Implementation Documentation

for

Book My Room

July 2015
Version 0.3

Prepared By:
Adam Dubicki - V00799637
Anthony Kohan - V00740142
Chrisanda Ann - V00811197
Jose Gordillo - V00773366
Mike Weicker - V00255449
Tim Salomonsson - V00807959
Ushanth Loganathan - V00810681

Change History

Date	Version	Description	Updated By
July 10, 2015	0.1	Initial version	Team Book My Room
July 11, 2015	0.2	Major update to all sections	Team Book My Room
July 12, 2015	0.3	Final Edit for submission	Team Book My Room

Document Approvals

Name	Role	Signature

Table of Contents

1. Major Components

1.1 Views

Name	File	Description
Login Page	login.html	The login page for the user. The user is required to use its Netlink ID and password to login.
Day Selector Page	dayselector.html	The homepage for BookMyRoom. Shows the days when a user can create a booking.
Schedule Page	schedule.html	The page where the user creates their booking. The user selects the booking duration, start time and equipment. A user must be logged in to visit this page.
Profile Page	profile.html	A profile page for the user, showing their name, netlink ID, phone number, email and current bookings. A user must be logged in to access their profile.
Error Page	error.html	An error page in case of a scheduling conflict.
Extend page	edit.html	A page which a user can extend or delete their booking. Users can only extend their bookings if they are within 2 hours of its start time. A user can only access the extend page if they are logged in.

1.2 Controllers

Name	File	Description
Booking Creator Controller	bookerCtrl.js	This controller is used to create a booking based on the information selected by the user. Based on the user's selection, the controller will find an available room and equipment (if equipment was added) using the Schedule service. Finally, the controller will use the Booking service to create a booking in the database.
Day Selector Controller	bookerCtrl.js	This controller is in charge of calculating which days the user can pick to create bookings. It is also in charge of verifying that the user is not locked out when trying to book a room. Finally, it saves the date the user selected.
Schedule Controller	bookerCtrl.js	This controller is in charge of dynamically calculating which time slots are available based on the user's selection of time and equipment. It uses the Booking service to grab all the bookings in the selected day.
Booking Edit	editCtrl.js	This controller takes care of extending the length of a

Controller		booking and deleting bookings. When a user deletes a booking, the controller checks to see if the user should be locked out. Then, it deletes the booking using the Booking service. In order to extend a booking, the controller figures out what is the earliest booking that it has a conflict with. That is used to set the maximum amount of time to extend.
Login Controller	loginCtrl.js	This controller is in charge of the user's login and logout. It uses the Auth Service to validate the user. It also uses the Auth Service to handle the user's logout.

1.3 Models

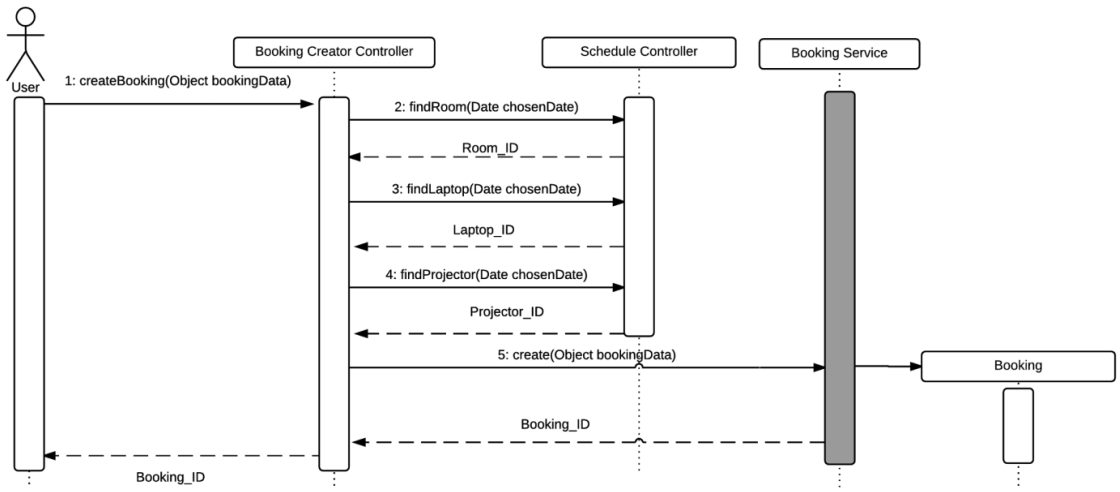
Name	File	Description
Booking	booking.js	The booking schema that is stored in the database. It contains the relevant information about each booking: - booking id - netlink id - room id - projector id - laptop id - start date (year,month,day,hour,minute) - end date (year,month, day, hour,minute)
User	user.js	The user schema that is stored in the database. It contains relevant information about each user: - name - last name - user type - username - password - email - phone number - lockout date It also contains methods for encrypting and hashing the password.

