to his account and accordingly shall be able to use all the platform's functionalities. Should a mismatch occur, the system shall display an error message informing the user about the incorrectness of his credentials.

**Profile Information:**

- **User:** A signed in user shall be able to view and update when necessary his profile information, except for the username. A signed in user shall also be allowed to view other users' profile information limited to the following: the first name, the last name, the email, the introduction and all suggestions that the concerned user posted in the platform.

- **System:** The system shall grant a user the right to edit his profile information. The update button shall be displayed as a button in the profile information page.

**Sign-Out:**

- **User:** To safely quit using the platform, the user shall be provided with a log out link.

- **System:** If a user chooses to sign out, the system shall close/destroy the user's session and redirect him to the landing page.

**Content Handling**

The functional requirements below, define how the system will provide access to suggestions posted: creation, viewing, searching, deleting and updating.
Viewing:
- **User:** Whether they are signed-in or not the system shall allow users to view suggestions posted on the platform.
- **System:** The system shall display all the suggestions found in the database regardless of the user's connection status.

Searching:
- **User:** Whether they are signed-in or not the system shall allow users to look for a specific suggestion using a fuzzy search.
- **System:** The system should search in the database for posts, whose title matches the keywords specified the user in the search bar.

Creating:
- **User:** Each signed-in user shall be granted the right to add, into the platform, new suggestions of places to visit. These suggestions shall be created by clicking the "Suggest a new place" button, which will be displayed on the main page. This button should redirect the user to an interface with a form into with he will be prompted to input the new place's information.
- **System:** If the user clicks on the "Suggest a new place" button, the system shall first check if this user is signed-in. If not signed-in, the user should be prompted to first sign-in to be granted the content creation privilege. Otherwise, the system displays the page in which the user will input the data to be inserted to the database. The user shall be required to specify the suggestion's title, a description, and finally upload a picture of the place. If all fields are correctly filled, the system should redirect the user to the newly created posted. Else, the system shall display an error message to the user.

Deleting:
- **User:** If a user wishes for his suggestions to no longer be on the platform, the system shall grant the owner of the post the ability to delete the content using a delete button which must be displayed under the suggestion's description.
• **System:** The system shall first check if the user is signed-in, then if he owns the post. If he does, the delete button should be viewable and the delete action shall destroy any content associated to the post from the database.

**Updating:**

• **User:** If the owner of a suggestion wishes to change any of his suggestions' content on the application, the system shall grant him the right to update the post for any reason. The update option shall be displayed as a button associated with the post.

• **System:** The system shall first check if the user is signed-in, then if he owns the post. If it is the case, this user will then be granted all privileges to the post, including updating it. If the owner of chooses to update a suggestion, the system shall update the post from the database alongside with the picture associated to the suggestion. The update button is to be displayed as part of the options which will only be viewable for the owner of the suggestion.

**Liking/Un-liking:**

• **User:** Signed-in users shall be able to like or unlike a suggestion using the like button that will be displayed under each suggestion. All users (visitors included) shall be able to view the total amount of likes that a specific suggestion has reached.

• **System:** The system shall first verify if the user is signed in, if not it should prompt him to sign in first to be able to record the like/ un-like into the database.

**Comments Handling**

This section is concerned with the comments functional requirements. Below I am defining how the system will handle comments written by the application's users.

**Viewing:**

• **User:** Whether they are signed-in or not the system shall allow users to view comments that are associated to suggestions in the platform.
• **System:** The system shall display all comments written by users. Comments shall be retrieved from the database and associated to the concerned suggestions.

**Creating:**

• **User:** Each signed-in user shall be granted the right to comment a suggestion. These comments shall be created by clicking the "Add comment" button, which will be displayed under the suggestion's description. Once the button clicked, the system shall unfold a comment creation box in which the user can write his feedback and submit it.

• **System:** If the user clicks on the "Add comment" button, the system shall first check if this user is signed-in. If not signed-in, the user should be prompted to first sign-in to be granted the comment creation privilege. Otherwise, the system shows the comment section, in the same page. The user will type the comment to be inserted and associated to the suggestion within the database.

**Deleting:**

• **User:** If a user wishes for his comment to no longer be associated to a suggestion on the platform, the system shall grant the owner of the comment the ability to delete it using a delete button which must be displayed near the user's comment.

