AE1 Report

Software Systems Development – SWD313

Southampton Solent University

BSc Software Engineering

Mark Tomlin – Q09947311

Contents

[Introduction 3](#_Toc420070313)

[Sprint 1 3](#_Toc420070314)

[Adoption to Scrum 3](#_Toc420070315)

[Evaluation 3](#_Toc420070316)

[Sprint 2 4](#_Toc420070317)

[Requirements Elicitation 4](#_Toc420070318)

[Requirements Analysis 4](#_Toc420070319)

[User Stories 4](#_Toc420070320)

[Evaluation 4](#_Toc420070321)

[Sprint 3 4](#_Toc420070322)

[Product Backlog 4](#_Toc420070323)

[Burn-down Chart 5](#_Toc420070324)

[ICONIX Process 5](#_Toc420070325)

[Use Cases and Expanded Use Cases 5](#_Toc420070326)

[Flow Chart, Robustness and Sequence Diagrams 5](#_Toc420070327)

[Evaluation 5](#_Toc420070328)

[Sprint 4 6](#_Toc420070329)

[MVC 6](#_Toc420070330)

[GUI Wireframes 6](#_Toc420070331)

[GitHub collaboration 6](#_Toc420070332)

[SQLite Database 6](#_Toc420070333)

[Implementation Started 6](#_Toc420070334)

[Evaluation 6](#_Toc420070335)

[Sprint 5 7](#_Toc420070336)

[Testing Strategy 7](#_Toc420070337)

[GitHub Problems 7](#_Toc420070338)

[Implementation Features Added 7](#_Toc420070339)

[Implementation Problems with JTable 7](#_Toc420070340)

[Evaluation 7](#_Toc420070341)

[Sprint 6 8](#_Toc420070342)

[Solution to JTables 8](#_Toc420070343)

[GUI Update 8](#_Toc420070344)

[Implementation Features Added 8](#_Toc420070345)

[Effort Matrix 8](#_Toc420070346)

[Evaluation 8](#_Toc420070347)

[References 9](#_Toc420070348)

[Appendices 10](#_Toc420070349)

[Appendix A – Client Requirements Questionnaire with Response 10](#_Toc420070350)

[Appendix B – Client Questionnaire Analysis 11](#_Toc420070351)

[Appendix C – User Stories 13](#_Toc420070352)

[Appendix D – Product Backlog 14](#_Toc420070353)

[Appendix E – Burn-down Chart 15](#_Toc420070354)

[Appendix F - Use Case Diagrams 17](#_Toc420070355)

[Appendix G – Expanded Use Case Diagrams 18](#_Toc420070356)

[Appendix H – Flow Diagrams 20](#_Toc420070357)

[Appendix I – Robustness Diagrams 22](#_Toc420070358)

[Appendix J – Sequence Diagrams 23](#_Toc420070359)

[Appendix K – GUI Wireframe Designs 25](#_Toc420070360)

[Appendix L – Effort Matrix 27](#_Toc420070361)

AE1 Report

# Introduction

This report will detail all the decisions and work done by the team, during the project’s entirety. The project was to create a Comedy Venue system for event bookings, customer bookings and feedback. Decisions made by the team will be justified and any work pieces done will be mentioned and attached into the appendices.

# Sprint 1

## Adoption to Scrum

As part of the requirements for this group project, SCRUM was introduced and used as a project management methodology. The use of Agile practises and management was familiar to the team, however on one was experienced in using SCRUM. As mentioned by M. Cohn “adopting Scrum pervade everything development team members do, but also many changes go against much of their past training” (Cohn, Succeeding with Agile: Software Development Using Scrum, 2009). Therefore the team needs to adapt from experience with test-driven development to effectively apply Scrum principals.

The first decision for using Scrum was to plan out the different sprints for the project’s development plan. As stated from a scrum methodology website “work is confined to a regular, repeatable work cycle, known as a sprint… today many teams prefer shorter sprints, such as one-week or two-week sprints” (James, Scrum Methodology, 2011). With this in mind the team decided to have a sprint length as two weeks, with six sprints in total for the project’s timeframe.

Research into Scrum practises and artefacts such as product backlogs, user stories and burn down charts were carried out by the entire team during the rest of this sprint.

## Evaluation

This first sprint mainly consisted of the team researching and understanding the Scrum agile methodology. Key artefacts such as the product backlogs, user stories and burn down were identified during this stage. A sprint plan was created as part of the project planning. Overall this sprint was necessary for the team’s adaption and understanding of Scrum for use in the following sprints. Understanding towards the system requirements is limited at this point.

# Sprint 2

## Requirements Elicitation

The description of the system’s requirements from the assessment brief was lightly detailed, so many assumptions were made to point about the desired software system. In order to understand the system requirements better, a client interview was necessary. As suggested by an agile requirements gathering article regarding client and user interviews “This very common approach to [gathering user stories](http://www.brighthubpm.com/methods-strategies/92832-extreme-programming-iteration-planning/) is certainly more effective” (Kirkham-Jones, 2014).

In order to elicit the most useful information about the system, regarding previous assumptions made. A questionnaire for the client interview was produced to prepare questions for the client to better focus answers necessary for understanding of the system requirements. The produced questionnaire with notes of the client’s responses can be found in appendix A.

## Requirements Analysis

For better understanding of the client’s responses during the interview. An analysis document of the Client interview requirements questionnaire was produced, see appendix B for this document. Also this analysis is used for detailing design considerations, from the new understanding of the requirements.

## User Stories

As stated by M. Cohn “A user story describes functionality that will be valuable to either a user or purchaser of a system or software” (Cohn, User Stories Applied: For Agile Software Development, 2004). User stories are one of main artefacts related to Scrum development. Therefore it necessary for the team to create a user story for each user of system, this includes the administrator, receptionist and customer. The user story artefact can be found in appendix C .User stories were very useful to the team for planning the requirements and functionality of the system. This is explained again by M. Cohn “Each story represents a discrete piece of functionality…This makes user stories appropriate as a planning tool” (Cohn, User Stories Applied: For Agile Software Development, 2004).

## Evaluation

This sprint was more productive towards meeting project goals than the previous, as the requirement elicitation and analysis took place during this second sprint. User Stories as part of the Scrum methodology were also created to good standard and detail for each potential user function required. Elicitation gave some insightful requirements, however further talks from other potential users of the system would be more ideal and help develop the user stories more accurately.

# Sprint 3

## Product Backlog

To expand from the user stories made in the previous sprint, a Product Backlog was made by the team to list the different functions required by the system. These are all referenced straight from the user stories. The list is ordered in terms of priority to overall system functionality. The produced Product Backlog can be found in appendix D.

The Product Backlog is another integral part using Scrum. As stated from a web article “In Scrum, the product backlog is the single most important artefact”. This article goes on to say that “The product backlog is, in essence, an incredibly detailed analysis document, which outlines every requirement for a system” (James, The Scrum Backlog, 2011).

## Burn-down Chart

Another important Scrum artefact is the Burn-down Chart. Creating a Burn-down was done by expanding the Product Backlog to give time estimates and actual completion times. This results in the plotting of a burn-down chart to illustrate the progress of implementing each product feature to the time/date given. The completed Burn-down Chart can be found in appendix E.

A key benefit for using a Burn-down chart for this project was described in a scrum article “The burn-down chart, on the other hand, provides daily feedback on effort and schedule, thereby mitigating the risks and raising alarms as soon as something goes wrong” (Mittal, 2013). The daily feedback of progression with the system features helped the team plan and manage the project during implementation.