1.4 Services

Name	File	Description
Authentication Service	authService.js	This service has methods for validating login, handling logout, verifying if the user is currently logged in, and grab the user's session information (from the token).
Booking Service	bookingService.js	This service has several different functionalities. It can fetch all the bookings ever made from the database. It can get from the database all the bookings made in a specific day. It can create a booking in the database. It can delete a booking in the database. It can grab all the bookings created a specific user. Finally, it can get all the bookings for a specific room, projector, or laptop.
Schedule Service	scheduleService.js	This service builds the schedule of available rooms, projectors, and laptops based on all the bookings made for a specific day.
User Service	userService.js	This service has several different functionalities. It can get a single user using the object id. It can get all the user. It can create a user in the database. It can update a user in the database. It can delete a user in the database. Finally, it can lockout a user by updating the lockout field in the database.

2. Sequence Diagrams

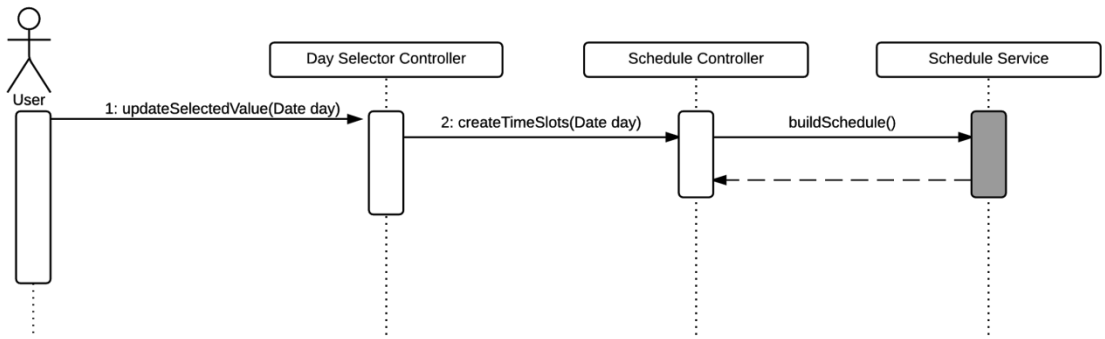
2.1 Create Booking



Methods

createBooking(Object bookingData)	This method grabs the date chosen by the user, the selected start time, the selected duration, and the selected equipment from the Schedule view. It also calculates the end time based on the user's selected duration. It also calls the code to find the projector, laptop, and room IDs. Finally, it grabs all the required information and calls the Booking.create() service.
findRoom(Date chosenDate)	This method grabs the ID of a room that is available during the time the user selected. Then, it adds that ID to the bookingData object.
findLaptop(Date chosenDate)	This method grabs the ID of a laptop that is available during the time the user selected. Then, it adds that ID to the bookingData object.
findProjector(Date chosenDate)	This method grabs the ID of a projector that is available during the time the user selected. Then, it adds that ID to the bookingData object.
create(Object bookingData)	This method is a method in the Booking Service. It is used to create a booking document in the database. The fields of the document are populated using the information from the bookingData object.

