**Booking Flask Application Demo – Requirements, design and user cases document**

Authors: Kamil Wezka, Gavin James Jones

Date: 06.02.2018

Last update: 11.02.2018

Version 0.1.3

Contents

[Introduction 3](#_Toc505696813)

[Simple workflow 3](#_Toc505696814)

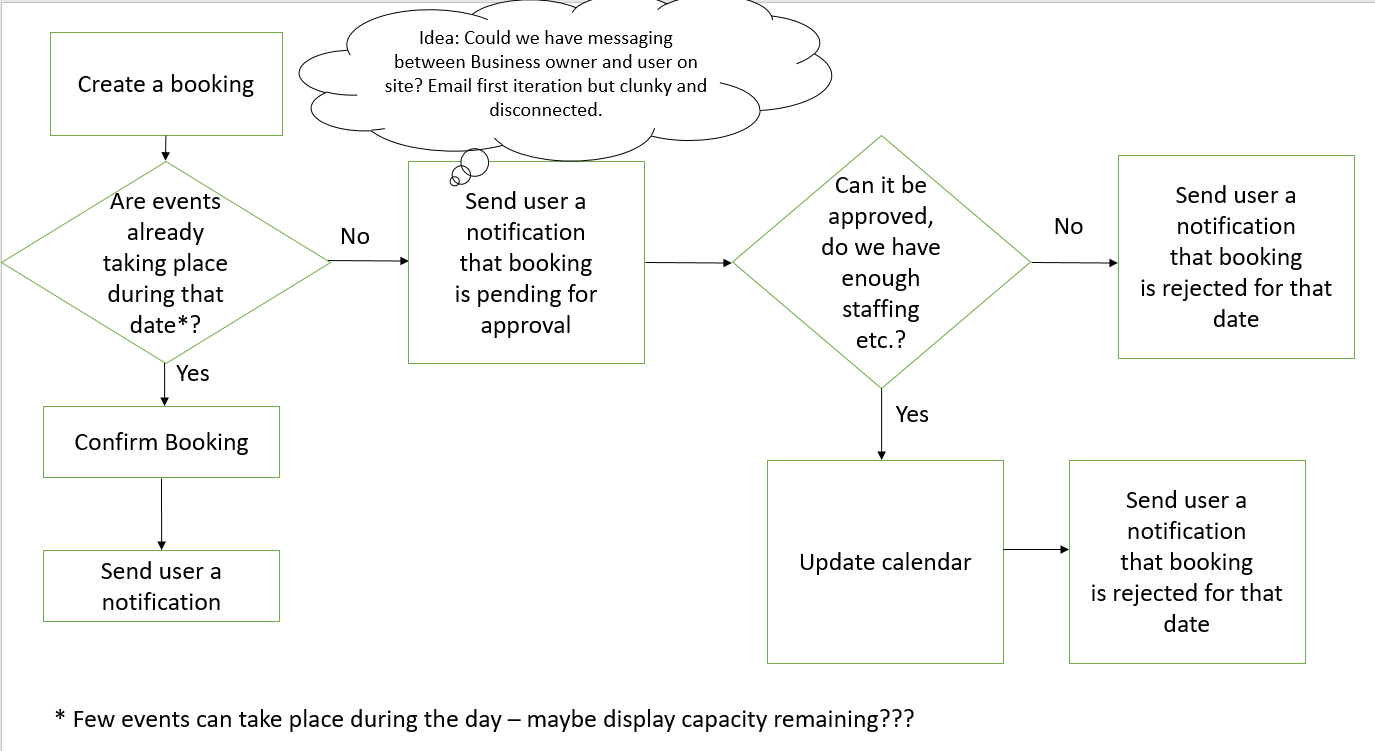
[User Scenarios – simple use 4](#_Toc505696815)

[User Scenarios – security 4](#_Toc505696816)