• **System:** The system shall first check if the user is signed-in, then if he owns the comment. If he does, the delete button should be viewable and the delete action shall remove the comment from the database.

**Updating:**

• **User:** If the owner of a comment wishes to change its content, the system shall grant him the right to update it for any reason. The update option shall be displayed as a button associated with the comment.

• **System:** The system shall first check if the user is signed-in, then if he owns the comment. If it is the case, this user will then be granted all privileges to the comment, which include updating it. If the owner of chooses to update his comment, the system shall remove the old comment and store the new one in
the database. The update button is to be displayed as part of the options which will only be viewable for the owner of the comment.

4.2. Non-Functional Requirements

As opposed to functional requirements, non-functional requirements define the constraints on the services provided by the system. Non-functional requirements can be divided into three main categories: Product requirements, Organizational requirements, and External requirements [6].

4.2.1. Product Requirements

Usability Requirements

The graphical interface of the web application should be user-friendly. The web application should make use of intuitive command name to provide an effective navigation to its users.

Efficiency Requirements

Performance Requirements: users often associate performance to the system’s response time. Thus, to meet my users’ performance expectations, the think time between a user’s request and system’s response should not exceed 100 milliseconds.

Space Requirements: the web application should be of a lightweight. It should be able to smoothly perform its functionalities even if the device only uses a 512 MB RAM.

Reliability Requirements

The web application should run for a minimum of six months without experiencing any type of bugs and therefore without the need of regular maintenance.

Portability Requirements

The web application should be able to appropriately work under “Chrome”, “Firefox” and “Opera” browsers.

4.2.2. Organizational Requirements

Delivery Requirements

The Travel-in web application will be delivered by the 22nd of April 2019.
**Implementation Requirements**

The technology enablers used should be NodeJS, ExpressJS, JavaScript, HTML, CSS, and Bootstrap. The database should be written using MongoDB.

**4.2.3. External Requirements**

**Legislative Requirements**

The user data shall be used for any purpose without prior consent of its owners.

**Safety Requirements**

Safety Requirements: the web application should not contain incorrect and/or harmful information.

Security Requirements: access to the platform shall be protected. Only accredited users should be able to manage the system.

**4.3. Use Case Diagram**

The following diagram illustrates the functional requirements that were detailed in the previous section.
Figure 2 Travel-in Use case diagram
5. Software Design

5.1. Collections Diagram

![Collections Diagram for Travel-in](image)

**Figure 3** Collections diagram for Travel-in

Relationships between the different entities, in the diagram, is described in the following statements:

- Each user owns an account.
- Each user has zero to many suggestions posted.
- Each user has zero to many comments posted.
- Each user has zero or one like on a suggestion.
- Each suggestion has zero or many likes.
- Each suggestion has zero or many comments.
5.2. Activity Diagram

![Activity Diagram for Travel-in](image)

**Figure 4** Activity diagram for Travel-in

5.3. Technology Enablers
The choice of JS technologies to develop this web application was first influenced by the growth and popularity as well as the availability, on several platforms, of documentation and solutions to frequently recurring bugs for such enablers. Furthermore, as web development technologies that I have worked with, over the past years, started to
deprecate, it was for me a necessity and an opportunity, at the same time, to challenge myself to learn and put into practice these new technologies.

The sections below give a brief description of the different technologies used for implementation.

5.3.1. Back End

**NodeJS**

I used NodeJS because it provides a server-side environment which allows, me, as a developer to generate web pages using JavaScript instead of using server language to handle the user's requests. Moreover, as opposed to traditional web serving techniques in which each connection creates a new thread, which eventually uses the available space RAM, NodeJS uses non-blocking, event-driven I/O calls which allow it to support multiple concurrent connections, and thus to remain lightweight[7].

**ExpressJS**

To layer in functions needed to build a web application, we need the Express framework, since NodeJS itself was not intended to implement web applications[8]. I used ExpressJS for sever side routing as it provides, through EJS templates, the dynamic rendering for HTML pages in addition to the ability to create middlewares in order to respond to HTTP requests. In other words, it eases the fast development of Node based applications.