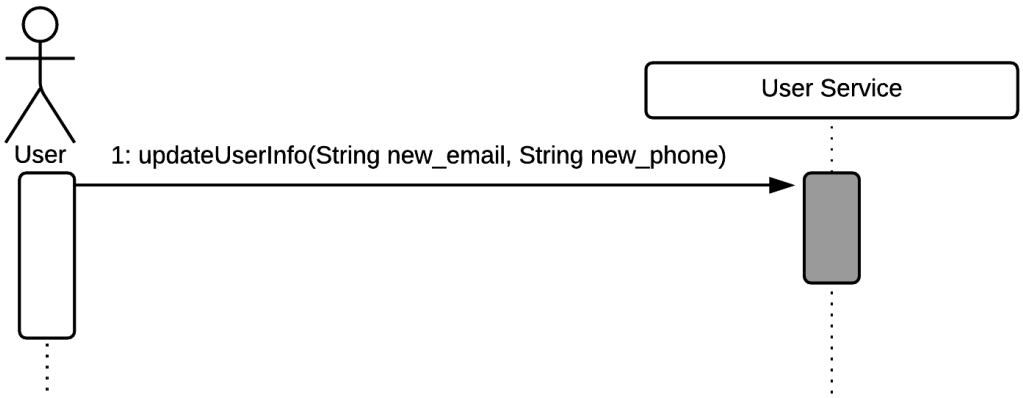
2.2 View Schedule



Methods

updateSelectedValue(Date day)	This method grabs the date that was chosen by the user from the view. If the user is not locked out, this method leads the user to the schedule view.
createTimeSlots(Date day)	This method creates a schedule for the chosen day based on the already existing bookings. It creates the time slots based on the day of the week. Then, it calls the Schedule.buildSchedule() service.
buildSchedule()	This method is in the Schedule Service. It is used to calculate the available time slots based on how long the booking will be, if the user adds equipment to the booking, and which rooms and equipment are available. Then, it sets the changes in the time slot objects in the controller.