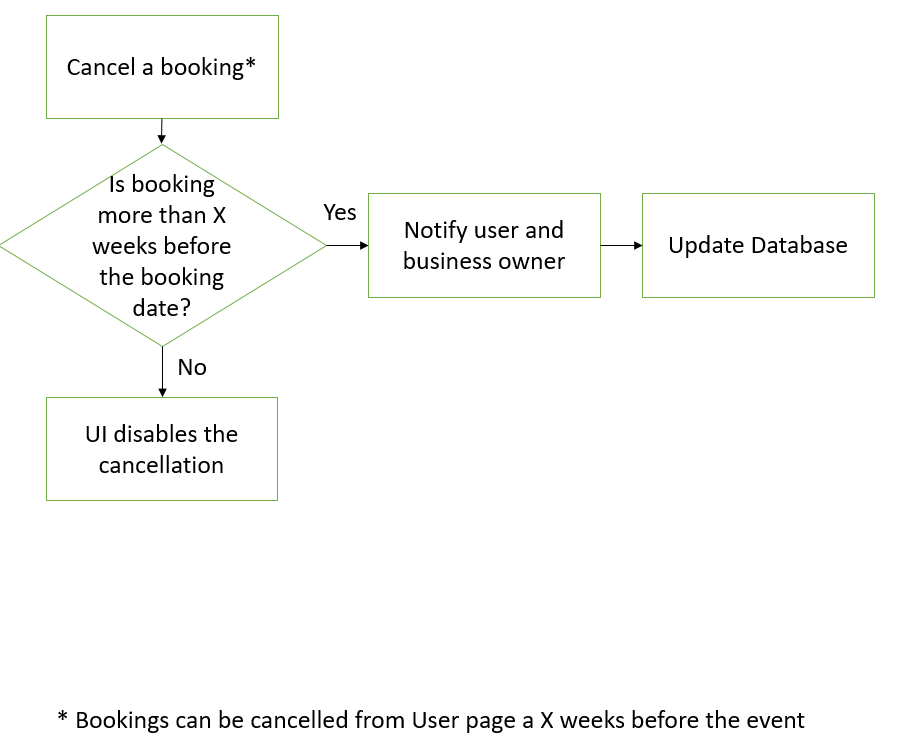
# Introduction

The documents summarise the initial specifications of what the booking application should contain i.e. list of features, user scenarios, infrastructure requirements etc.

# Simple workflow – booking



# Simple workflow -cancellation



# User Scenarios – account creation

1. User reaches the site by typing <https://app.app-domain.com> and can see the landing page.
2. User clicks on hyperlink and is redirected to /register page. User fills the details which are validated\* on the page. Upon completion the user is redirected to /registration\_confirmation \*\*.

\*Password validation is required to cohere to NIST standard <https://pages.nist.gov/800-63-3/sp800-63b.html> - in summary no repeated letters, password needs to have combinations of alphanumeric characters, can’t be repeated character string, or can’t have name / surname.

\*\* at this stage no tokens are sent to the users.

1. User logs in with their credential.

# User Scenarios – simple use

1. User reaches the site by typing <https://app.app-domain.com> and can see the landing page.
2. User types in correct password / email gets redirected to homepage
3. User requests a booking by clicking ‘request a booking’. They are redirected to /request\_booking page. On the page - available, booked with availability and unavailable dates are rendered– each having an adequate colour scheme.
4. User requests a booking with full availability and gets auto approval message and they see their booking on their homepage.
5. User can cancel their booking from their homepage by pressing cancel and is prompted with confirmation. Upon confirmation the event is released and calendar is updated.
6. User presses log out and they get redirected to <https://app.app-domain.com>.
7. User can create a message from /my\_message page that has a hyperlink on the page ribbon. The message will be picked up by Admin (Business owner) of the application.
8. User can log any issue by clicking ‘Submit an issues’ on the ribbon. They are redirected to /submit\_issue page.

# User Scenarios – security

1. User forgets their password. User clicks forgot password hyperlink. User gets redirected to /forgot\_password page. User can request password reset after typing their email and pressing submit button. User receives an email notification with a link to /reset\_password page.

# Admin user Scenarios

1. Admin is set up through script. Admin is a user with a admin flag set to True.
2. Admin logs with their password. They get redirected to Admin Home which contains calendar view with all events, dashboard with pending approval items and messages from users. Admin can reject the booking.
3. Logged issues can be seen by Admin. Logged issues can be viewed and closed. Closed issues are removed from view

# Out of scope on first iteration

1. Users don’t confirm their email they are just sent a login link.
2. Users don’t use their third-party identity providers.

# Intended Tools to be used

1. Flask
2. MySQL
3. Google API
4. React for Front-End / Bootstrap

# Simple Architecture diagram

