Iteration0

Brandon Koepke, Alberto Saavedra, Garrick Van Der Lee, Julia Paredes, Tyler Gibb, Justin Milanovic

Abstract

The following document describes the proposed Hotel Manager 4000 system including the intended goal of the project, a tentative release plan, the system requirements, critical success factors and the key development technologies to be used in the system's development.

Contents

1. Vision Statement

Hotel Manager 4000 has a single goal: developing the customer base. HM4000 provides a dual benefit of helping hotels maintain a mounting database of customers with their vacation preferences, in addition to helping hotel staff perform essential daily tasks such as room management, reservations, room service tracking, invoicing and generating detailed sales reports quickly and easily with minimal training. The overall mission is to create apparent value for the hotel by studying clients behavior in order to fashion a list of their future desires and making hotel staff daily duties as simple as possible. HM4000 can provide added value incentives, tied directly to sales, to increase purchases among clients. Hospitality management is about growing the relationships with hotel guests and HM4000 lets your hotel staff spend more time pleasing clients than learning software.

Hotel manager will attempt to provide an efficient and easy to use web interface that will be used by your staff and your customers. We are aware that the staff of your hotel will be busy on certain periods of the year, so that it is mandatory to provide them with a tool that will make them more efficient, and their job easier. These tasks will include basic hotel management features like keeping records, manage customers, and so on. Similarly, we will provide to your customers an easy option for searching and booking rooms.

2. Release Plan

The purpose of this release plan is to determine and prioritize the set of features that the client desires the system to have, as well as to show the commitment of the team members to each of the project tasks. In this release plane, three main iterations have been identified in total for a single and final product release by April 9th, 2013. The fourth iteration has been set aside as a buffer iteration, that means, that if the workload of an iteration is unbalanced, tasks from iteration 4 will be suggested. These iterations will be approximately three weeks long, and will entail the implementation of different system components.

Iteration 1 encompasses the set of user stories that deal with the bootstrap of project, the partial setup of a User Control Management component and the implementation of the Customer Information Management module.

This iteration will look into the implementation of a user control/access mechanism provided by a Ruby plugin for Registered Customers, Sales Associates, Hotel Managers, and the System Administrator.

Furthermore, at the end of this iteration, both the Hotel Manager and the Sales Associates should be able to access the system and perform create, read, update and delete (CRUD) operations on Customers information. The Admin user type will only be able to maintain system accounts.

Iteration 2 includes the implementation of the following modules: Hotel Management, Room Management, and the Room Bookings.

In this iteration, the Admin users should be able to perform CRUD operations on Hotels, doing this will allow any user to search for different hotels. The Room Management module, which consists on the CRUD operations on Rooms and Room Types, will be available in this iteration to Hotel Managers, doing this will allow the Sales Associates to see a list of rooms as well.

At the end of this iteration, Sales Associates will be able to book rooms for Customers in the Room Booking module.

Iteration 3 entails the following modules: Invoicing, Room Service, and Customer self-service.

In the Invoicing module, Sales Associates will be able to record and lookup for customer payments. These payments will also be part of the room service billing history of the customers.

Finally, iteration 3 aims at having Customers to be able to view available rooms, and manage their Room Bookings by themselves. This will require the prior registration of customers to the system in order to perform these tasks. This iteration will conclude the agreed-upon user stories for the project; however, iteration 4 was designed to include some nice-to-have features that could be added to the system if time permits it.

Iteration 4 which is the buffer iteration will implement a basic Trouble Ticket Management module to address customers inquiries, a Housekeeping module to see the cleanliness of rooms, and a Reporting module for business decisions.

It is important to mention that this release plan is tentative, since the team velocity achieved at each iteration will dictate the amount of tasks to commit to in future iterations.

3. Requirements

3.1. Client Requirements

Any operating system capable of running a browser from figure 1 will be supported. This includes Windows Vista and later, Mac OS X 10.5 and later and most Linux variants.

Browser	**Minimum Version**	**Layout Engine**
Firefox	18	Gecko
Google Chrome	24	WebKit
Safari	5	Webkit
Internet Explorer	9	Trident

Table 1: Supported Browsers

These web browsers were chosen because it enables us to support 93.34% of non-mobile web browsers as of October 2012 [2].

