Requirements and Design

PR Project in Media Informatics

WS 2017/18

Document Control

Contributors

|  |  |  |  |
| --- | --- | --- | --- |
| Person | Role | Company | Contribution |
| Bozhidar Ivanov | Use Case Lead | UNIVIE | Template draft |

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Issue | Author | Date | Description |
| V0.1 | Bozhidar Ivanov | 29-Oct-2017 | 1st version written |
| V0.2 | Bozhidar Ivanov | 02-Nov-2017 | 2nd version written |
| V0.3 | Bozhidar Ivanov | 14-Nov-2017 | 3rd version written |
| V0.4 | Bozhidar Ivanov | 20-Nov-2017 | 4th version written |

Table of Contents

[1 Application Overview and Objectives 3](#_Toc498985523)

[1.1 Application Overview 3](#_Toc498985524)

[1.2 Objectives 3](#_Toc498985525)

[2 Use Cases 4](#_Toc498985526)

[2.1 Roles and Actors 4](#_Toc498985527)

[2.2 Use Case Diagrams 4](#_Toc498985528)

[2.3 Textual Use Case Descriptions 6](#_Toc498985529)

[3 Architecture 9](#_Toc498985530)

[3.1 Overview 9](#_Toc498985531)

[3.2 Architecture for Start page 9](#_Toc498985532)

[3.3 Architecture for Home 10](#_Toc498985533)

[3.4 Architecture for My places 11](#_Toc498985534)

[3.5 Architecture for Profile 11](#_Toc498985535)

[3.6 Architecture for Help menu 12](#_Toc498985536)

[3.7 Architecture for Search place 12](#_Toc498985537)

[3.8 Architecture for CouchDB 13](#_Toc498985538)

[4 Project Management 16](#_Toc498985539)

[4.1 Milestones and Schedules 16](#_Toc498985540)

[4.2 Workload diagram 17](#_Toc498985541)

# Application Overview and Objectives

## Application Overview

The Spot is an online Content Management System (CMS) for storing different places of interest. The purpose of the system is to provide the registered in it users the ability to create, edit and delete Places in their own personal pages. Search and import functionalities in/from Google Maps are also implemented in the application. The CMS will be accessible both through desktop and mobile devices. All the information for the application (user logins, user passwords, stored places, etc.) will be stored in a server, and it is accessible through any device, connected to Internet. The key-users here are people who want to plan a trip, keep points of interested close, at hand, or just for scouting the globe.

## Objectives

The application has two sides: a client-side and a server-side.

1. Client-side - also called front-end, consists of all the user interfaces, functionalities and visual designs the user sees and interacts with. It is created using WEB languages and libraries (HTML, CSS, JavaScript, Google Maps API for WEB). The functionalities of this side are:
   1. Search - ability to search for different places in the supported databases;
   2. Add - ability to selected places from the databases and storing them in a personal database in the application;
   3. View - ability to view all saved places in the personal database;
   4. Edit - ability to edit a saved place in the personal database;
   5. Delete - ability to delete a saved place in the personal database;
2. Server-side - also called back-end, is the data center of the application. It is a NodeJS server using Apache CouchDB for the database management system. All the data is stored as documents in JSON format and CouchDB allows for easy creation of nodes (objects), editing and deleting of existing ones. The functionalities of this side are:
   1. Create a document - a document represents a user profile. When a user is registered a document is created to store all his personal data and stored places;
   2. Create a node - a node is a JSON object stored in a document. It contains information of the user like profile credentials and stored places;
   3. Edit - all nodes can be edited to keep the information up to date;
   4. Delete - a node or a document can be deleted if it is no longer in use;

# Use Cases

Identify the main actors and use cases of the application.

## Roles and Actors

This section lists the roles interacting with the application and provides a short description.

### Administrator

The administrator has full control over the system. He must be a well-educated WEB developer to be able to fix problems in the system if the occur. He has access to the database and user interface functions.

### User

The users are the main target group for this application. He will be able to log in the system and use all of its functions. If problems occur and a user can noto handle them himself, he should turn to the help menu of to the Administrator.

