Perfect Time - Plan your trip
Software Requirements Specification

Version 1.0

Revision History

Date	Version	Description	Author
<17/10/2018>	<1.0>	First Draft	Jan Rickel

Perfect Time - Plan your trip	Version: <1.0>
Software Requirements Specification	Date: [Veröffentlichungsdatum]
<document identifier>	

Table of Contents

Software Requirements Specification

1. Introduction

For our course Software Engineering we are creating a web application. Our application is called Perfect Time – plan your trip.

Our vision is to create an Application that allows you to plan your trip by setting different locations or stages. You will also be able to insert generic data for the whole vacation. After your holiday you can add a blog in the App using the already available data from the planning of your trip. Afterwards you can publish your trip and share it with your friends. They can use your plan as an outline for their own trip by simply importing your published data.

This document will specify the software requirements we will have to fulfill, if we want to transform our vision into a satisfying software product. This Software Requirements Specification (SRS) adheres to the standards defined by IEEE830-1998.

1.1 Purpose

This **SRS** fully describes the external behavior of the Perfect Time Application. It also describes nonfunctional requirements, design constraints, and other factors necessary to provide a complete and comprehensive description of the requirements for the software.

This SRS is a requirement for the course Software Engineering at the Corporate State University Baden-Württemberg in Karlsruhe.

1.2 Scope

This SRS will cover the complete Perfect Time Application with all defined features and subsystems. The defined subsystems are:

- Basic Web Application

  Goal: Build a running website that provides our basic application.

- Location Pages

  Goal: Allow subpages for individual locations / events. One page should contain all the relevant data (address, date, map, etc.) on one location.

- Vacation Info

  Goal: Allow general information for the whole vacation and statistics. I.e. your flight schedule, your budget, etc.

- Travel Blog

  Goal: Allow redesign of location subpages into a travel blog. The individual location pages will look like a travel blog (pictures, videos, text, …).

- Share

  Goal: Allow other users to use the travel plan as a layout for their own travel. The system will create the subpages and the vacation info based on the other project.

- Multiple Users (idea based on blog comment by Christian)

  Goal: Provide different users and change tracking. This feature should make planning with multiple persons easier.

Those subsystems will reflect into the functionality and usability requirements which are defined in section 3.

Perfect Time - Plan your trip	Version: <1.0>
Software Requirements Specification	Date: [Veröffentlichungsdatum]
<document identifier>	

1.3 Definitions, Acronyms, and Abbreviations

MVC = Model View Controller

SRS = Software Requirements Specification

Travel = one specific trip / vacation. For each travel the user will create an individual plan.

Trip = Travel

Location = one location visited during the travel. A location might be a city or a region. A travel contains multiple locations.

Activity = one specific activity planned for a location. An activity might be a place or an event. Each activity must be assigned to a location.

1.4 References

- Perfect Time Blog: https://perfecttime608150251.wordpress.com/

- Perfect Time YouTrack: https://perfecttime.myjetbrains.com/youtrack/agiles

- Perfect Time Git Repository: https://github.com/Tallround3r/PerfectTime

- Perfect Time Application Website: https://perfecttime-planyourtrip.firebaseapp.com/

1.5 Overview

An overall description of the software and its requirements will be given in Section 2. Section 3 will define the specific requirements. Section 4 will include supporting information (i.e. table of content, etc.).

2. Overall Description

This section describes the general factors that affect Perfect Time and its requirements.

• product perspective

Perfect Time was created to fill a market gap. The Perfect Time Team could not identify a similar product on the market. However, it can be assumed that there is a market demand, since many people go on vacation and therefore must plan their trip. A good market perspective can be predicted.

