Design Document

**Teachers:** Kiavash Bahreini, Márcio Paixão Dantas

**Student Name:** Omar Abou Dehn

**Student Number:** 3560813

**Interface Git Repository:** https://git.fhict.nl/I407846/safar\_travelapp



Document Version History:

|  |  |  |
| --- | --- | --- |
| Version | Date | Changes |

Contents

[Entity Relational Diagram: 4](#_Toc99134420)

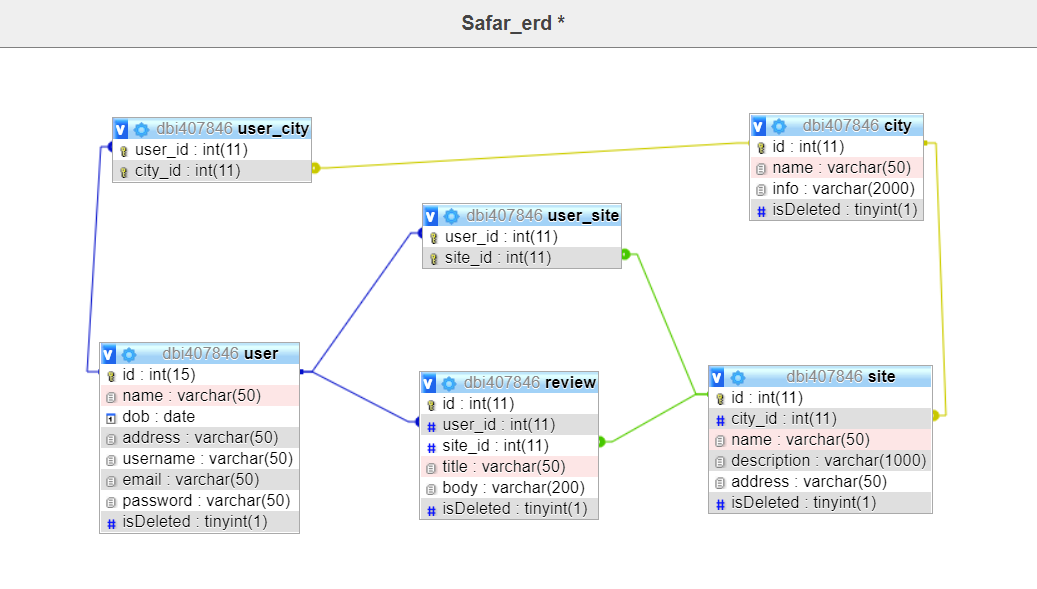
[UML Class Diagram: 5](#_Toc99134421)

[Backend Framework: 6](#_Toc99134422)