3.2. Server Requirements

The server is officially supported on Heroku. The system can be installed on other Linux and UNIX systems, but the installation will not be officially supported. Table 2 lists the necessary packages and their versions.

Package	**Minimum Version**	**Description**	**Rationale**
Ruby	1.9.3	Programming Language	Ruby is a dependency of the Ruby on Rails framework.
RubyGems	1.8.24	Ruby Package Manager	RubyGems eases the installation of third-party ruby libraries. Using ruby gems, you can install the latest version of a library without having to use the system package manager.
Ruby on Rails	3.2	Full-Stack Web Application Framework	Most of the group is writing their paper assignment on Ruby on Rails and we currently only have one group member that has previous experience with Ruby on Rails. In order to write a well-reasoned paper, the group decided that Ruby on Rails should also be used for the supplier assignment in order to gain experience with the framework. Ruby on Rails also favours convention over configuration which makes the creation of simple create, read, update, and delete (CRUD) apps easy.
Postgresql	8.4.13	Object-Relational SQL Database	Postgresql is the default database supported by Heroku. Many modern Linux distributions are also leaving MySql as their default database due to the recent Oracle acquisition of Sun (who previously owned MySql) [1].

Table 2: Required Software

4. Critical Success Factors

The hotel management system must be easy to use, extendible, and scalable. These are non functional qualities that will aid users to have a better experience with the system, and developers to be able to easily extend, improve, adapt, and add features to the system without breaking its main functionality. We will also like to make the webpage scalable so that the hotel can continue to serve their users better at each stage of their growing process as a company. By allowing the hotel management website to be able to grow together with the company, we are also allowing the hotel to take care of their old and new customers so that they can enjoy of all the amenities and services that the hotel offers in the present and in the future.

The hotel manager website must be easy to use for all the users of this service. This will initially include both staff and the hotel customers. We plan on making the as simple and intuitive as possible so employees will need very little training. The website should also improve employee productivity and make it easy to do their jobs. We also want to make it easy for customers to find and book rooms. This will be achieved by focusing early on users and tasks. We will also try to leverage the current experience that staff and customers have and will try to improve upon it.

In order to be successful the system will also need to be very flexible and extensible. This is a critical success factor because we will be adding many features during each iteration of the project and we will have to make sure that adding other features is simple and easy. One such future feature will enable customers to anonymously browse the site without an account. Extensibility is also important because we may have not thought about every single use case and will have to modify the system in the future to support those new use cases.

5. Key Development Technologies

We have decided to implement the webpage using Ruby on Rails framework. Ruby on rails is an open source web application framework for Ruby. We have chosen to use this framework because it allows developers to code features fast. Moreover, some members of our group decided to work on the topic of Ruby of Rails development for their paper assignment, so that we all decided that using this framework for this other project would be beneficial to them.

For deployment we have decided to use Heroku, which is a cloud platform as a service. We have decided to use Heroku because it integrates well with Ruby on Rails, and it is easy to deploy projects with just a git command. Furthermore, this is recommended by most people in the rails community. Heroku also offers a Postgres free data plan to start, so we are going to use Postgres to store our data.

Other key development technologies include the integrated development environment, the continuous integration server, the testing framework, and the version control system. Eclipse was chosen as the IDE due to the fact that it is both open source and cross-platform. Eclipse has also been promoted in previous courses, so

the majority of the group is already familiar with it. Unfortunately Eclipse does not have out of the box support for Ruby, so the plugin Aptana is being used for Ruby support.

Jenkins is being used as the CI server, and is being hosted here. Jenkins was chosen because it has an extremely large number of plugins and it is easy to extend. It also has support for many of the common version control systems including git. We also looked at CruiseControl.rb, but chose not to use it because it does not have a graphical reporting system. Git is being used as the version control system, and Trac is being used as the web interface to git. The git repository can be checked out using:

```
git clone <username>@seng403.ssh22.net:/var/cache/git/hotelmanager
```