2.3 Edit Profile

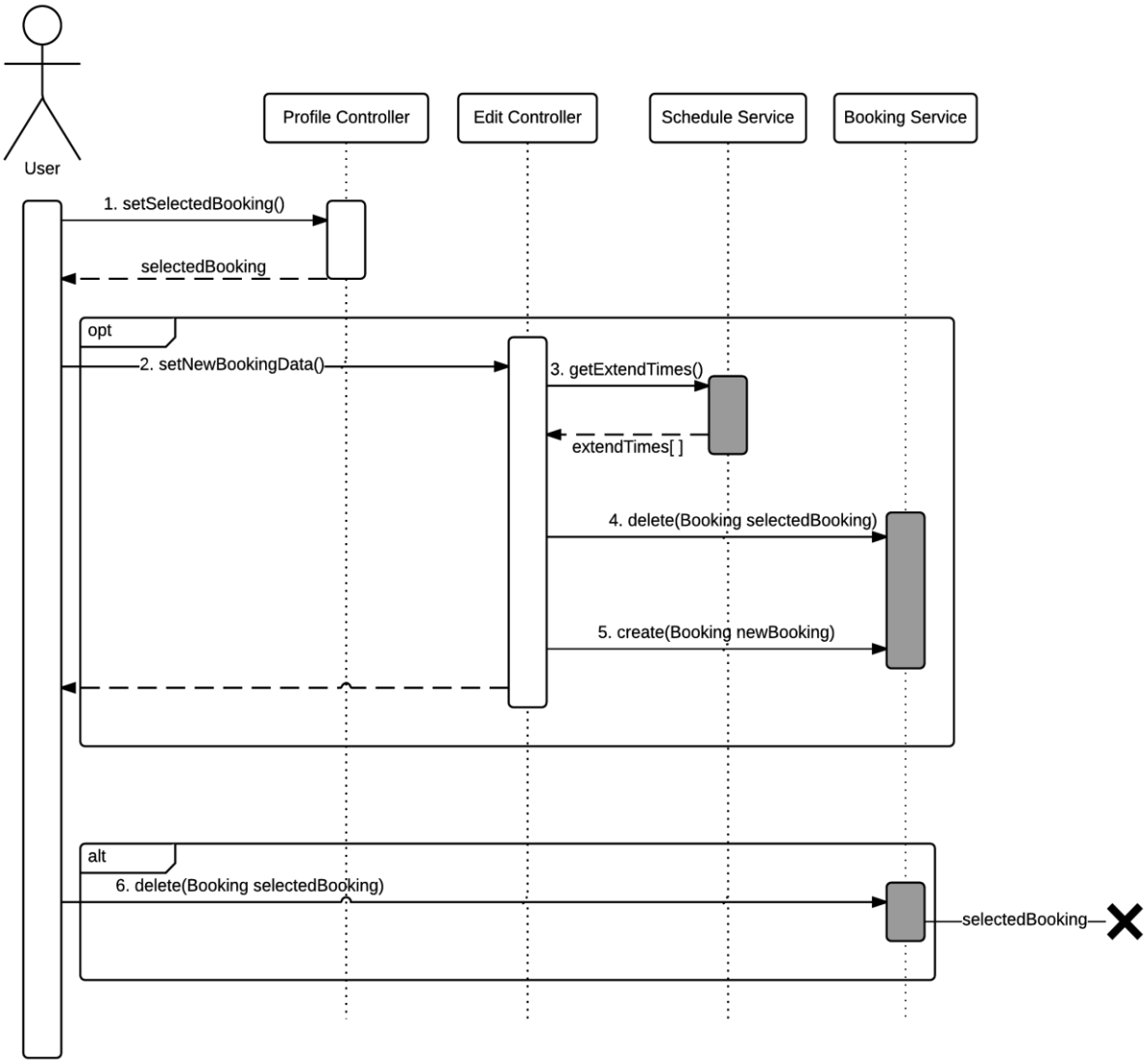


Methods

updateUserInfo(String new_email, String new_phone)	This method is in the User Service. This service looks up the user in the database and updates its email and phone fields.

2.4 Delete Booking/Extend Booking

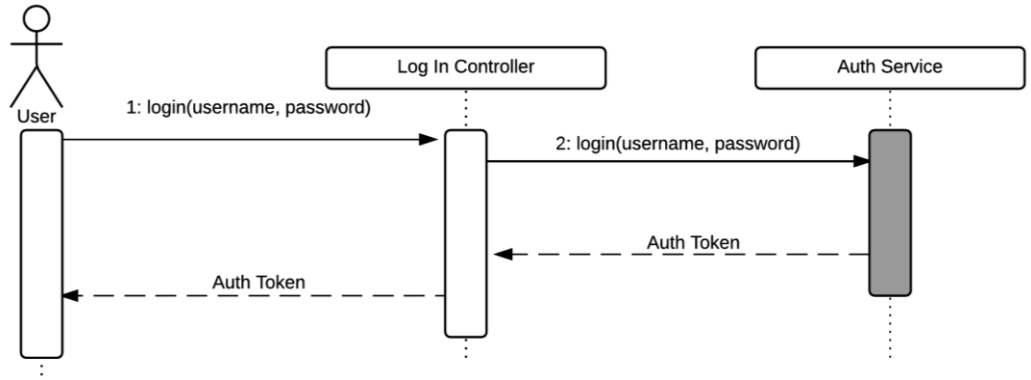
Diagram



Methods

setSelectedBooking()	This method grabs the booking that was selected by the user in the Profile view.
setNewBookingData()	This method adds the new booking information to a booking object.
getExtendTimes()	Grabs the possible extending times based on the type of user and the availability of the equipment and the room.
delete(Booking selectedBooking)	This method deletes the old booking using the Booking service.
create(Booking newBooking)	This method creates a new booking object using the Booking service.

2.5 Login



Methods

login(username, password)	This is a method in the Login Controller which is called by the login.html page. It is passed the username and password which is input by the user. This method then passes the information to the Auth Service.
login(username, password)	The login method in the Auth Service takes in the username and password and checks it against the database. If the user is successfully authenticated then an authentication token is handed to the login controller.

3. Design Patterns

This project uses two design patterns: facade and singleton. The services work as facades that interact with servers [1]. The services simplify the code by separating server interaction and error handling from the model. This simplifies data fetching. Also, each service is also a singleton [2]. Each component dependent on a service gets a reference to the single instance generated by the service factory.

Sources:

[1] http://trochette.github.io/Angular-Design-Patterns-Best-Practices/#/facade
[2] https://docs.angularjs.org/guide/services

4. Test Plans

Refer to SRS for full details of use cases. Refer to Readme.txt in repo for detailed instructions.
If you want to connect to our mongoLab database our login info is, username: bookmyroom, password: hellohello123.