## ICONIX Process

In order to successfully transition between the analysis and design phases, the ICONIX process was decided by the team to be used for the project. As stated in the Agile manifesto “Individuals and interactions over processes and tools” (K Beck, 2001) meaning functionality and features should be prioritised. ICONIX is suited to this Agile principle because as explained by D Rosenberg, M Collins-Cope and M Stephens “Of the agile methodologies described here, Feature-Driven Development (FDD) is in many ways closest at heart to ICONIX Process” (Doug Rosenberg, 2005).

## Use Cases and Expanded Use Cases

As part of the ICONIX process, some basic use cases were made for each user that show their wanted interaction/functionality with the system. This is in reference to the user stories and product backlog. To see the use cases, see appendix F.

To further develop and include more details to the use cases, several expanded use cases were made by the team. These show an overview of the feature and its need from the user story. Also shown is a step by step of how the user and system would interact, as well as alternative courses for certain steps where non ideal interactions are detailed (such as what happens when no events are available). To see the team’s expanded use cases, see appendix G.

## Flow Chart, Robustness and Sequence Diagrams

To further proceed with the ICONIX process some more diagrams where made by the team. These diagrams are more design heavy toward the software system.

Firstly some flow diagrams were made by individual of the team to further illustrate the system process pathing. See Appendix H for these.

Robustness diagrams were created and show design detailed interactions with the users and data type functions. These being customer bookings, events and feedback. They also show interaction with the software system and database entities. See appendix I for the robustness diagrams.

Sequence diagrams were produced by developing from the robustness and expanded use cases. To view these see appendix J.

## Evaluation

This third sprint was very productive in developing further analysis and designing the system. The project backlog and burn-down charts were also introduced at the start of this sprint, however they wouldn’t be used till the next sprints, as implementation hasn’t stared yet. For the ICONIX process artifacts, the team believes they were well made and give great insight into development of the design for the system (especially the expanded use cases).

# Sprint 4

## MVC

The team decided to use MVC as the design pattern for this software system. MVC was chosen as a design pattern for the implementation because of it separation of concerns, being the Model View and Controller. This means team members who wish to focus on specific implementation points (such as GUI) can do so with ease. As suggest from a programming resource site “Developer specialization and focus: -The developers of UI can focus exclusively on the UI screens without bogged down with business logic” (Sagar, 2009).

## GUI Wireframes

In relation with MVC and as a basic design tool, a GUI (graphical user interface) wireframe was made to illustrate the design of the interface. These wireframes show a guide to the view of the system from the eyes of a user. To see these GUI wireframes see appendix K.

## GitHub collaboration

Because multiple team members will be working on the implementation together simultaneously. Also because of Agile and Scrum principals with task assignment by the Product Backlog. It was decided to use GitHub as developer collaboration tool for this projects implementation (GitHub Inc., 2015).

## SQLite Database

For the database method for this Java software system. It was decided to use SQLite as the database in order to store the tables necessary for this system. As described on their website “implements a [self-contained](https://www.sqlite.org/selfcontained.html), [serverless](https://www.sqlite.org/serverless.html), [zero-configuration](https://www.sqlite.org/zeroconf.html), [transactional](https://www.sqlite.org/transactional.html) SQL database engine” (SQLite, 2015).

## Implementation Started

Tasks from the Product backlog were self-assigned by different team members. A simple implementation of the GUI (View) was established for testing, this is because from the GUI Wireframes it shows that majority of functionality of the system will be done from the same interface screen. Viewing and adding the event and booking information have been implemented, as part of the first four features in the Project Backlog. These features however are not to finished standard as viewing tables can’t be refreshed after adding entries.

## Evaluation

Several design & implementation choices were made over the course of this sprint. The GUI wireframe were created as necessary design tool for the view of the overall system. Implementation is started at the end of this sprint, tasks were allocated by the project backlog, main features regarding the viewing and adding of bookings and events were implemented. However they are not to a finished standard yet.

# Sprint 5

## Testing Strategy

The testing strategy chosen by the team is firstly driven by the story and product backlog for that feature. This type of testing can be known as ‘story test driven development’. According to Agile Alliance the benefit of using this kind of testing is for “encouraging closer collaboration between developers on the one hand and customers, users or domain experts on the other” (Agile Alliance , 2013).

## GitHub Problems

Upon trying to collaborate work together for different releases, there were many problems faced by each team member when trying to sync and send the Java files through a GitHub plugin on Eclipse. One team member had their code lost when trying to send it to the repository and instead rewrite program code back to the previous release. From the time that it was taken to solve issues with GitHub that time could have been spent developing the software system.

To stop the decline in the Burn-down chart it was decided by the team to stop using GitHub as coloration tool as too much time was being spent learn to use it correctly. Instead team members manually spliced the relevant changes in code together using USB devices.

## Implementation Features Added

The tasks implemented from the product backlog this sprint were the next four items. These consisted of implementing the creation and viewing of customer feedback to a given event. What also was added is the feature to delete bookings and events. These can performed through the use of textboxes with an inputted event ID.

## Implementation Problems with JTable

There were continued problems with being able to refresh the display tables in the current system, when any changes to the databases are done. This is due to nature of the system’s interface being all from one window. A current work around for this problem is to display the tables once needing refresh as a pop-out JDialog window. A solution for this problem wouldn’t be realised till next sprint.

## Evaluation

This sprint involved a lot of implementation and testing with the developing software system. There were some setbacks during testing and implementation however. For displaying the different listings, a workaround was achieve but this isn’t suitable as an actual solution. A fix for this problem was planned to be achieved during the next and last sprint.

# Sprint 6

## Solution to JTables

A solution was found for the displaying the listing tables so they updated during this split. The problem that occur was due to referencing issues when setting a table model to the listing table while in a Java try – catch statement. A code snippet of the solution can be seen below in figure 1.