## Use Case Diagrams

### Register

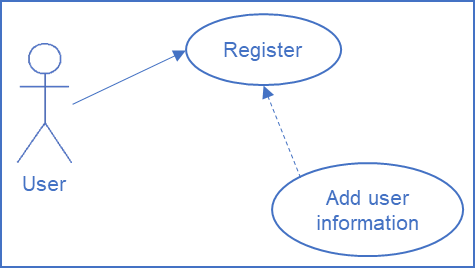


Figure 1

### Login

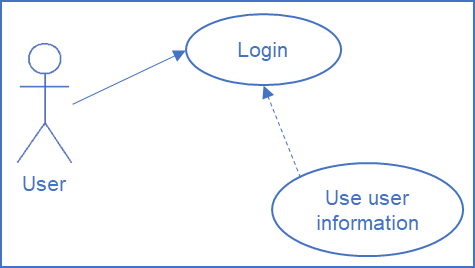


Figure 2

### Search

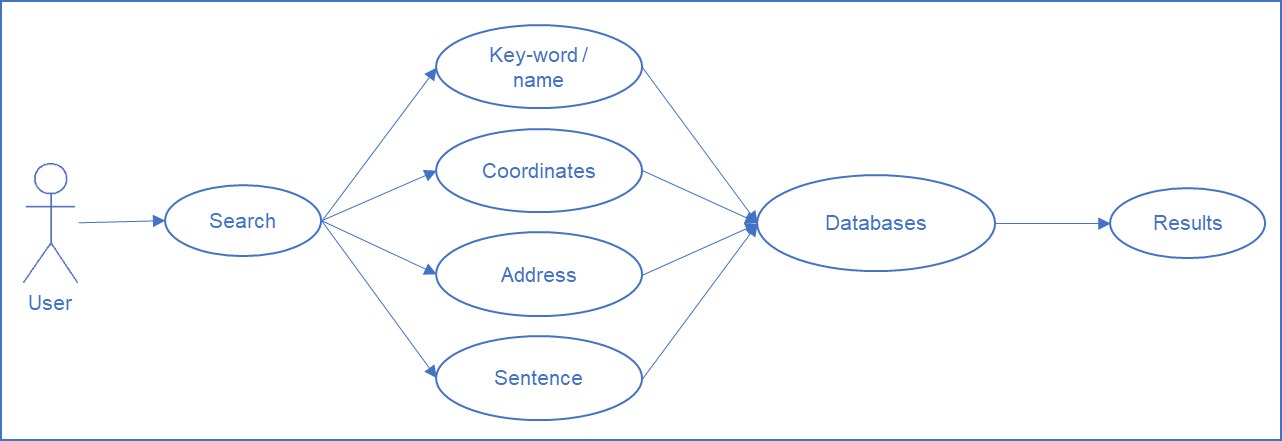


Figure 3

### Save a place

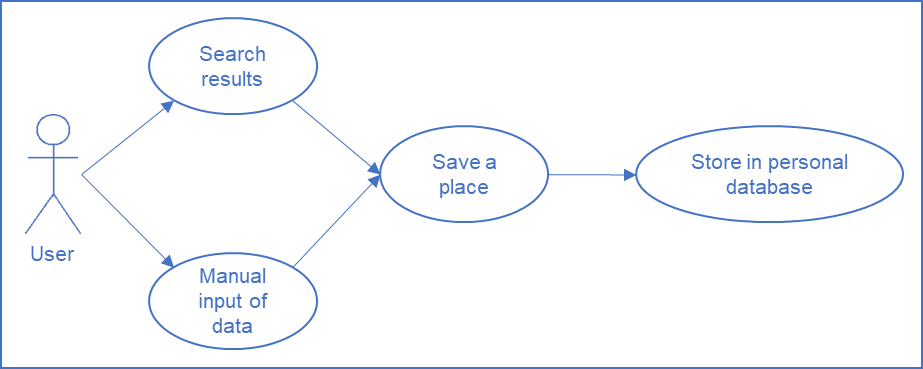


Figure 4

### Edit profile

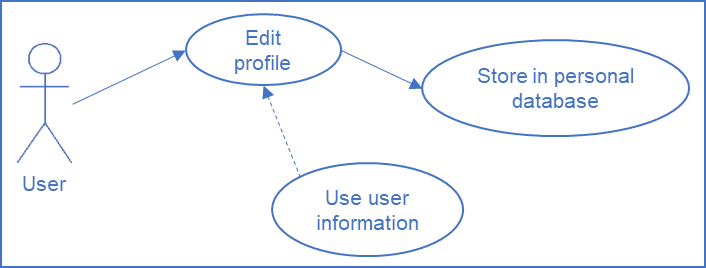


Figure 5

### Troubleshooting

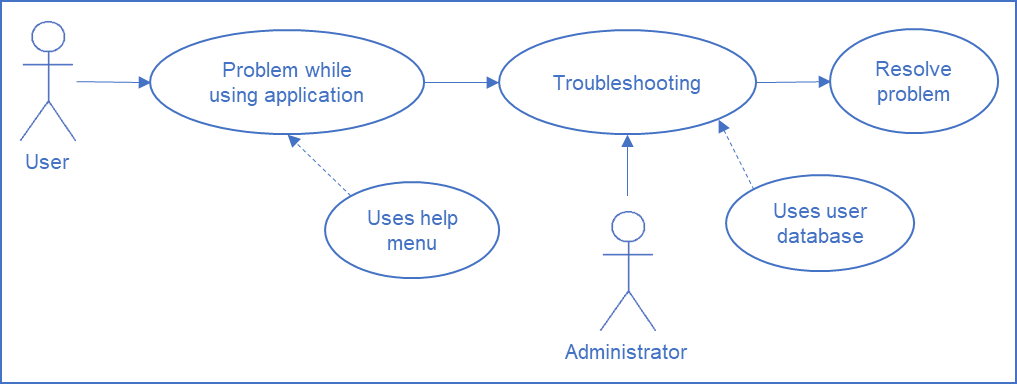


Figure 6

## Textual Use Case Descriptions

Add textual use case descriptions for each use case defined in the Use Case Diagram. Use the provided form:

|  |  |
| --- | --- |
| **Category** | **Entry** |
| Use Case ID | **UCD01** |
| Title | **Register** |
| Scope | The Spot |
| Actors | User |
| Preconditions | The User has not been registered before |
| Success Guarantees | The User is registered in the system and can log in with his/her e-mail and password |
| Trigger | A User wants to use the functionalities of the application |
| Main Success Scenario | 1. The User opens the registration form from the main page; 2. The User enters the user details (e-mail, password, first name, last name and birthdate); 3. The registration is complete; 4. The User logs into his/her account; |
| Exceptions | 1. The User is already registered - a message is displayed that the User already has an account and invites him/her to log in; |
| Version | 0.1 |
| Last updated | 08-Oct-2017 |
| Issues/Comments |  |