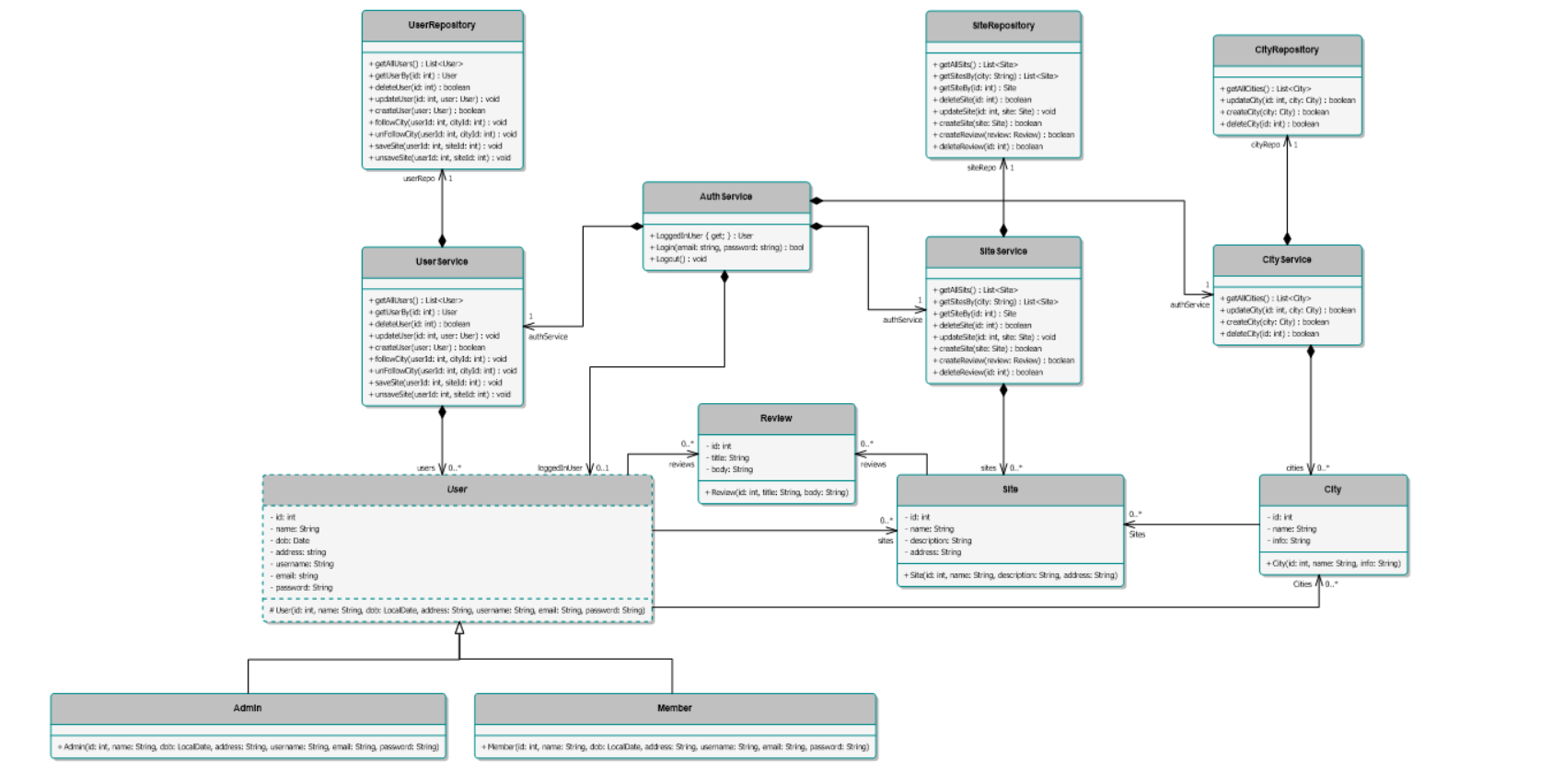
[Frontend Framework: 7](#_Toc99134423)

[Database Framework: 8](#_Toc99134424)

# Entity Relational Diagram:



# UML Class Diagram:



# Backend Framework:

Spring Boot will be used to build the restful API as it is one of the most powerful and famous Java frameworks, and due to its popularity, it has a good community and support and a lot of online resources.

# Frontend Framework:

In this chapter the three most used JS frontend frameworks will be compared based on different criteria in order to determine which one will be used to build Safar interface.

Angular: an application design framework and development platform for creating efficient and sophisticated single-page apps [(1)](https://angular.io/docs).

React: a JavaScript library for building user interfaces.

Vue: a progressive JS framework for building user interfaces.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Angular** | **React** | **Vue** |
| **User Friendly** | + | + | + |
| **Performance** | + | + | + |
| **Reusable UI Components** | + | + | + |
| **MVC Support** | + | - | - |
| **Documentation** | + | - | + |
| **Strong/Large Communities** | + | + | - |
| **Testability** | + | + | + |

[(2)](https://www.codeinwp.com/blog/angular-vs-vue-vs-react/), [(3)](https://athemes.com/guides/angular-vs-react-vs-vue/), [(4)](https://academind.com/tutorials/angular-vs-react-vs-vue-my-thoughts/), [(5)](https://medium.com/techmagic/reactjs-vs-angular5-vs-vue-js-what-to-choose-in-2018-b91e028fa91d), [(6)](https://blog.logrocket.com/angular-vs-react-vs-vue-a-performance-comparison/)

# Database Framework:

|  |  |  |
| --- | --- | --- |
|  | **JDBC** | **JPA** |
| **Abstraction Level** | Low level standard for interaction with databases | High level standard for interaction with databases |
| **Usage** | JDBC allows to do more things with the Database directly which gives more flexibility | JPA allows to use an object model in the application which requires less code |
| **Migrating Database** | Due to its low abstraction, a lot of changes are required when changing the database | Due to its high abstraction, the database can be switched by changing some configurations |