**MongoDB**

For my database, I used MongDB which uses JSON documents to store data. Unlike traditional relational databases in which data is stored into columns and rows, MongoDB data objects are stored within a collection as separate documents. This NoSQL database is extremely simple to install and implement. The high flexibility of MongoDB offers the ability to enforce any type of data. Similarly, in terms of performance, MongoDB is better when the number of queries is higher[9].
To handle the database back-end operation with MongoDB, I used MongooseJS an Object Document Mapper which provides schema based solutions to model data.

5.3.2. Front End

Bootstrap

To build and style the visual elements of the application, I relied on Bootstrap which is a popular front-end CSS framework with pre-written style sheets to ease and fasten the process of building web pages [10]. I opted for this framework as it is easy to use if you have prior knowledge in both HTML and CSS in addition to the fact that it offers several features such as responsiveness, consistency and customization.

5.4. System Architecture

The figure below, illustrates the Travel-in web application's system architecture.

![System Architecture Diagram]

**Figure 5** Travel-in System Architecture
When a client sends a request, it is processed by the front end technology enablers first (Bootstrap in my case). The request is then translated using NodeJS and moved to Express JS which makes request to the Database. Then, MongoDB retrieves the requested data and return a response to ExpressJS which, in its turn, returns the received response to NodeJS who transmits it as well to the front-end framework which displays the results to the user.

6. Software Implementation

To implement my application, I mainly relied on Cloud9 IDE and its command line to run the code and preview the results. Due to the time constraints and the lack of experience I had with technologies I used, I relied on a code-and-fix development model.

Cloud9 IDE eased the implementation process, to some extent. Since this latter is an IDE that is hosted on cloud and provides its users with workspaces on which many recent technology enablers are already pre-installed and only few commands need to be executed in order to start using them.

While the front end of my application was developed using Node JS, Express JS and MongoDB, the front end of Travel-in web application was implementing using HTML, CSS, Bootstrap as well as JavaScript.

6.1. Back End

First step I went through while implementing the back-end was to learn how to create routes using Express, which is a JavaScript framework used to quickly and easily create routes. Next step after learning how to create routes, was to know more about RESTful routes, understand how they work and put them into practice on my web application.

As opposed to a normal route which exclusively relies on the URL to designate which page is requested, a RESTful route maps HTTP verbs (get, post, put, delete, patch) with controller CRUD actions (create, read, update, delete) to determine which response needs to be sent back to the user [11].

To further clarify the link between routes within my application, the schema below illustrates their structure:
The first interface the user will interact with after the landing page, is the home page. When the /home is loaded, the only accessible paths are the following: /Login, /Register/ and /Landmarks. These paths allow visitors to view public information available on the platform as well as the choice to either sign in, if they already own an account, or register and become a member of the Travel-in community.

Once the user is logged in, he is granted privileges to also request the following: /Profile, /Comments/ and /Logout. /Landmarks allows users to view a suggestion, like it, and comment it. Furthermore, if the user owns the said suggestion, he is also allowed to update or delete the post. Similarly, if a user owns any of the comments that were written on the suggestion, he is granted the rights to make changes or delete it.

With regard to the Incremental Development Model mentioned earlier in this report, the following table details work achieved while following this model.
The main challenge I encountered while working on my product was to be able to come up with a working prototype of the application, within the short time period we were given added to the limited background I have in web development. The implementation phase was particularly challenging as I had to handle several issues related to multiple function callbacks and promise handling.

6.2. Font End
The Front End implementation mainly consisted on building the visual elements of the web application using HTML and CSS during the early stages of the software product. Later on, I added Bootstrap4 components to enhance the user interface.

7. Software Testing and Validation
Testing and validation is a crucial step in the software engineering process. The travel-in application in my case, fulfills the requirements that I initially stipulated. Unit tests were performed every now and then throughout the implementation of the web application due to the use of the incremental model which required me to implement an increment and release it to get my supervisor's feedback before working on the next increment. An integration testing was performed too to ensure that components (increments) are correctly interacting between each others. A system testing was also conducted to test the system as a whole and ensure that
both types of requirements are met. Finally, an acceptance testing was conducted in front of Dr. Samadi who determined that all requirements that we agreed on were met.