Table 1

|  |  |
| --- | --- |
| **Category** | **Entry** |
| **Use Case ID** | **UCD02** |
| **Title** | **Login** |
| Scope | The Spot |
| Actors | User, Administrator |
| Preconditions | The User is registered in the system |
| Success Guarantees | The User can log in with his/her username and password |
| Trigger | A User wants to use the functionalities of the application |
| Main Success Scenario | 1. The User enters his/her username and password; 2. The User logs into his/her account; |
| Exceptions | 1. The User’s profile has been deleted - a new registration is needed; 2. The User’s data contains errors, making the application unresponsive - contact Administrator to restore data; |
| Version | 0.1 |
| Last updated | 08-Oct-2017 |
| Issues/Comments |  |

Table 2

|  |  |
| --- | --- |
| **Category** | **Entry** |
| **Use Case ID** | **UCD03** |
| **Title** | **Search** |
| Scope | The Spot |
| Actors | User |
| Preconditions | The User is logged in the system |
| Success Guarantees | The User receives results for his/her search query |
| Trigger | A User wants to find a specific place from the databases |
| Main Success Scenario | 1. The User opens the search interface; 2. The User selects the type of search (key-word/name, coordinates, address or sentence); 3. The User fills in the required fields, depending on the search type; 4. A search is performed in the databases; 5. Results are displayed in the interface; |
| Exceptions | 1. Not all required fields are filled - an error message will appear next to the field; 2. The query did no yield any results - a info message will appear to inform the user; |
| Version | 0.1 |
| Last updated | 08-Oct-2017 |
| Issues/Comments |  |

Table 3

|  |  |
| --- | --- |
| **Category** | **Entry** |
| **Use Case ID** | **UCD04** |
| **Title** | **Save a place** |
| Scope | The Spot |
| Actors | User |
| Preconditions | The User is logged in the system |
| Success Guarantees | The User saves a place (city, monument, monument, etc.) in his personal database |
| Trigger | A User wants to save a specific place to his/her personal database |
| Main Success Scenario | 1. The User opens the search interface; 2. The User searches for a place and selects one of the results OR selects to create his/her own place; 3. The User fills in additional information if needed; 4. The new data is saved; |
| Exceptions |  |
| Version | 0.1 |
| Last updated | 08-Oct-2017 |
| Issues/Comments |  |

Table 4

|  |  |
| --- | --- |
| **Category** | **Entry** |
| **Use Case ID** | **UCD05** |
| **Title** | **Edit profile** |
| Scope | The Spots |
| Actors | User |
| Preconditions | The User is logged in the system |
| Success Guarantees | The User makes changes on his/her personal information |
| Trigger | A User wants to change his/her personal information |
| Main Success Scenario | 1. The User opens the profile interface; 2. The User makes changes to the desired fields (new password, first name, last name and birthdate); 3. The User enters his current password; 4. The new data is saved; |
| Exceptions | 1. The User makes a mistake typing his password or he has forgotten it - contact Administrator; |
| Version | 0.1 |
| Last updated | 08-Oct-2017 |
| Issues/Comments |  |

Table 5

# Architecture

## Overview

Identify and describe the main application components and their interdependencies. Illustrate them using an UML component diagram. The Spot is made of 8 components and the interaction of the user with them and the communication between the make the functionalities of the application. The interdependencies between the components are shown Figure 7.

