# 4. Tools & Technologies

In this report section, the tools and technologies used during the course of the projects are presented. The different components of the geographic database are described as well as the tools to design the conversational interface. Additionally, some alternatives to the eventually used tools are discussed in order to explain why the specific tools are chosen.

## 4.1 Geographic database

In order to extract tourist points of interests from *OpenStreetMap*, a geographic database was set up as the application’s backend. The following tools form either a part of the mentioned database or were used to set it up.

### 4.1.1 PostgreSQL 9.5.5

*PostgresSQL*1 is an object-relational database system. It is an open source software that is most extensively conform to the SQL standard ANSI-SQL 2008 ­. In this project, PostgreSQL is used to store and manage the geographical database.

### 4.1.2 PostGIS 2.2.1

*PostGIS* 2 is an open source 3 spatial extension for the object-relational database system *PostgreSQL*. The extension allows the database system to manage geographic objects and run spatial queries in *SQL*. Combined, *PostgreSQL* and *PostGIS* form a geographic database that can be used in geographic information systems, such as *QGIS* 4.

### 4.1.3 pgAdmin III 1.22.0

*pgAdmin III* 5 is an open source tool to facilitate the management of *PostgreSQL* databases by providing a graphic user interface. Among many features, it comes with an SQL editor tool to create and run queries and displays data entries 6.

### 4.1.4 Osmosis 0.44.1

Osmosis 7 is an open source command-line based application that is able to process data from *Open Street Maps.* In this project, it was mainly used to import data from OSM files into the PostgreSQL database.

At the beginning, the similar tool osm2psql was investigated and used in this project. However, during research it turned out that osm2psql is not the right fit because its main objective is the import of .osm files for rendering purposes. Due to this reason, only data that are render relevant are imported into the PostgresSQL database. On the other hand, Osmosis imports all of the raw .osm data, namely Nodes, Ways, Relations and their corresponding tags 8.

## 4.2. Chatbot platform

Due to the recent hype about chatbots and their integration into mobile apps, there are several competing platforms that permits developers to build their own chatbot. Most of the major software big-players nowadays, such as Google, Facebook, Microsoft, are taking part in the development of these platforms and provide their own solutions, for example API.ai (Google), wit.ai (Facebook), luis.ai (Microsoft) or IBM Watson. Out of these contestants, api.ai and wit.ai seem to be the most widely known 9 and easiest to use. Therefore, both options were investigated to see which one would be the most suitable for this project.

### 4.2.1 api.ai

api.ai 10 is a natural language processing platform by Google that facilitates creating conversational user interfaces. In order to model conversations, entities and intents are used as key concepts. api.ai provides a robust and rich management toolset as well as a simply one-click-integration mechanism to import chatbot into mobile apps. *api.ai comes in two versions, an enterprise option that provides the integration of different domains and an extensive support, whereas the free option has as its only limitation a limited bandwidth.*

### 4.2.2 wit.ai

Like api.ai, wit.ai is a natural language processing platform. It was acquired by Facebook in January 2015 and is free to use ever since then 11. Its key concepts for language processing include entities, intents and, additionally, stories. Stories help to create basic user scenarios in which typical user and chatbot conversations are designed 12­­­. Regarding integration, wit.ai provides a web service API to integrate the designed conversational interface in messaging apps.

### 4.2.3. Conclusion: Choice of platform (API.ai)

After testing both platforms, both left a good impression and seem to be suitable to use. On the one hand, wit.ai’s story feature seems promising to design conversations in an easy way, but is still in beta status. However, api.ai convinces with its rich management toolset, the one-click integration which makes it especially easy to integrate the conversational interface on several different mobile apps and an extensive documentation. Its only flaw, the split into a free and an enterprise plan, has consequences concerning a limited bandwidth which can be disregarded because the application is not presented to a big audience. For this reason, the free option seem to be more than sufficient, so api.ai is chosen in this project to design the conversational interface.

# Bibliography

1: <https://www.postgresql.org/about/>, 02.02.17

2: http://postgis.net/

3: <https://en.wikipedia.org/wiki/PostGIS>

4: <https://de.wikipedia.org/wiki/PostGIS>

5: https://www.pgadmin.org/

6: <https://www.pgadmin.org/features.php>

7: <https://wiki.openstreetmap.org/wiki/Osmosis>

8: <https://me4bruno.wordpress.com/2012/03/25/osmosis-osm2postgresql-osm2pgsql-openstreetmap-daten-datenbanken-und-spielplatze/>

9: Google Hits: API.ai 155.000, wit.ai 92.200, luis.ai (20.800) (31.1.2017)

10:

[https://docs.api.ai/docs/welcome](https://docs.api.ai/docs/welcome 11)

[11](https://docs.api.ai/docs/welcome 11): https://wit.ai/blog/2015/01/05/wit-ai-facebook

12: https://wit.ai/docs/quickstart