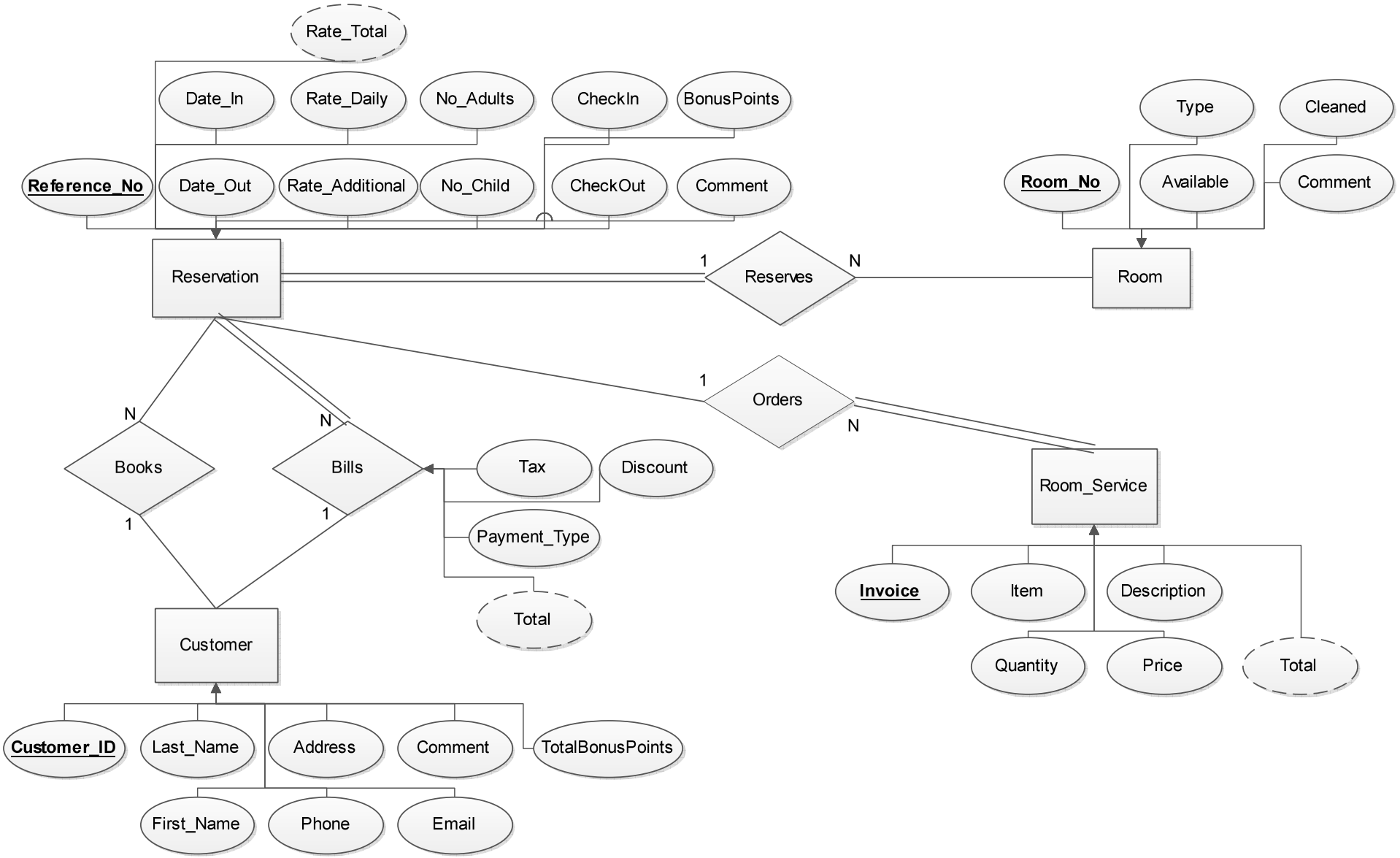
The trac repository can be accessed here. We also tried the Gitlab web interface, but it was missing timeline and roadmap functionality and its bug reporting functionality was poor.

Shoulda will be used as the unit testing framework. This is because it is an extension of the built in Ruby unit testing framework Test::Unit. Unlike RSpec which encourages Behaviour Driven Development, Shoulda uses Test Driven Development which the group is more familiar with.