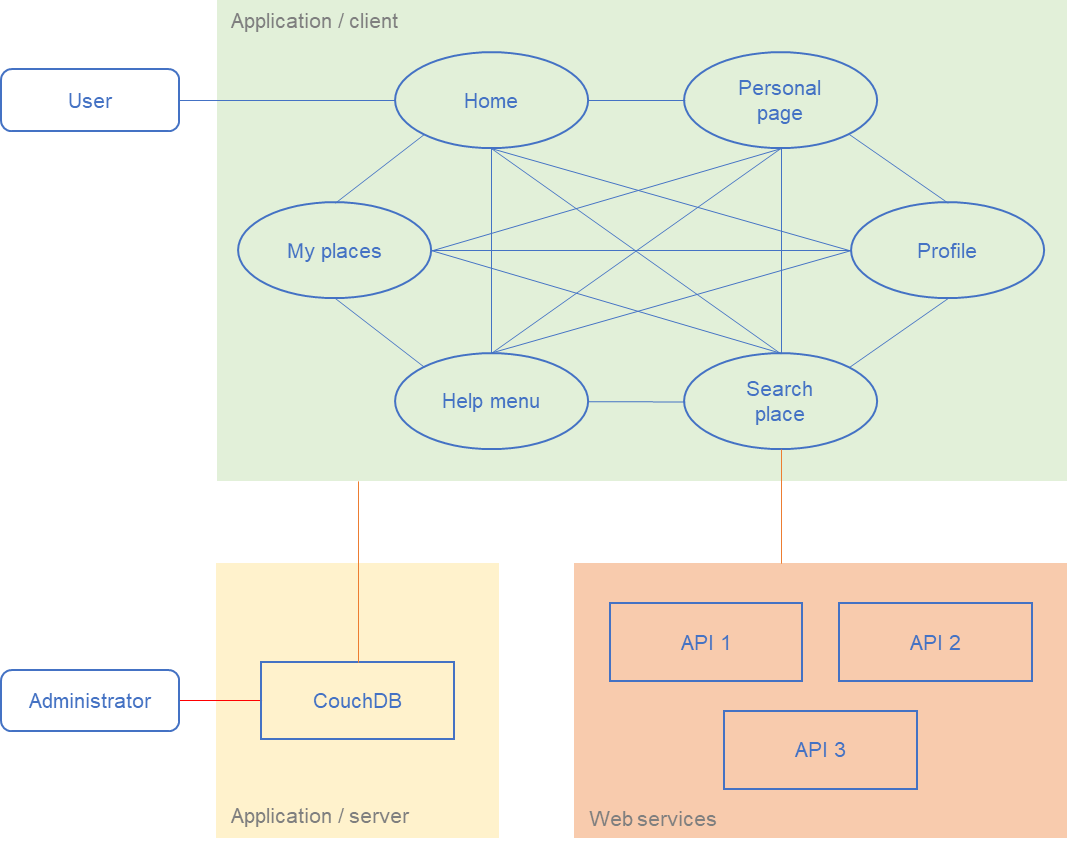


Figure 7

## Architecture for Start page

#### Design Considerations

The Start page is the first thing the User sees when he opens The Spot. It provides the possibilities to login, register or get additional information for the application.

#### Overview of data design

The Start page uses the User’s data. If he/she is registered the data is loaded from CouchDB. If the Users registered he will be asked to fill up his personal data, for it to be stored in the database.

#### Interfaces to/from internal and external components

The Start page communicates with CouchDB to get or store the relevant information of the User in a JSON format.

#### User Interface (if applicable)

The Start page functionalities are represented by three buttons:

* Login - opens a form for the User to input his e-mail and password to log in;
* Register - opens a form for the User to fill in his information to be able to use The Spot. All input fields in this form have data validation to ensure a correct store of data and limiting human errors;
* About - opens a window with information about The Spot;

#### Cross References

* [UCD01](#_Textual_Use_Case);
* [UCD02](#_Textual_Use_Case);

## Architecture for Home

#### Design Considerations

The Home is the home page of the logged in User. It provides an overview of the User’s data, shows the User’s saved places on a map and allows navigating to other pages of The Spot.