4.1 Use Case 1: View Schedule

Input	Expected Output
Go to localhost:8080	The user will be redirected to the login screen if they are not logged in.
Login with valid login credentials. See readme for more info. Username: 'Student', 'Staff', 'Faculty' or 'Admin' Password for all accounts are: f	User should be taken back to home screen logged in as account _____
	The home screen for 'Book My Room' should display. There should be a button for the next 7 days. Today's date should be below the top navbar.
Click a day	User should be taken to the login screen
Click a day	User should be taken to the schedule for that day. Equipment should be not be selected by default. Days that are full with bookings should be disabled.

4.2 Use Case 2: Log in and Log out

(This use case assumes the user is not logged in)

Input	Expected Output
Click the 'Login' button on the top right of the navbar.	The user will be taken to the 'Login' page.
Enter login credentials. Username: Student Password: f Try first with only the username, no password.	Without the password, the login should fail. Once the password is given with the username, you will get logged into the account bunny, and taken back to the home page. The 'Login' button should turn into a 'Logout' button.
Click 'Logout'.	The user should be logged out of the account Student and redirected to the home page. 'Logout' button should become a 'Login' button. Clicking either the 'Login' button or any of the days on the schedule will prompt the user to login like the first 2 steps.

4.3 Use Case 3: Create booking

(This use case assumes that the user is already logged in).

Click a day on the home page.	User should be taken to a scheduler screen ● The date at the top should correspond to the date chosen on the home screen. ● The default duration should be 0.5 hours with no equipment selected. ● A list of times should populate the screen in ½ hour time slots. ● The available times are ○ Monday - Friday 8:00am to 9:30 pm, 11:00 am to 5:30pm on weekends. Repeat this step for all the days of the week to verify that the schedule is correctly generated.
Select the booking duration and desired equipment. Select a desired start time. Click confirm.	The selected time must have black text. The confirm text should give the correct booking info.
Confirm that the booking information is correct. Click 'ok' on the Booking info popup.	The user should be taken to their profile. The new booking should display under 'My Bookings'. It should contain the correct information.

4.4 Use Case 4: Edit Room Booking

(This use case assumes that the user is logged in and has several bookings).

Input	Expected Output
Click on the user profile icon on the in the top right corner.	The user should be taken to the profile screen.
Click on the 'edit' icon.	User should be taken to the 'edit' booking screen for the selected booking. All information about the booking should match the selected booking.
Extend the time of the booking by 1 hour: Click 'Save Changes' and confirm the booking.	The popup contain all the updated booking information. Once clicked is confirmed, the user should be taken to the profile. The updated booking should appear under 'My Bookings'. A user can not modify their selected equipment. A user can not extend the booking end time unless there is less than two hours before the start time of the booking.
Pick another booking to edit. Click 'Delete Booking'. Confirm the deletion.	You should get a popup warning you about the deletion. Once the booking is deleted, you should be taken back to the profile. The booking should no longer appear under 'My Bookings'. *If the booking is deleted within 5 hours of its start time, the user will be 'locked out' for 24 hours (they will be unable to create bookings). Test this case as well.

4.5 Use Case 5: Edit Profile

(This use case assumes that the user is logged in).

Input	Expected Output
Click on the user profile icon on the in the top right corner.	The profile should display the… - Username and status (not editable) - Name (not editable) - Phone number (editable) - Email (editable)
Click the phone number and email fields and enter new values. Click 'Save Changes'	The profile should refresh and contain the updated information in the editable fields.

5. Read me

Install and run the BookMyRoom application:

1. Download and unzip project folder from https://github.com/SENG299/BookMyRoom
2. Open a terminal inside the BookMyRoom folder and enter 'npm install --save'
3. After the packages install, enter 'node server.js'
4. Go to either Chrome or Firefox and enter Private Browsing
5. Visit http://localhost:8080

Note:
May require Private Browsing when testing to automatically remove the previous browser cache, cookies, etc.

Users for testing/use cases:

- Username: Staff Password: f User type: Staff
- Username: Faculty Password: f User type: Faculty
- Username: Admin Password: f User type: Administrator
- Username: Student Password: f User type: Student

If you want to connect to our mongoLab database our login info is
username: bookmyroom
password: hellohello123.