University of Innsbruck
Wednesday 9 December 2015

703128 PS/2 Web Services

Project Mid-Term Report

Team Name : HATI (Hiking and Transportation in Innsbruck)

No.	Matrikel Nr.	Name
1	1316136	Bernhard Fritz
2	1318235	Thibault Gerrier
3	1217362	Raphael Gruber
4	1315434	Philipp Häusle
5	1315417	Mario Zelger

Project Description

Describe your project idea, settings and technical and usage restrictions, limitations (they can be listed using bullet points, providing sufficient details).

The idea of HATI is a web application which offers you possible hiking routes around Innsbruck. The routes are visualized in a map and more information about the routes like length or duration is given. Most of these routes don't start in the center of Innsbruck so you have to get to the start point of the route. For that problem HATI also shows you how to get there using public transport (buses from IVB). As a feature our application also provides weather data in the form of forecasts for the next 5 days, so you can plan your trip accordingly.
HATI will be hosted on www.openshift.com for an easy access.
An example usage of HATI:

1. You are in Innsbruck (e.g. as a tourist) and plan to do a hiking trip.
2. Start HATI and choose one of the offered beautiful hiking routes all around Innsbruck.
3. Check the route informations (also pictures in most of the routes) and confirm your selection.
4. Based on your current position HATI will show you how to get to the start point of the route using public transport.
5. The application also shows the weather forecast for the trip in the area of the selected hiking route.
6. After you have started your trip you are always able to visualize your current position on the map and check it with the hiking route also drawn on the map.
7. Simply follow the route to the end point and enjoy your trip!

HATI combines hiking routes, public transport and weather information to offer beautiful recommended hiking routes around Innsbruck and how to get there easily. The weather forecast is a useful feature to plan the trip accordingly.
HATI is an easy way to get to know the beauty of nature around Innsbruck.

Project Technical Solution

Describe the technical solution of your project, particularly, depict an architecture and list and describe technical components and interactions among them, used datasets, servers, data structures, databases/repositories, etc.

For this project we decided to use node.js to implement our webserver. Node.js gives you the freedom to organize your project as you would like to. Even if we could write our own webserver using node.js right out of the box, we decided to use express.js, a npm (node package manager) package, to make our lifes easier. Express.js gives you the opportunity to program in a MVC (model, view, control) like fashion. The good thing about MVC pattern is that it allows you to swap an entire part (either model, view or control) in a later stage of your project. This can be valuable if you realize you made a mistake in the planning stage. Express.js also allows the server to handle requests with ease. A simple JavaScript function will be called if the server receives a request.

Our view consists of a web app which is hosted on our exress.js server. This web app is implemented using Jade, a node templating engine. Jade allows us to program dynamic web pages by giving us an interface to share variables between server and client. Once Jade is 'rendered' the client will receive a plain HTML page. Bootstrap is used to make our web app better looking and responsive.

The web app is basically an interface to the server's functionality. The server handles all requests to other APIs like openstreetmap, outdooractive and openweathermap. It can be considered as a wrapper which modifies the data received by APIs accordingly and presents the result to the client in a pleasant way. In it's current state the web app provides functionality to choose from a list of tours or search for a specific one by entering a keyword. These tours are provided by outdooractive API. Once the user found a tour (s)he can check out tour details like description, length of the tour, altitude or duration. In addition to that, the user has the opportunity to visualize the route on openstreetmap. A weather forecast provided by openweathermap API allows the user to plan the trip accordingly.

We made a wrapper for each API that we are using:

- openStreetMap.js:
  Allows us to fetch Relations and Nodes provided by openstreetmap API. Relation can be considered as a collection of related nodes. For example a bus route consists of a relation of bus stops. Each bus stop has a corresponding openstreetmap node which includes valuable information like bus stop name as well as latitude and longitude coordinates. By using the user's geolocation and the bus stop coordinates we are able to guide the user to nearest bus stop.

  Since IVB (Innsbrucker Verkehrsbetriebe) does not provide open data we decided to write our own algorithm to determine the shortest path between two busstops. This will be accomplished by using the aquired data from openstreetmap in a graph like data structure and Djikstra shortest path algorithm for weighted directed graphs.
- outdooractive.js:
  Outdooractive offers tons of tours for hikers, as well as bikers all around the world. By using specific requests we were able to filter for hiking tours around Innsbruck. Some tours were way

to long in length considered as tour in Innsbruck, though. That's why we decided to only consider tours which are shorter than 15km and require less than 6 hours.

- openweathermap.js:
  This wrapper offers functions to retrieve current weather data from openweathermap API by GPS coordinates or by city name. The provided information consists of temperature, humidity, windspeed, atmospheric pressure information as well as basic indicators of weather states like cloudy, rainy, clear sky and so on.

Ultimately this project will be hosted on www.openshift.com. Openshift is a free solution to host node.js on the internet. To host a node.js project on openshift, it's as simple as modifying some express.js ports to make it compatible with openshift and pushing the whole project to openshift's private git repository.

Included Services

List and describe all external services you have currently included in your project and for each service, describe the following aspects:

1. What technologies do you currently use to communicate with the service (e.g. HTTP, SOAP, REST)
2. What data format do you currently use to communicate with the service (e.g. XML, JSON)
3. Which programming libraries do you currently use to send requests and analyze the received responses from the service, please be specific (e.g. Jersey, Spring Social, PHP, SOAP)

No.	Name of the used external services	Description (e.g. technical details for the service and its use, technologies involved, other remarks)
1	Openstreetmap API	1: We use HTTP GET requests to retrieve bus stations from the Openstreetmap REST API 2: The Openstreetmap REST API responds with XML which we convert to JSON using a npm package called "xml2json" (https://www.npmjs.com/package/xml2json). The JSON data can be parsed to a JavaScript object using the JSON.parse() function. 3: We use a npm package called "request" (https://www.npmjs.com/package/request) Example request to http://api.openstreetmap.org: /api/0.6/relation/2611781 Yields information about the bus line O including all bus stops as well as their GPS coordinates.
2	Outdooractive API	1: We use HTTP GET requests to retrieve tours around Innsbruck from the Outdooractive REST API 2: The Outdooractive REST API responds with JSON which can be parsed to a JavaScript object using the JSON.parse() function. 3: We use a npm package called "request" Example request to http://www.outdooractive.com: /api/project/alpenvereinaktiv/filter/tour?key=yourtest-outdoora-ctiveapi&area=1022439&category=1566480 Yields ids to specific hiking tours around innsbruck. These ids can used to retrieve more information in a seperate request.
3	Openweathermap API	1: We ues HTTP GET requests to retrieve Innsbruck's weather infromation from the Openweathermap REST API 2: The Openweathermap REST API responds with JSON which can be parsed to a JavaScript object using the JSON.parse() function. 3: We use a npm package called "request"

		Example request to http://api.openweathermap.org: /data/2.5/weather?q=Innsbruck&units=metric &appid=2de143494c0b295cca9337e1e96b00e0 Yields current weather information for Innsbruck.
4	Yournavigation API	1: We use HTTP GET requests to retrieve route information from the Yournavigation REST API 2: The Yournavigation REST API responds with JSON which can be parsed to a JavaScript object using the JSON.parse() function. 3: We use a npm package called “request” Example request to http://www.yournavigation.org: /api/1.0/gosmore.php? format=geojson&flat=...&flon=...&tlat=...&tlon=... Yields route information from GPS coordinates flat, flon to GPS coordinates tlat, tlon. This is useful for us to guide the user to the next bus stop.