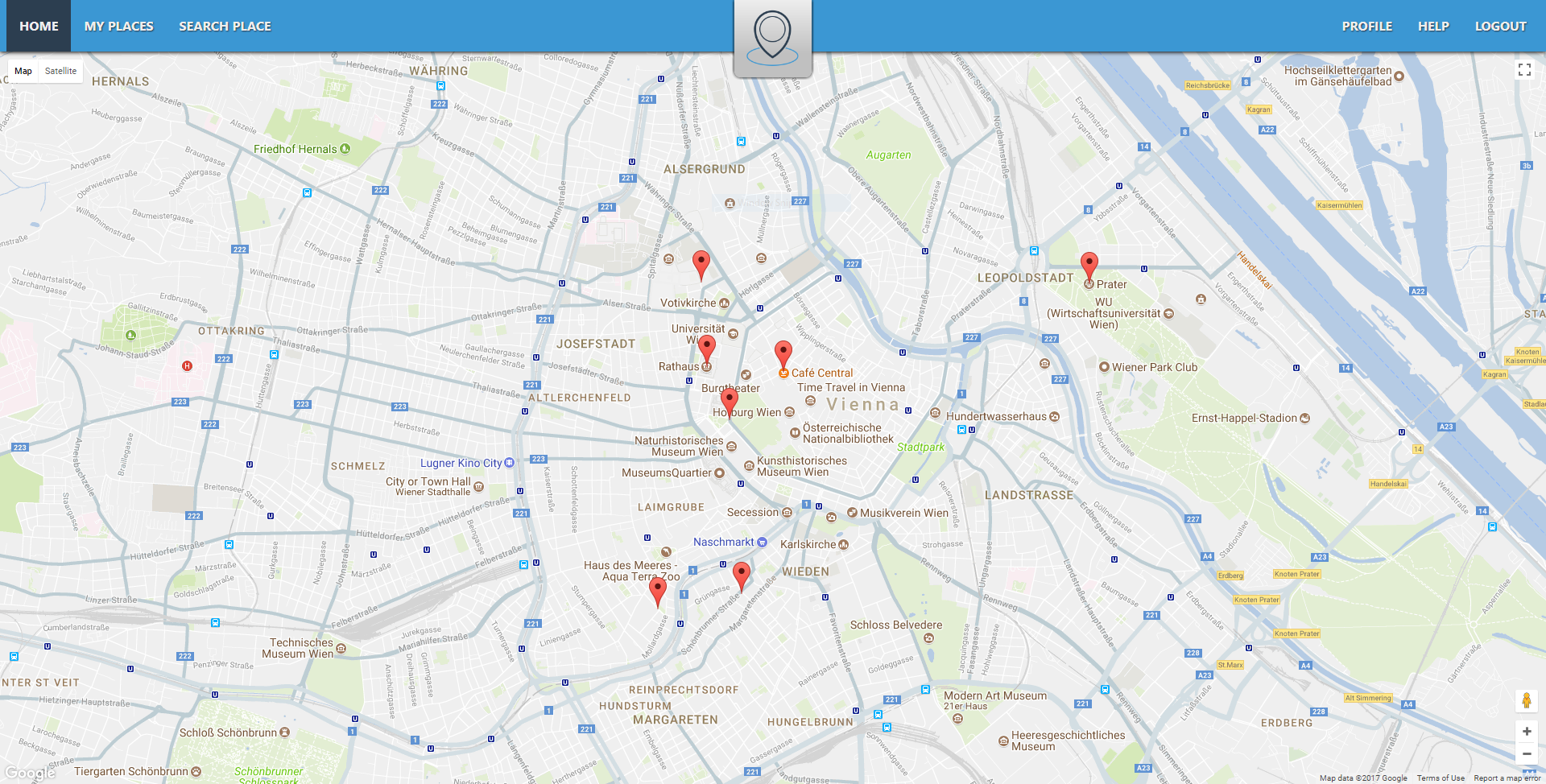
#### Overview of data design

This component uses the User’s data loaded from CouchDB.

#### Interfaces to/from internal and external components

Using the navigation bar at the top of the page the User can navigate to the other components: My places, Search place, Profile, Help menu and Start page (logout). Additionally, the component communicates with CouchDB to retrieve relevant information.

#### User Interface (if applicable)



The Home functionalities are represented by:

* Navigation bar - allows navigation between the application’s components;
* Map - shows visible places stored in the User’s profile;

#### Cross References

* No associated use cases;

## Architecture for My places

#### Design Considerations

My places focuses the User’s attention on his stored landmarks, displaying not only where they are on the map but also options for editing and deleting. The component provides functionalities as searching and filtering, as well (valid only for stored places).