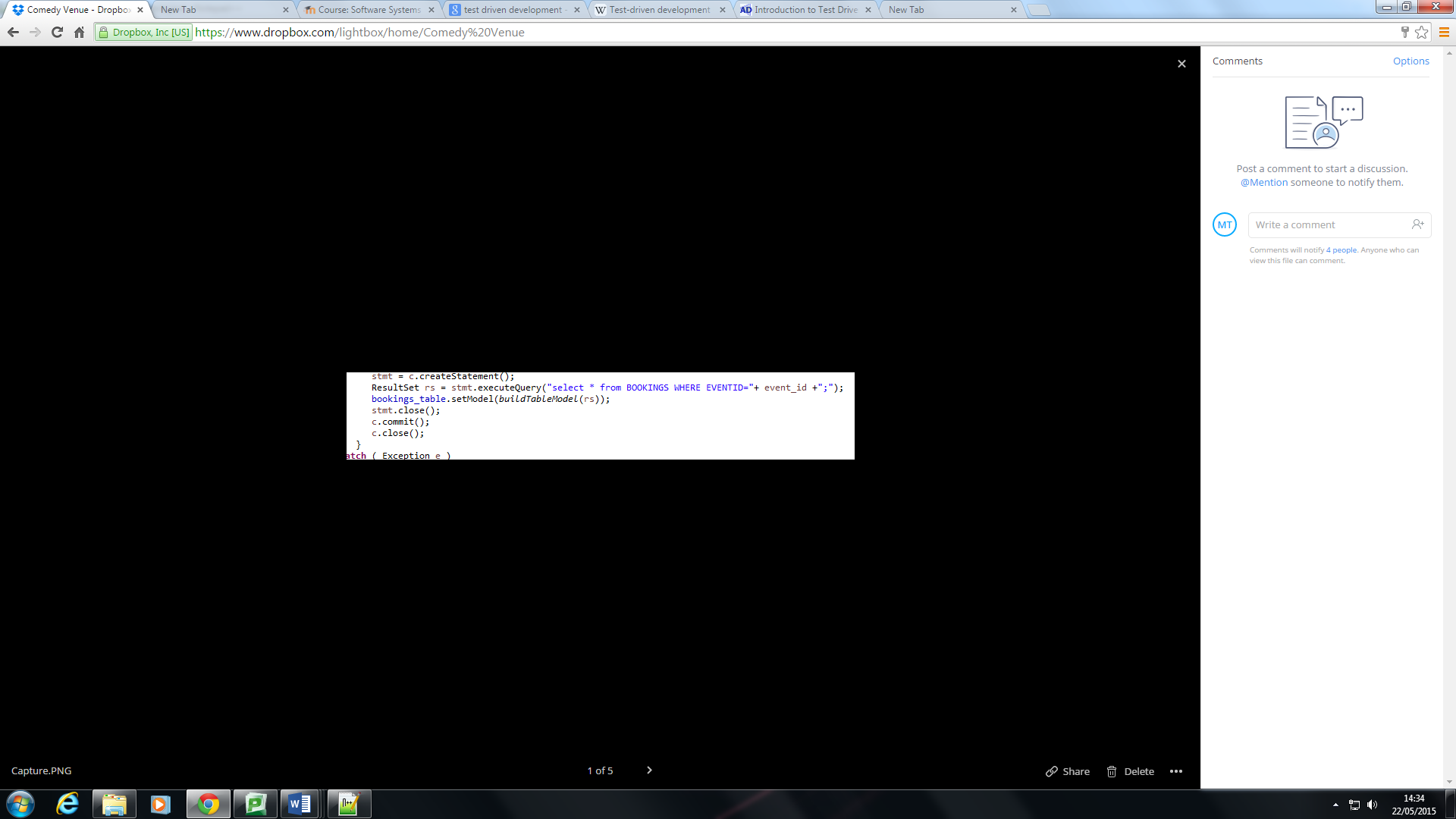


Figure 1- Tabled listing solution

## GUI Update

The GUI was updated to better display the listing tables, inputs and buttons so it matches the design proposed in the GUI wireframes better. Empty spaces were more minimised compared to the wireframe design, this was decided to by team as it increase visibility and aesthetics.

## Implementation Features Added

The final product backlog features were finally implemented correctly. These involved showing the availability of seats to an event and denying any bookings where seats aren’t available. The second part to this sprint was creating the login system for an admin, as well as put access restrictions on editing the Events when not logged as an admin. This complies with access control required for the receptionist as stated in the user stories.

## Effort Matrix

An effort matrix document was produced to show time spent on tasks by which team member. This document also includes time estimates of man hours with the actual recorded hours. To view the effort matrix see appendix L.

## Evaluation

The finished system was subject to unit and story testing to judge if features were implemented correctly. This sprint was the final iteration of the software system and has the correct GUI. Further validation from the client however should have been carried out to further develop system requirements.

# References

Agile Alliance . (2013). *Acceptance Testing*[online][viewed 21 May 2015]. Available from: http://guide.agilealliance.org/guide/acceptance.html

Cohn, M. (2004). *User Stories Applied: For Agile Software Development.* Addison Wesley.

Cohn, M. (2009). *Succeeding with Agile: Software Development Using Scrum.* Addison Wesley.

Doug Rosenberg, M. C.-C. (2005). Agile Development with ICONIX Process: People, Process, and Pragmatism. New York City: Apress.

GitHub Inc. (2015) [online][viewed 21 May 2015]. Available from: GitHub: https://github.com/

James, M. (2011, OCT 10). Scrum Methodology[online][viewed 21 May 2015]. Available from: http://scrummethodology.com/scrum-sprint/

James, M. (2011, September). *The Scrum Backlog*[online][viewed 21 May 2015]. Available from: Scrum Methodology: http://scrummethodology.com/the-scrum-backlog/

K Beck, M. B. (2001). *Manifesto for Agile Software Development*[online][viewed 21 May 2015]. Available from: http://www.agilemanifesto.org/

Kirkham-Jones, M. (2014, September). *Agile Project Requirements Gathering: Techniques to Identify User Stories*[online][viewed 21 May 2015]. Available from: http://www.brighthubpm.com/agile/123103-assembling-a-toolbox-for-agile-requirements/

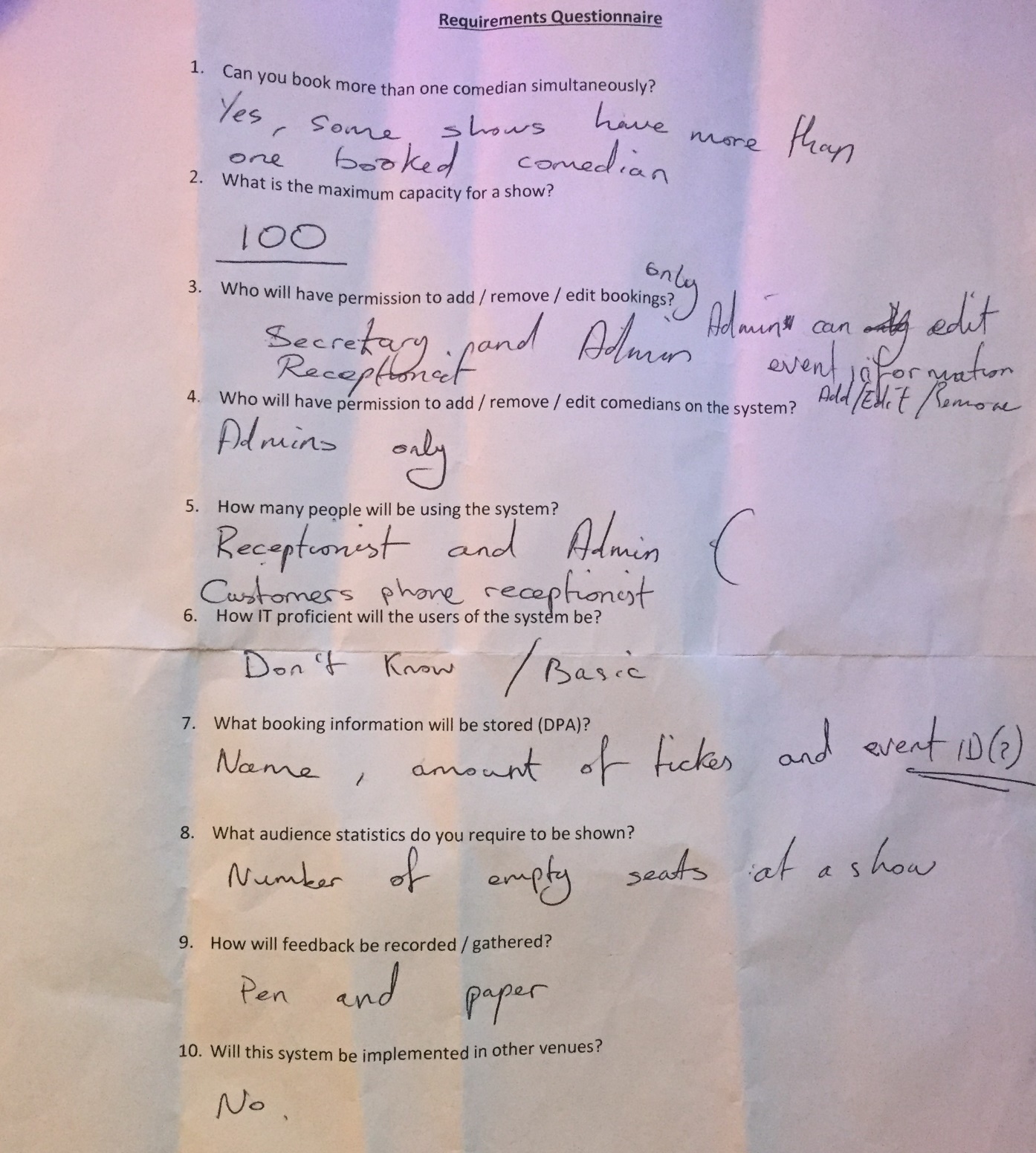
Mittal, N. (2013, August 7). *The Burn-Down Chart: An Effective Planning and Tracking Tool*[online][viewed 21 May 2015]. Available from: Scrum Alliance: https://www.scrumalliance.org/community/articles/2013/august/burn-down-chart-%E2%80%93-an-effective-planning-and-tracki