6. Contributions

Name	Description
Alberto	Helped on Release plan Wrote the Release Plan section
Brandon	Evaluated and setup continuous integration server, git, and trac Wrote the key development technologies section Re-wrote the release plan table and operational requirements Combined the report and produced the rough-draft
Garrick	Helped on Release plan Chose the IDE plugin Wrote the Release Plan Section
Julia	Wrote the critical success factors Chose the IDE plugin Updated technologies section, adding more information Re-wrote the release plan table
Justin	Created ERD Diagram Tested trial versions of various hotel management systems to build the database Wrote the Vision Statement Created the Relational Database Schema (not included in report)
Tyler	Wrote operational requirements

Iteration	User Stories	Tasks	Difficulty/ Estimate	Status	Owner	Description
Iteration 0 Deadline Feb	Development Setup	system	Hard	Complete	Brandon	
		environment	Medium		Everyone	Install Aptana, Ruby, Rails.
Iteration 1 Deadline Feb 26	Administrator is able to maintain users	control	Hard			base user access control.
		administrator	Medium			specific behaviour.
		users	Medium			modify user accounts.
		User permissions	Hard			of other users and roles.
	Users are able to login to the system.	User control for sales associates	Easy			
		User control for hotel managers	Easy			
		User control for registered customers	Easy			
	Customer information management	Sales associates can manage customers	Hard			Sales associates will be able to add, delete, and modify customers.
		search for specific customers	Hard			Sales associates will be able to do a very simple search of the customers by the customer name.
Iteration 2 Deadline Mar 19	Hotel management	Administrators can manage hotels	Hard			Administrators will be able to add, delete, and modify hotels.
		hotels	Hard			Users will be able to search hotels
	Room management	Hotel managers can manage rooms	Medium			Hotel managers will be able to add, delete, and modify rooms.
		Hotel managers can manage room types	Easy			Hotel managers will be able to add, delete, and modify room types.
		Sales associates can view the list of rooms	Medium			Sales associates will be able to do a very simple search of the rooms.
	Room booking and management	manage room bookings	Hard			Sales associates will be able to add, delete, and modify room bookings.
		Sales associates can view bookings	Medium			Sales associates will be able to search for room bookings.
Iteration 3 Deadline Apr 9	Invoicing	record customer payments	Hard			Sales associates will be able to record customer payments and manage invoicing status.
		view customer billing history	Medium			Sales associates will be able to view customer billing history.
	Room service	service	Easy			
		view customers by room number	Medium			Room service associates will be able to view customers by room number.
		manage the room service bill	Medium			and modify the room service bill for customers at their hotel.
	Customer self-service	New customers can register to the system	Hard			Customers should be able to register with the system if they do not already have an account.
		customers	Easy			
		Customers can view available rooms	Easy			Registered customers will be able to view the available rooms.
		manage their room bookings	Easy			remove reservations, and view their current reservations and payment history.
Iteration 4 Deadline May 1	Trouble ticket management	Customers can issue trouble tickets	Hard			Customers should be able to issue trouble tickets which can be managed by hotel workers.
		Sales associates can resolve trouble tickets	Hard			trouble tickets, including responding and resolving issues.
	Housekeeping	housekeeping	Easy			
		manage cleanliness status	Hard			cleanliness status and mark rooms as cleaned, as well as view the rooms they are responsible for.
	Reporting	Hotel managers can produce reports on accounts receivable, hotel utilization, etc.	Hard			Hotel managers should be able to view and manage reports on the hotel's accounts receivable, view the hotel utilization, view the average rate where customers do not show up, view and suspend delinquent accounts, etc.


Rate_Total
Date_In
Rate_Daily
No_Adults
CheckIn
BonusPoints
Reference_No
Date_Out
Rate_Additional
No_Child
CheckOut
Comment
Reservation
1
Reserves
N
Room
Type
Cleaned
Room_No
Available
Comment
1
Orders
N
N
Books
N
Bills
1
1
Tax
Discount
Payment_Type
Total
Customer
Room_Service
Invoice
Item
Description
Quantity
Price
Total
Customer_ID
Last_Name
Address
Comment
TotalBonusPoints
First_Name
Phone
Email

7. References

[1] M. Maslanova. Summary/minutes for fesco 2013-01-30, 2013. [Online; accessed 5-February-2013].

[2] Wikipedia. Wikimedia traffic analysis report - browsers, 2012. [Online; accessed 5-February-2013].