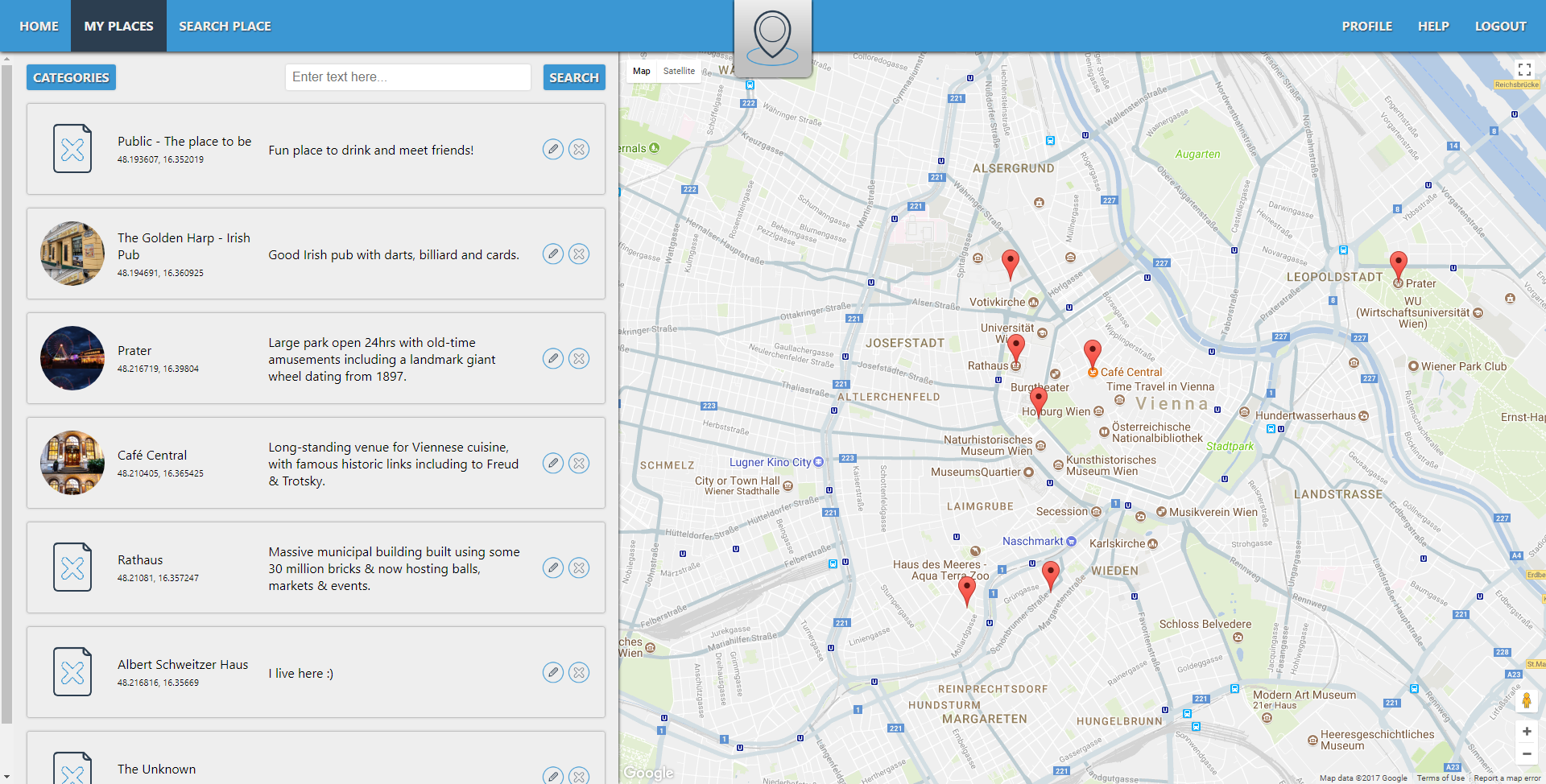
#### Overview of data design

This component uses the User’s data loaded from CouchDB.

#### Interfaces to/from internal and external components

Using the navigation bar at the top of the page the User can navigate to the other components: Home, Search place, Profile, Help menu and Start page (logout). Additionally, the component communicates with CouchDB to retrieve relevant information.

#### User Interface (if applicable)



My places functionalities are represented by:

* Navigation bar - allows navigation between the application’s components;
* Map - shows visible places stored in the User’s profile;
* List of places - a list of all the stored places by the User with options for searching and filtering;

#### Cross References

* No associated use cases;

## Architecture for Profile

#### Design Considerations

Profile allows the User to view, edit and save his/her personal information (password, first name, last name and birthdate).

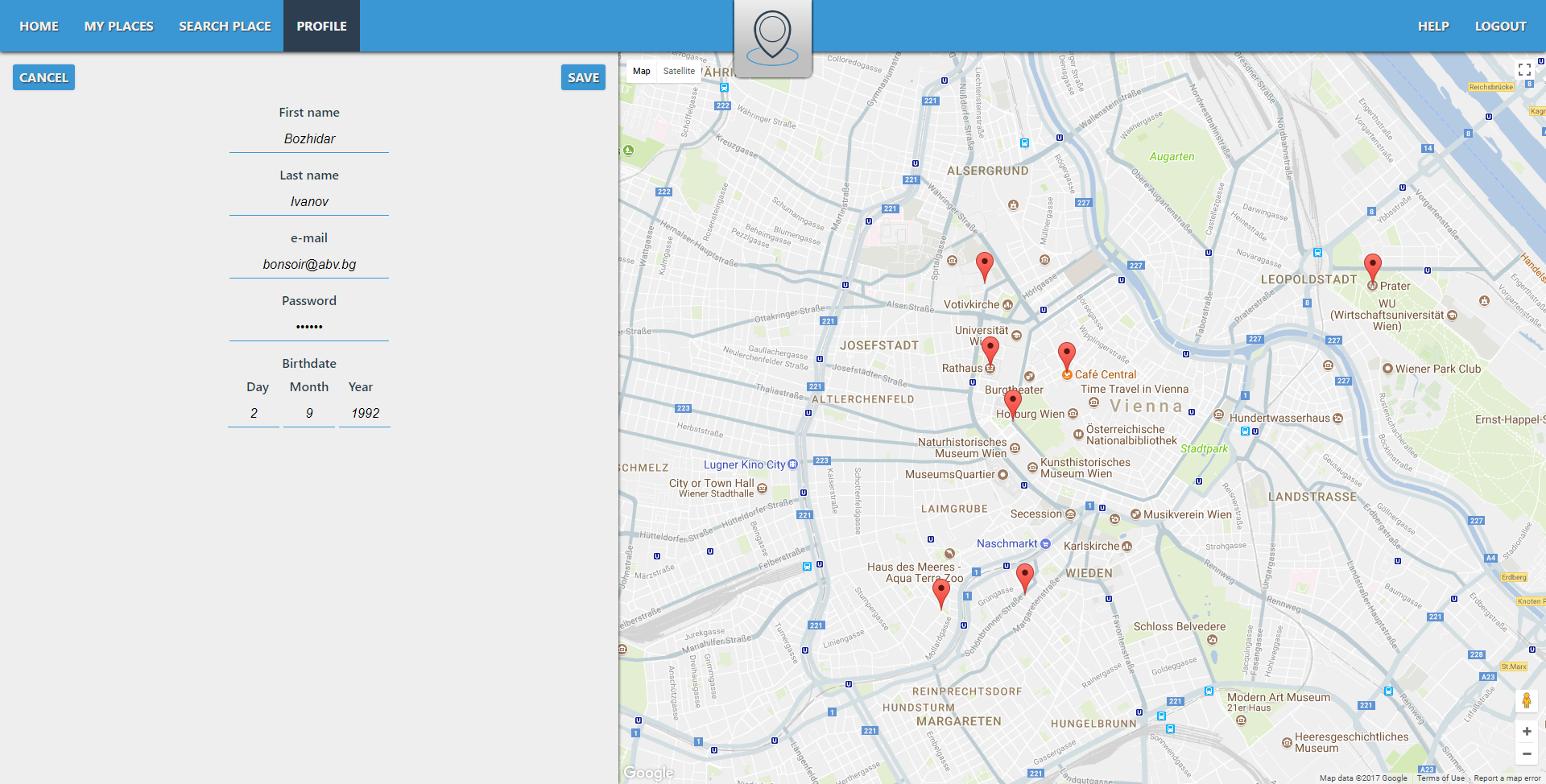
#### Overview of data design

This component uses the User’s data loaded from CouchDB.