Sagar, V. (2009, Jan 19). *Benefits of MVC pattern*[online][viewed 21 May 2015]. Available from: CareerRide: http://www.careerride.com/MVC-benefits.aspx

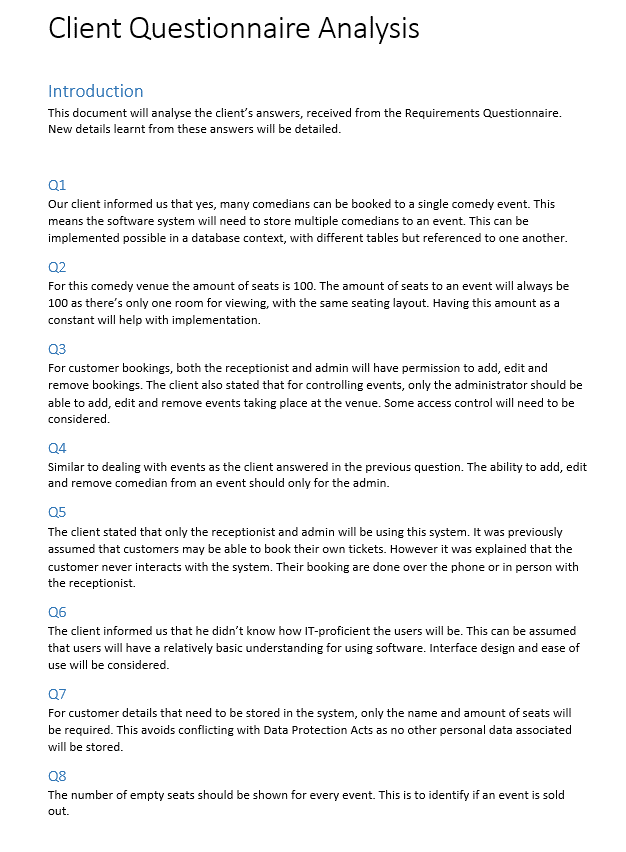
SQLite. (2015, May 20). SQLite[online][viewed 21 May 2015]. Available from: https://www.sqlite.org/