8. STEEPLE Analysis

8.1. Social Analysis

As access to information became easier, travelers today no longer rely on guidebooks to look for and plan their next journey. Relevant tips and pictures depicting adventures are commonly shared through social media. Yet, in Morocco, as of the time being, applications and websites that promote tourism usually focus on imperial cities of Morocco, rather than hidden and gorgeous spots that very few people know about. Through the Travel-in platform I intend to promote tourism in Morocco by encouraging travelers to share information and comments regarding hidden touristic gems in Morocco.

8.2. Technical Analysis

Travel-in users will have the opportunity to use an application with a pleasant navigation as implemented using the latest technology enablers.

8.3. Environmental Analysis

Travel-in web application does not have a direct impact on the environment. Nevertheless, encouraging people to plan their next hikes to some rural areas, and/or mountains may indirectly contribute to protecting the environment.

8.4. Ethical Analysis

The application will not violate the privacy of the users as its main goal is to help them find hidden spots to visit in Morocco.

8.5. Political Analysis

Travel-in web application does not intend to be affiliated to any political party.

8.6. Legal Analysis

To ensure that Travel-in does not violate any copyright law, only free resources are being used to implement my web application.
8.7. Economic Analysis

The web application is initially designed to inspire travelers and share useful information through the platform. I do not intend to make economic gain from that. However, promoting spots 8 that are not well-known among travelers may help the communities living around to economically evolve over time.

9. Conclusion

This capstone project was a golden opportunity to me. I was able throughout the semester to not only learn a fair amount of new technologies but also put them into practice. Indeed, I became more familiar with JavaScript, HTML, CSS, and Bootstrap. Furthermore, through the implementation of Travel-in, I was able to learn more about non-relational databases through the use of MongoDB. I was also able to learn, understand and put into practice the REST architectural philosophy, NodeJS, and ExpressJS in addition to learning about a variety of packages that I used to ease the implementation process.

The initial idea for my capstone project was to implement a web or a mobile application which would target users of the Facebook group “J’ai testé…Je vous le recommande”. After discussing with my father, he suggested to be more specific as this group encompasses information about many aspects of our daily life such as: recommendations regarding products, services, travel deals and experiences. Upon his suggestion, I have decided to design and implement web platform on which travelers would share information and feedback about unique places to visit around Morocco.

Last but not least, the implementation of the web application would not have been possible without the knowledge and skills that I have acquired throughout my academic journey at AUI.

10. Future Work

As part of my future work I would like to not only implement additional features, but also move from a Multi-Page Application (MPA) to a Single Page Application (SPA). Rather than loading new pages each time a request is made, an SPA consists of only one page which is dynamically updated as the user is interacting with the application. A well-known example of such applications could be: Google maps [10].
The main advantage of using an SPA rather than an MPA, is that it keeps the user in one page on which content is presented in a simple and workable fashion. In other words, it provides to the user an outstanding UX experience as through the use of advanced JavaScript frameworks such as: AngularJS, Ember.js, Meteor.js, Knockout.js [12].

I am very concerned about my users’ internal experience while interacting with my web application. To this aim, after digging on several JavaScript frameworks, I was triggered by the philosophy of building single page web applications using ReactJS and would love to invest my time learning it and putting it into practice on my application.

As for the features of my web application, I am planning to add the following:

**Notifications**

The notification system will give users the following abilities:

- Be notified when a new suggestion is added to the platform.
- Be notified when your suggestion is liked by another user.
- Be notified when other users like/or comment your own suggestions.
- Be notified when another user is following you.
- Be notified about new suggestions made by users that you are following.

**Suggestions based on user preferences and/or localization**

This feature would show the most relevant suggestions to a user based on the preferences of a user and/or his localization.

**Private messaging**

This feature would allow people to communicate between each other through the platform.

**Ability to upload more than one picture and/or videos:**

As the web application currently handles only one picture per suggestion and no videos at all, this feature will allow users to share multiple pictures as well as videos.

**Follow/Un-follow a user**

This feature would give the followers the ability to see suggestions of the users they are following at the top of their news feed.

**Admin panel**
Implement an admin panel which will allow the administrator of the web application to manage users and suggestions. Through this panel, an admin will have the ability to ban users who continuously publish irrelevant content, delete suggestions which do not comply with the rules of the platform, and delete irrelevant comments.
11. References