#### Interfaces to/from internal and external components

Using the navigation bar at the top of the page the User can navigate to the other components: Home, My places, Search place, Help menu and Start page (logout). Additionally, the component communicates with CouchDB to retrieve relevant information.

#### User Interface (if applicable)



Profile functionalities are represented by:

* Navigation bar - allows navigation between the application’s components;
* Form - all input fields in this form have data validation to ensure a correct store of data and limiting human errors;

#### Cross References

* [UCD05](#_Textual_Use_Case)

## Architecture for Help menu

#### Design Considerations

The Help menu is a user guide describing how to use The Spot. It also contains useful tips, solutions to frequent troubles using the user interface and a contact form for further assistance.

## Architecture for Search place

#### Design Considerations

Search place is designed to allow the User to search and add places to his/her personal database. The user interface provides a map and a search form for finding the desired landmark.

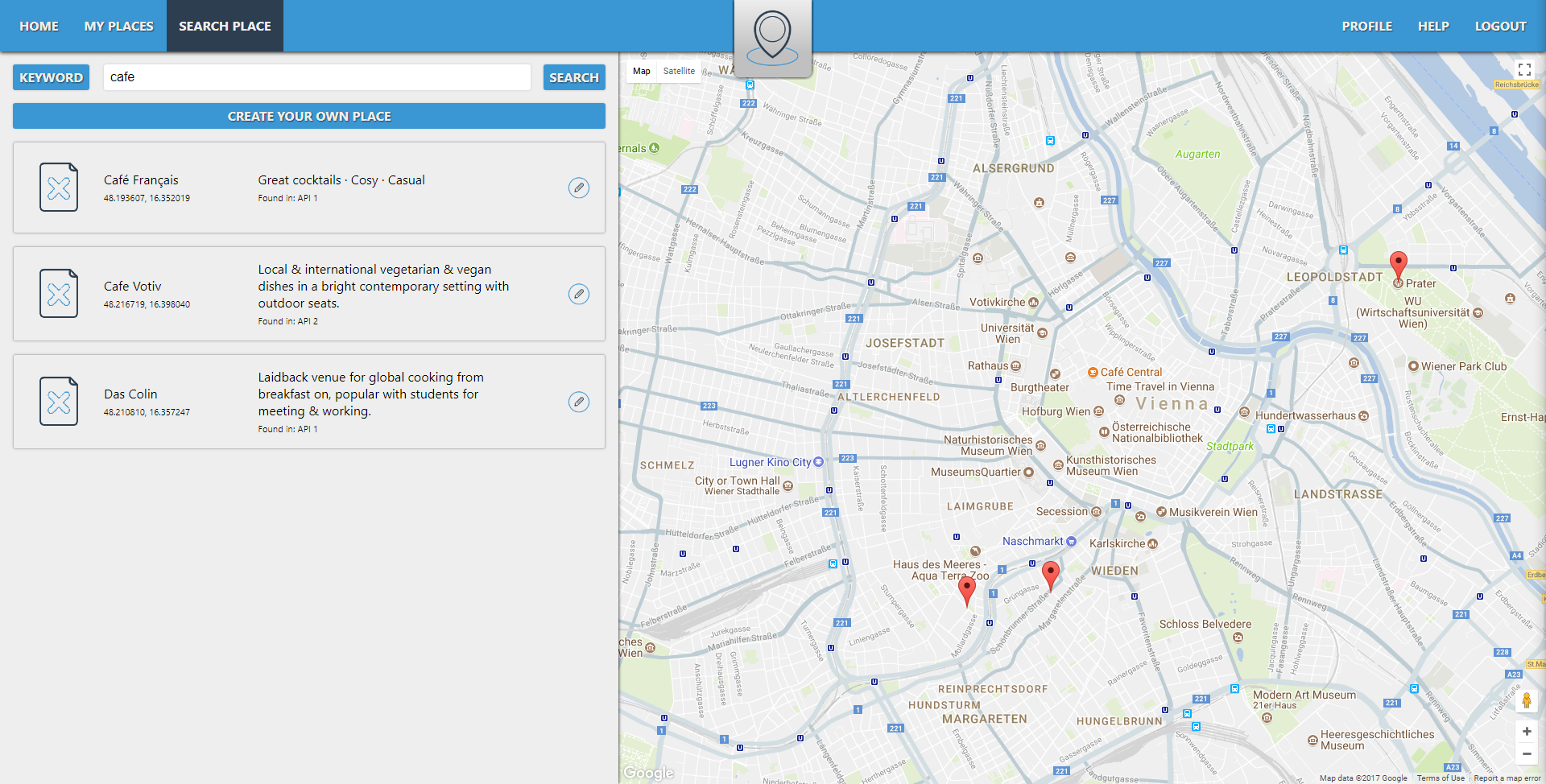
#### Overview of data design

This component uses the User’s data loaded from CouchDB and external APIs to gather more data.