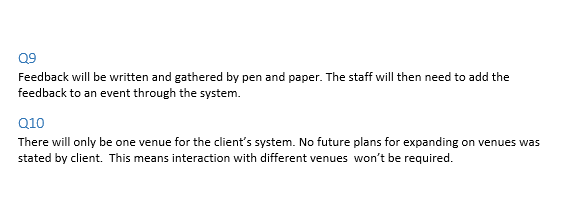
# Appendices

## Appendix A – Client Requirements Questionnaire with Response

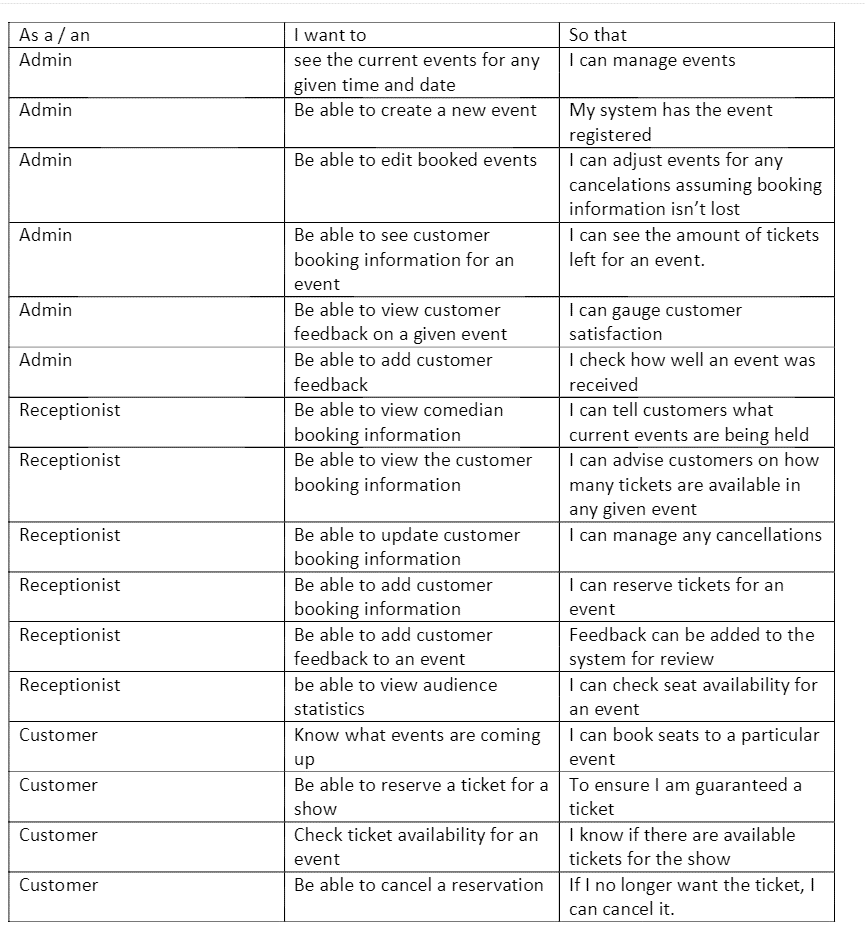


## Appendix B – Client Questionnaire Analysis

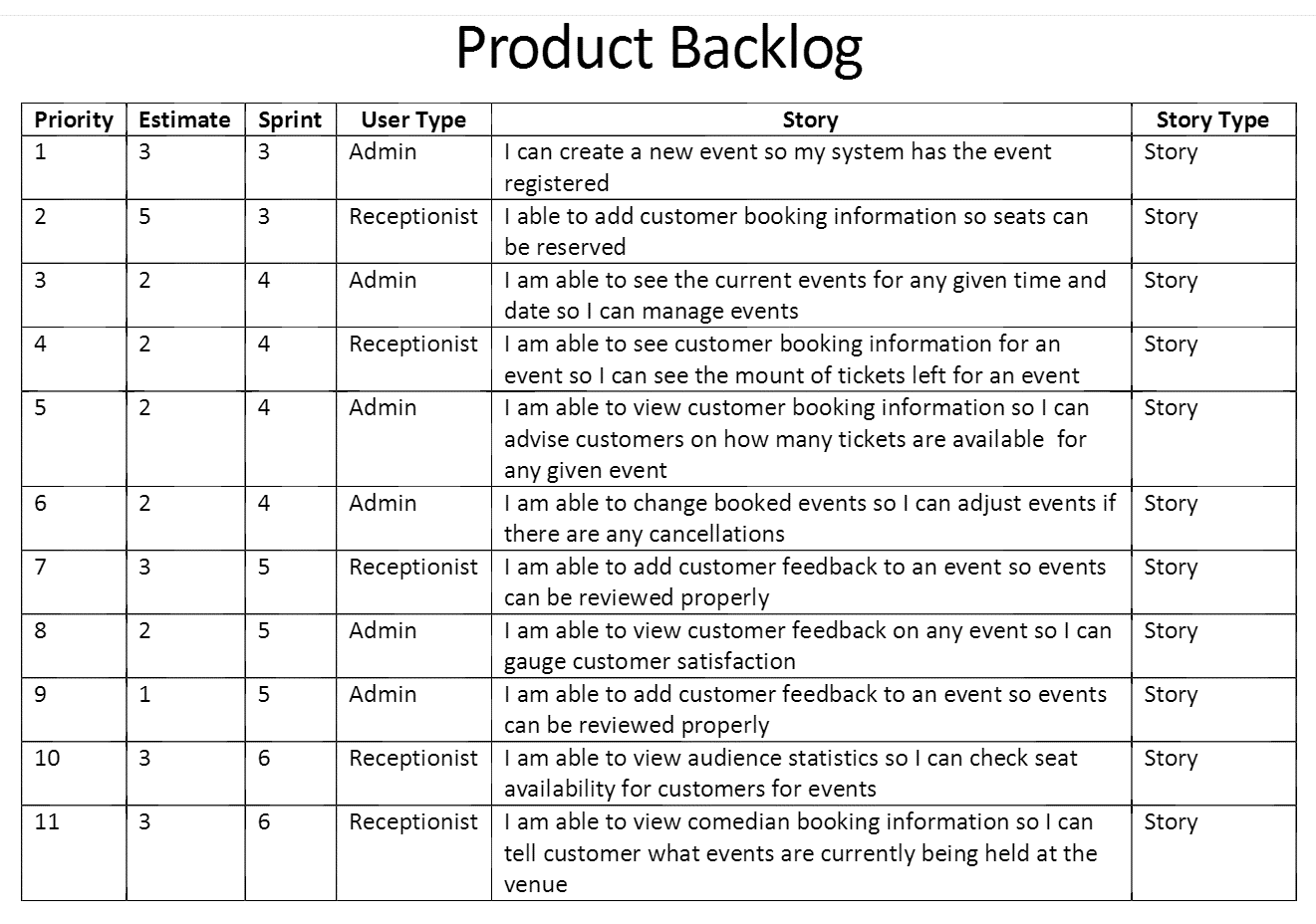




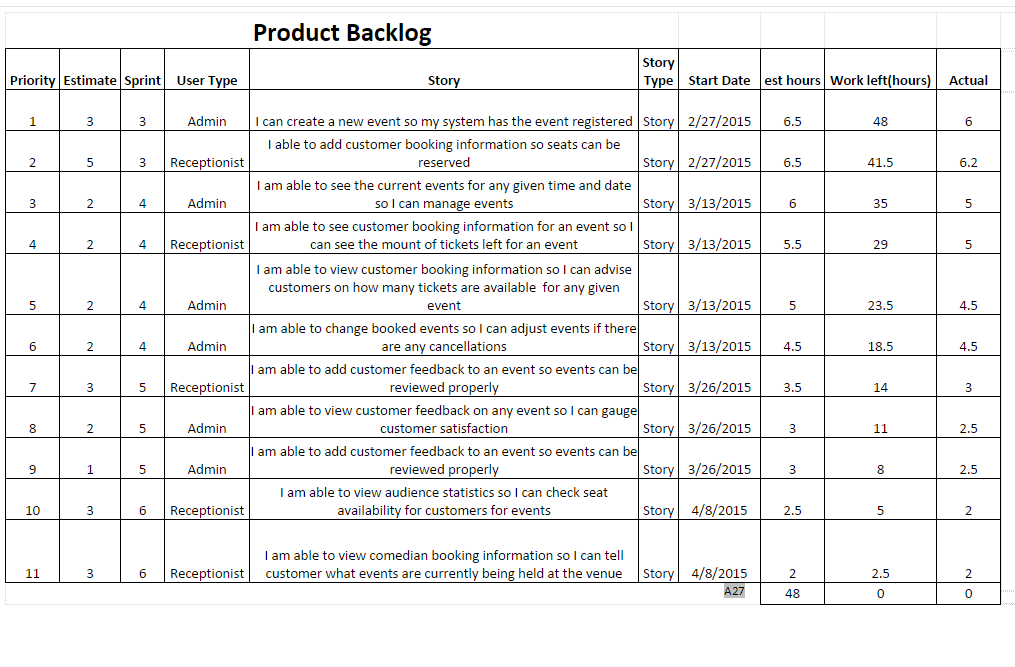
## Appendix C – User Stories

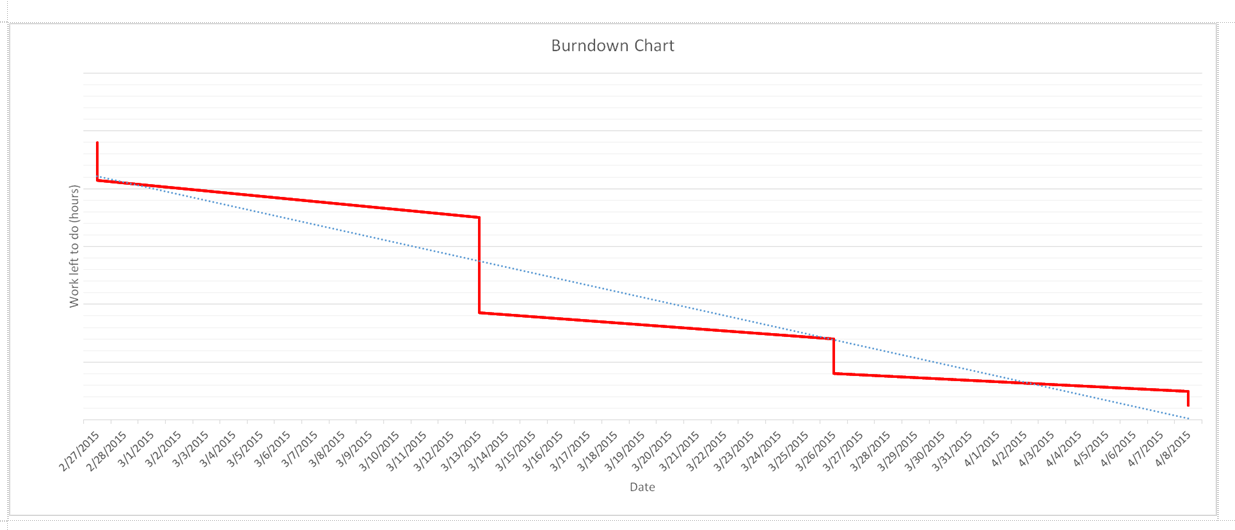