• product functions

The product has different functions. They are displayed in the use case diagram.
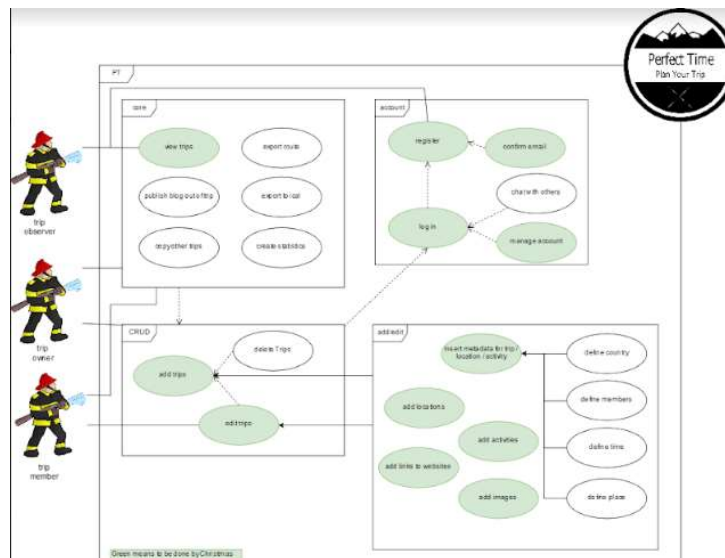
- user characteristics

The users of Perfect Time get divided into three categories. The first category are trip owners. They use the applications to plan one or multiple trips. They decide what the (privacy) settings are and who becomes a trip member. The second category are trip members. They got invited by the trip owner and can contribute to plan the trip. They can add and edit locations. The third group are observers. They can view public travels and use their information as a base layout for creating their own trip and becoming an owner.

- constraints

There are constraints to the number of accesses of our application, since we use the free Google Firebase subscription to host our application. Thereby the scalability of the application is limited. However, by using a higher (paid) subscription plan, the available limits can be easily increased.

The application is also designed as a web application. It can only be accessed in a modern web browser and with an active Internet connection.

- assumptions and dependencies

The application is designed to have only a few external dependencies. It is dependent on the technology it uses (React, Redux) and the service it is deployed on (Google Firebase). Apart from that, there are no further dependencies.

This application is written during two terms at the DHBW. It is uncertain if the time will be enough to create the full application, as it is described in this document. It might be possible that only some of the features will be implemented on time. The team will still try to implement the whole application. This is the assumption this application is based on.

- requirements subsets

The requirements will be divided into the following categories: core (core functions), account (user accounts) and CRUD (Working on a trip).

3. Specific Requirements

This section describes the specific requirements of the application.

3.1 Functionality

[This section describes the functional requirements of the system for those requirements that are expressed

Perfect Time - Plan your trip	Version: <1.0>
Software Requirements Specification	Date: [Veröffentlichungsdatum]
<document identifier>	

in the natural language style. For many applications, this may constitute the bulk of the **SRS** package and thought should be given to the organization of this section. This section is typically organized by feature, but alternative organization methods may also be appropriate; for example, organization by user or organization by subsystem. Functional requirements may include feature sets, capabilities, and security.

Where application development tools, such as requirements tools, modeling tools, and the like, are employed to capture the functionality, this section of the document would refer to the availability of that data, indicating the location and name of the tool used to capture the data.]

3.1.1 *<Core>*

The core functionality provides the basic web application that can be accessed by the user through a web browser. It is an html page that contains several components created with React. It provides a navigation to and an overview over the other functions.

It also includes the possibility to view trips, which you are a member or the owner of. Any user can also see public trips. It provides a search function to identify travels.

The core functions include the possibility to export data from existing (public) trips. The data can be exported to a local drive as a backup. The data can be exported as calendar information (ical), so that the user can see all the locations in his/her calendar. Data from an existing (public) trip can be used to create another trip, which will use the layout of the old trip.

The basic functions will also provide a series of statistics. The user can see the budget, the expenses calculated, the km driven (per day) and the duration of the trip.

The last core function is to turn a trip into a travel blog. Using the planning layout, the user can insert pictures, text and videos into each location / activity page.

3.1.2 *<Account>*

The application provides a user management. Any user can become a registered user by filling out the registry form and responding to the confirmation email. A registered user can log into the application. The user can manage his/her account (change password, change email, change user name). He or she can use the CRUD functions.

