*Schejewel*

Product Design Specification

Version *1.0*

*2/14/2015*

TABLE OF CONTENTS

[1 Introduction 3](#_Toc180482593)

[1.1 Project Overview 3](#_Toc180482594)

[2 General Design 3](#_Toc180482595)

[2.1 Design Overview 3](#_Toc180482598)

[2.2 Assumptions / Constraints / Technologies 3](#_Toc180482596)

[3 Backend Design 4](#_Toc180482597)

[3.1 API 4](#_Toc180482600)

[3.2 Security 5](#_Toc180482601)

[3.3 Database 6](#_Toc180482602)

[3.4 Server 6](#_Toc180482602)

[4 FrontEnd Design 7](#_Toc180482604)

[4.5 User Interface 7](#_Toc180482609)

# Introduction

## Project Overview

Schejewel is a web based application used by excursion companies in Alaska.

# General Design

This section gives a general overview of the schejewel application and the chosen technologies used in the application.

## Design Overview

## 

## Assumptions / Constraints / Technologies

Frontend:

Technologies: Angular.js

Details: Angular front end pages being served by a node backend

Server:

Technologies: Express.js

Details: Express server that pulls data from the api and serves the angular font end pages

API:

Technologies: Java, Spring, Tomcat

Details: The API is written in the Java language. Java was chosen because of the ability to use the Spring framework to create secure REST API’s, as well as it being a common language amongst our team. The API is hosted using an Apache Tomcat Server.

Database:

Technologies: MySql

Details: The data being dealt with in this application is able to be easily modeled using SQL. MySql was chosen because it’s a reliable opensource relational database.

# Backend Design

This section outlines the specifics in design relating to the backend components of the schejewel project.

## API

## Technologies: Java, Spring

## Details: The API uses Spring to create RESTful endpoints accessible to the server. Below are endpoints required by the frontend and server.

* URL: "/api/login"
* TYPE: "POST"
* TOKEN: Base64 Encoded String in X-Token-Auth header.
* URL: "/api/user"
* TYPE: "POST"
* JSON: User object
* URL: "api/user/{user id}"
* TYPE: "GET"
* JSON: User object
* URL: "api/resource/{resource id}"
* TYPE: "GET"
* JSON: Resource Object
* URL: "api/resources"
* TYPE: "GET"
* JSON: List Resource Object
* URL: "api/event/{event i}"
* TYPE: “GET”
* JSON: Event Object

## Security

## Technologies: Spring Security, JSON Web Tokens, HTTPS

## Security Flow: The Schejewel Project is using token based authentication for security. First a user sends their username and password credentials to the server which then relays that data to the API (<http://schejewel.com/api/login>). The api then uses spring security to validate the user credentials. If valid a JSON web token containing the user data is then sent back to the server.

## 

## The server stores this token and uses it for subsequent calls to the API. After the initial login the API uses the token to validate the users and return the requested data. Note: All communication is done via HTTPS.

## 

## Database

## Technology: MySql

## Schema:

## 

## 

## server

## Technologies: Node.JS, Express.js

Details: Serves the front end web pages. It is used as an extra level of abstraction and to reduce the complexity of the API. Rather than the API having to worry about serving the webpages, it can focus on data manipulation.

# frontend Design

This section outlines the specifics in design relating to the frontend components of the schejewel project.

## User Interface

## Technologies: Javascript, Angular

Views:

* Dashboard: Shows an overview of the tours for the day, as well as messages that the user has recently received.
* Day view: Shows an overview of what exactly is going to happen on that day. More detailed than the dashboard view.
* Login view: Allows a user to log in
* Create resource view: Create a new resource, such as a boat or dining room, and assign it to an owner.
* Edit resource view: Change available times, owners, etc.. of a resource.