## Appendix D – Product Backlog

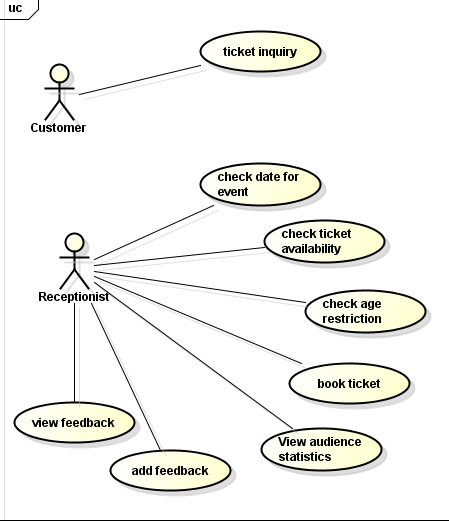


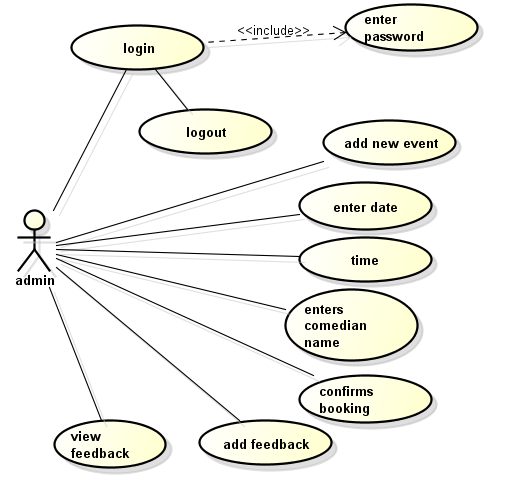
## Appendix E – Burn-down Chart



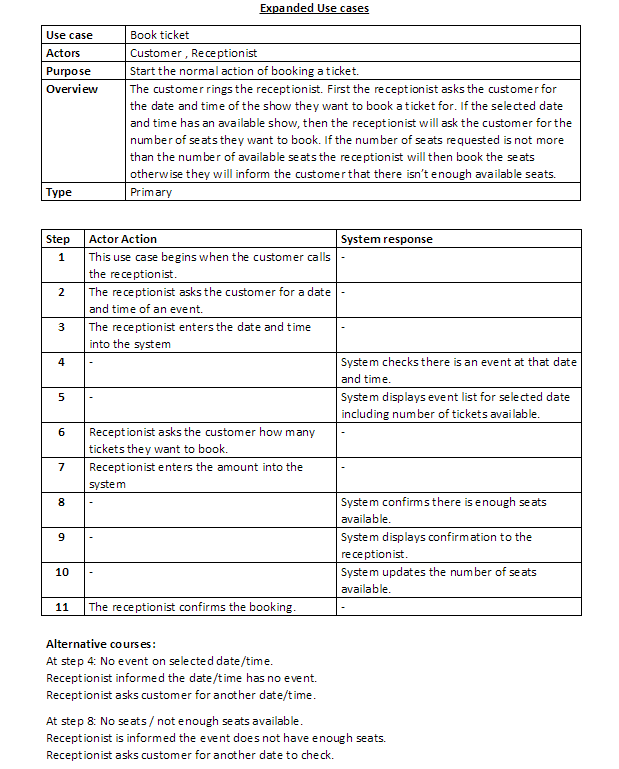


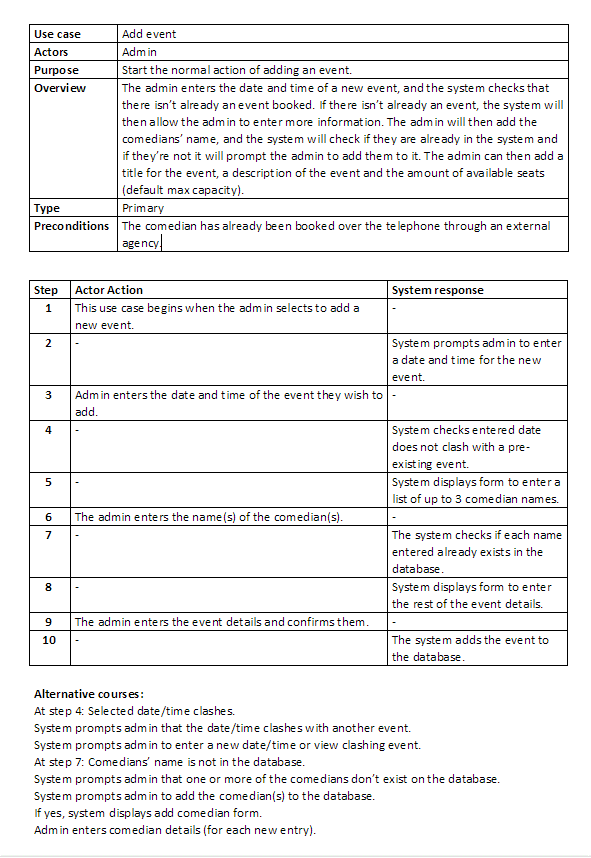
## Appendix F - Use Case Diagrams



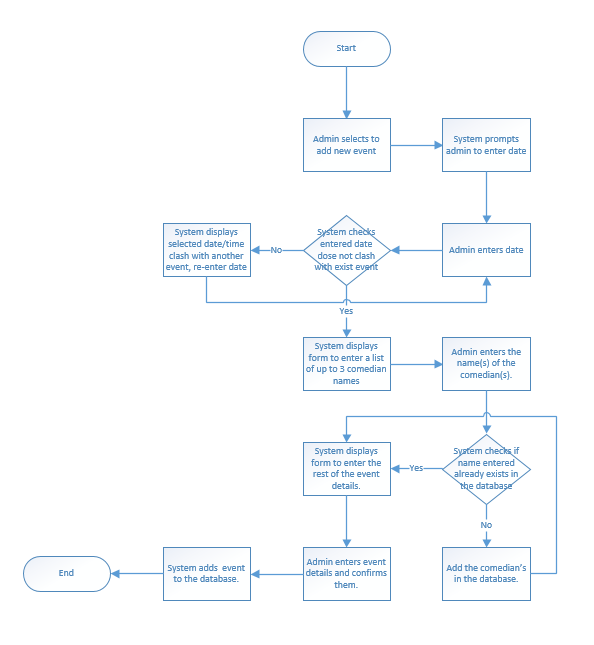


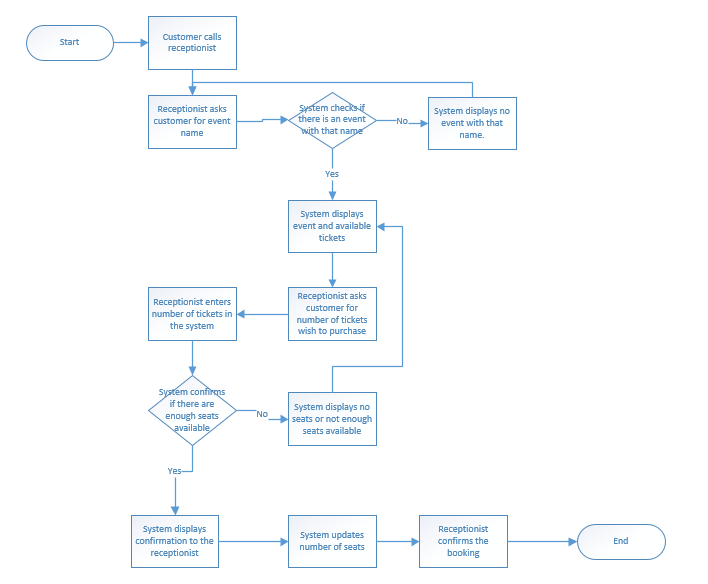
## Appendix G – Expanded Use Case Diagrams