#### Interfaces to/from internal and external components

Using the navigation bar at the top of the page the User can navigate to the other components: Home, My places, Profile, Help menu and Start page (logout). Additionally, the component communicates with CouchDB to retrieve relevant information.

#### User Interface (if applicable)



Profile functionalities are represented by:

* Navigation bar - allows navigation between the application’s components;
* Search form - made of a search field for the query input and a dropdown for the type of search (key-word/name, coordinates, address or sentence) this form sends a query to external APIs to find corresponding data;
* Result list - list with all the results from the query;
* Create button - allowing the User to create his/her own place if it is not found in the external APIs (name and coordinates are obligatory fields);

#### Cross References

* [UCD03](#_Textual_Use_Case);
* [UCD04](#_Textual_Use_Case);

## Architecture for CouchDB

#### Design Considerations

The database is designed to store securely the users’ information. All the data is stored on a NodeJS server, using Apache CouchDB for the database management system.

#### Overview of data design

The data in CouchDB is represented in a form of documents. Each document contains a contains data in a JSON format. For this application there are two types of documents: an all\_users (single) document and a user documents (multiple).

The first one contains all the registered users with their mail (name of object), password and id (properties of object). An example of the data structure of the all\_users document is presented in Code snippet 1.

The second type of document contains all the data (personal information and stored places) for a specific user. An example of the data structure of a user document type is presented in Code snippet 2. After a user has logged in the system will find the associated data by his/her user id (id of user document).

{

<user 1 e-mail> : {

password: "string",

id: "string"

},

<user 2 e-mail> : {

password: "string",

id: "string"

},

<user 3 e-mail> : {

password: "string",

id: "string"

},

{ ... },

{ ... }, ...

{ ... }

}

Code snippet 1

{

id: "string",

mail: "string",

password: "string",

firstName: "string",

lastName: "string",

birthdate: {

day: integer,

month: integer,

year: integer

},

places: [{

id: "string",

name: "string",

image: "string",

latitude: float,

longitude: float,

address: {

street: "string",

number: "string",

city: "string",

postalCode: "string"

},

description: "string",

category: [],

website: "string",

phone: "string"

},

{ ... },

{ ... }, ...

{ ... }

]

}

Code snippet 2

#### Interfaces to/from internal and external components

CouchDB provides the necessary user data for the application to work properly. It communicates with the other components (Start page, Home, My places, Profile and Search place) responding to queries send by them.

# Project Management

## Milestones and Schedules

All the tasks and sub-tasks of this project are presented in Table 6 with their start and end dates. The milestones are marked with blue background. All tasks will be completed by Bozhidar Ivanov.

|  |  |  |  |
| --- | --- | --- | --- |
| **Task** | **Sub-tasks** | **Start date** | **End date** |
| Requirements and design document | Fill-in project information | 02/10/2017 | 20/112017 |
| Add screenshots | 15/11/2017 | 20/11/2017 |
| **Requirements and design document submitted** | | | **20/11/2017** |
| Database | Add user | 21/11/2017 | 22/11/2017 |
| Edit user | 21/11/2017 | 22/11/2017 |
| Delete user | 21/11/2017 | 22/11/2017 |
| Add place | 22/11/2017 | 23/11/2017 |
| Edit place | 22/11/2017 | 23/11/2017 |
| Delete place | 22/11/2017 | 23/11/2017 |
| **Server-side finished** | | | **23/11/2017** |
| Start page | User interface | 24/11/2017 | 25/11/2017 |
| Connection to database services | 24/11/2017 | 25/11/2017 |
| Home | User interface | 25/11/2017 | 27/11/2017 |
| Connection to database services | 25/11/2017 | 27/11/2017 |
| My places | User interface | 27/11/2017 | 29/11/2017 |
| Connection to database services | 27/11/2017 | 29/11/2017 |
| Search place | User interface | 29/11/2017 | 02/12/2017 |
| Connection to database services | 29/11/2017 | 02/12/2017 |
| Profile | User interface | 02/12/2017 | 03/12/2017 |
| Connection to database services | 02/12/2017 | 03/12/2017 |
| Help menu | User interface | 03/12/2017 | 04/12/2017 |
| **Client-side finished** | | | **05/12/2017** |
| External APIs | Integrating API 1 | 05/12/2017 | 06/12/2017 |
| Integrating API 2 | 07/12/2017 | 08/12/2017 |
| Integrating API 3 | 09/12/2017 | 10/12/2017 |
| Testing | Whole system (bug fixes) | 12/12/2017 | 15/12/2017 |
| Final presentation preparation | Slides | 16/12/2017 | 19/12/2017 |
| Video | 20/12/2017 | 21/12/2017 |
| Demo | 22/12/2017 | 23/12/2017 |
| **Final presentation** | | | **29/01/2018** |

Table 6

## Workload diagram

All the sub-tasks are presented in Figure 8 as a Gantt chart to show the whole development period.