The registered users can also chat with each other within a travel community (all contributors to a trip). The owner of a trip can invite other registered users to become members of the trip.

3.1.3 *<CRUD>*

Any registered users can create a trip / travel for him-/herself. The owner of a trip can also delete or edit the trip. He or she can make a trip public or private and enable the blog function.

Any trip members can edit the information contained in a trip. They can add meta data (i.e. budget, time frame, country, etc.). They can add locations to the trip and edit the information of the location (place, date, costs, etc.). They can also add activities to the locations.

3.2 Usability

[This section includes all those requirements that affect usability. For example,

- specify the required training time for a normal user and a power user to become productive at operations

- specify measurable task times for typical tasks or base the new system's usability requirements on other systems that the users know and like

- specify requirement to conform to common usability standards, such as IBM's CUA standards Microsoft's GUI standards]

Perfect Time - Plan your trip	Version: <1.0>
Software Requirements Specification	Date: [Veröffentlichungsdatum]
<document identifier>	

3.2.1 <User training>

The application is designed to allow an intuitive usage. Any user should be able to use every aspect of the application within three hours. The application provides enough documentation. A video tutorial will be provided by the Perfect Time team, as soon as the application is complete.

3.2.2 <Common features to other applications>

The application is designed to mimic common features of other applications. The design used (see section 4) is closely related to designs used by Google and Facebook.

3.3 Reliability

3.3.1 <Availability >

Since the application does not require a long deployment time, there should be only short maintenance times. The application will be available 99.00% of the time.

3.3.2 <Mean Time to Repair >

Any serious bug will be fixed within a week. The application will not require any updates by the user.

3.3.3 <Accuracy >

The application is designed to fulfill all the features described in this document to the full extend.

3.3.4 <Maximum Bugs >

The application should not have more than one bug per 500 lines of code at any time. The finished application should contain way less bugs.

3.4 Performance

3.4.1 <Response time>

The application should not have a long response time. The response time should only depend on the speed of the Internet connection. It should not take longer than two seconds for any transaction.

3.4.2 <Capacity>

The application is limited by the free Firebase subscription to 1 GB of storage space and 10 GB download volume per month.

3.4.3 <Degradation mode>

If the application should not be operational for any reason the last running version should be deployed as soon as the matter has been recognized.

3.5 Supportability

3.5.1 <Testing>

The software uses Unit Tests to ensure it is working correctly. Perfect Time uses JEST to write the tests.

3.5.2 <Coding>

The software uses the clean code standards to ensure maintainability.

3.6 Design Constraints

3.6.1 <React>

The application uses the React Framework. Future updates and API changes must be implemented into the Perfect Time Application.

3.6.2 <Redux>

The application uses Redux to save the component states. Changes in Redux or the React compatibility will have to be adapted to the application.

3.7 On-line User Documentation and Help System Requirements

The documentation of our software can be found in our Git-Repository or on our blog. The links can be found in the reference section of this document.

3.8 Purchased Components

The backend of the Web application is running on Google Firebase (free subscription plan).

3.9 Interfaces

[This section defines the interfaces that must be supported by the application. It should contain adequate specificity, protocols, ports and logical addresses, and the like, so that the software can be developed and verified against the interface requirements.]

3.9.1 User Interfaces

The application is designed as a web application. It can be accessed with any modern web browser. The design components that are used can be found in section 4.

3.9.2 Hardware Interfaces

This software id designed to run on any webserver. It can be accessed by any PC with a modern browser.

3.9.3 Software Interfaces

This application is developed to work with Google Firebase. It uses the interfaces of Firebase.

3.9.4 Communications Interfaces

The communication of the application depends on the server / Firebase settings. It can be accessed via https. The port and access settings depend on the server properties.

3.10 Licensing Requirements

N/A

3.11 Legal, Copyright, and Other Notices

N/A

3.12 Applicable Standards

N/A

4. Supporting Information

The following design will be used for the user interface.

Login Page

Register Page

Home Page

Location/Activity Page

Location Detail Page

Activity Detail Page