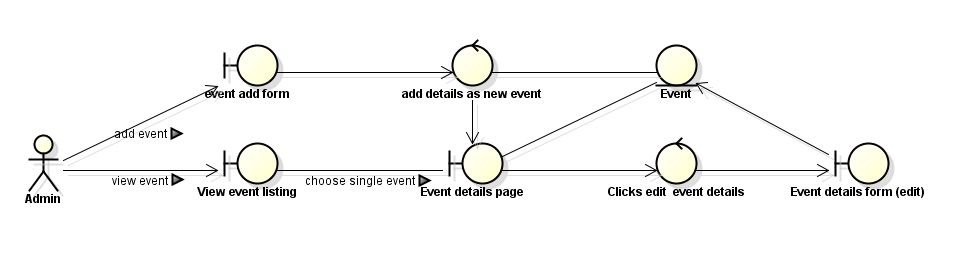


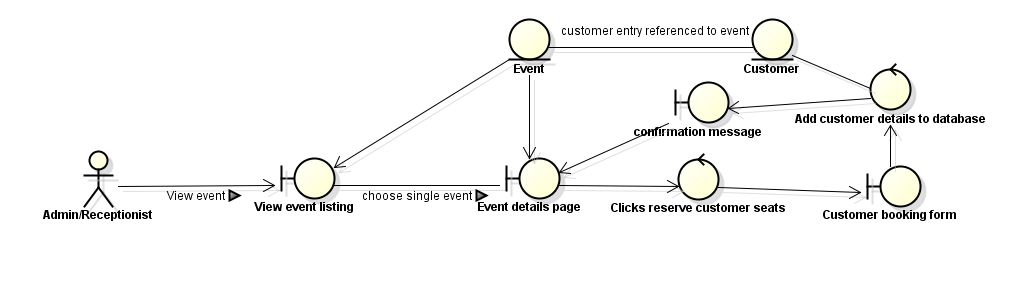
## Appendix H – Flow Diagrams

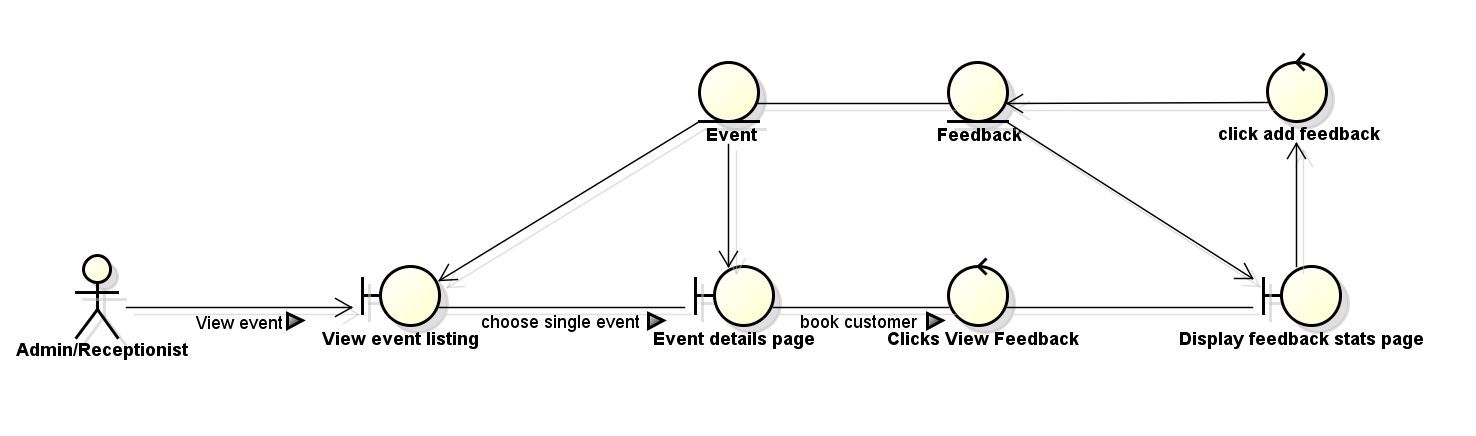




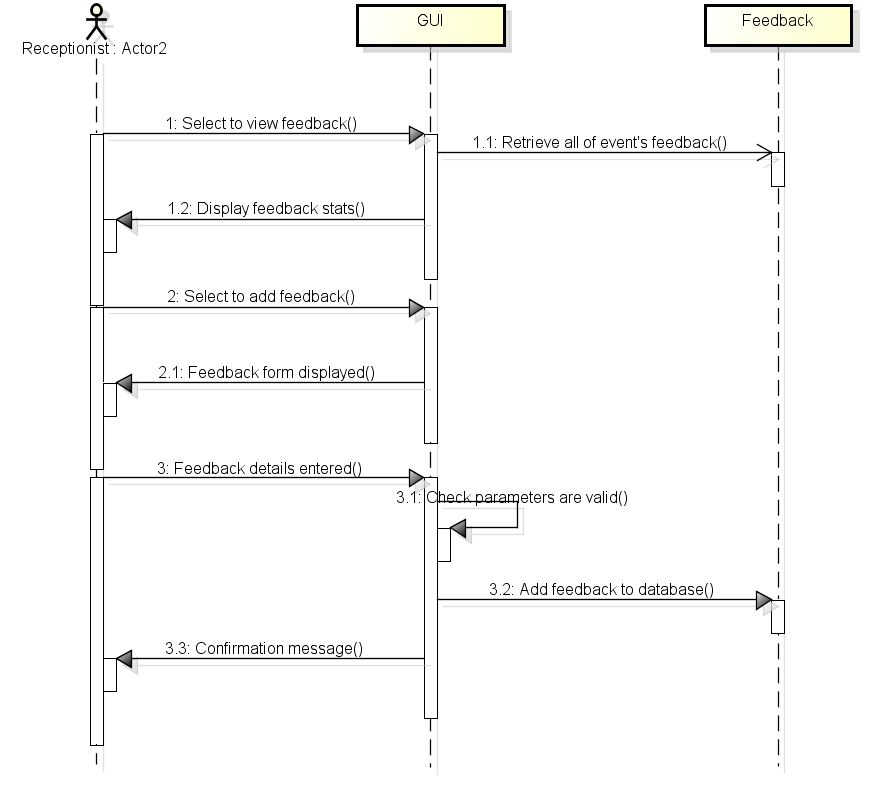
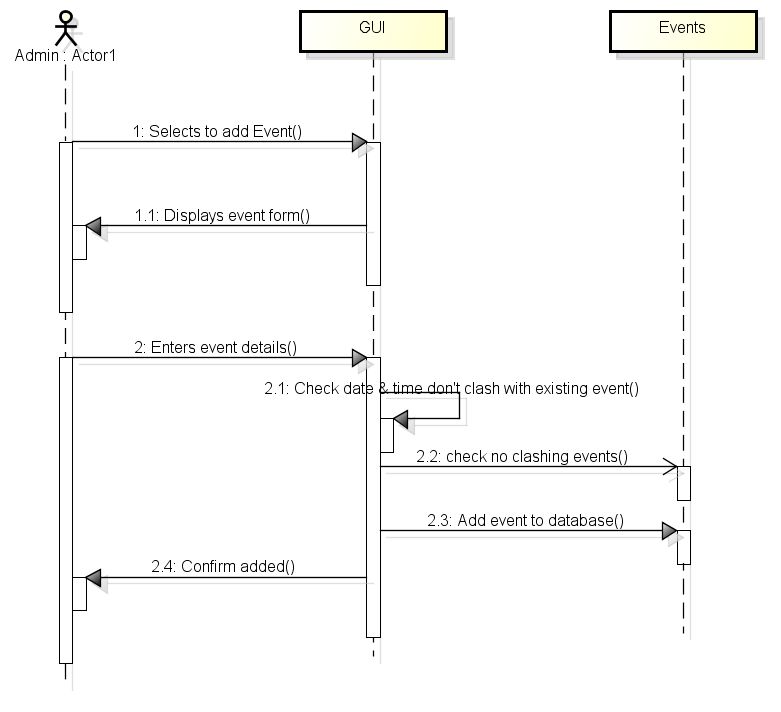
## Appendix I – Robustness Diagrams

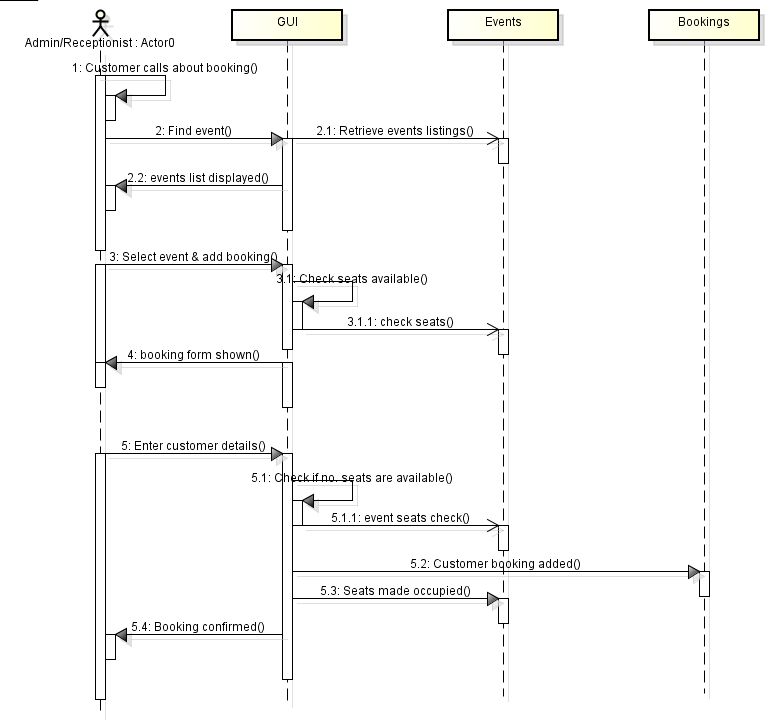




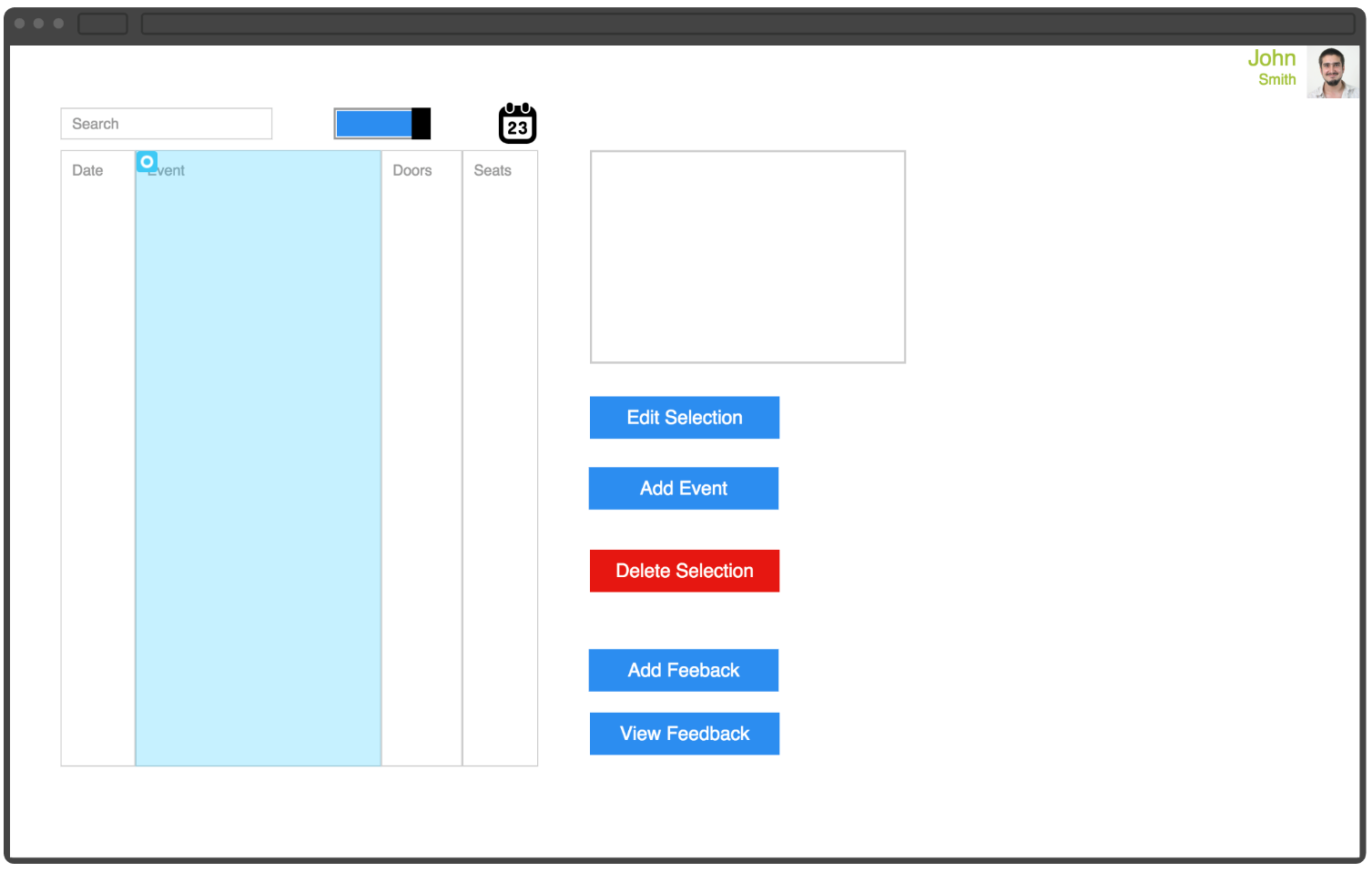


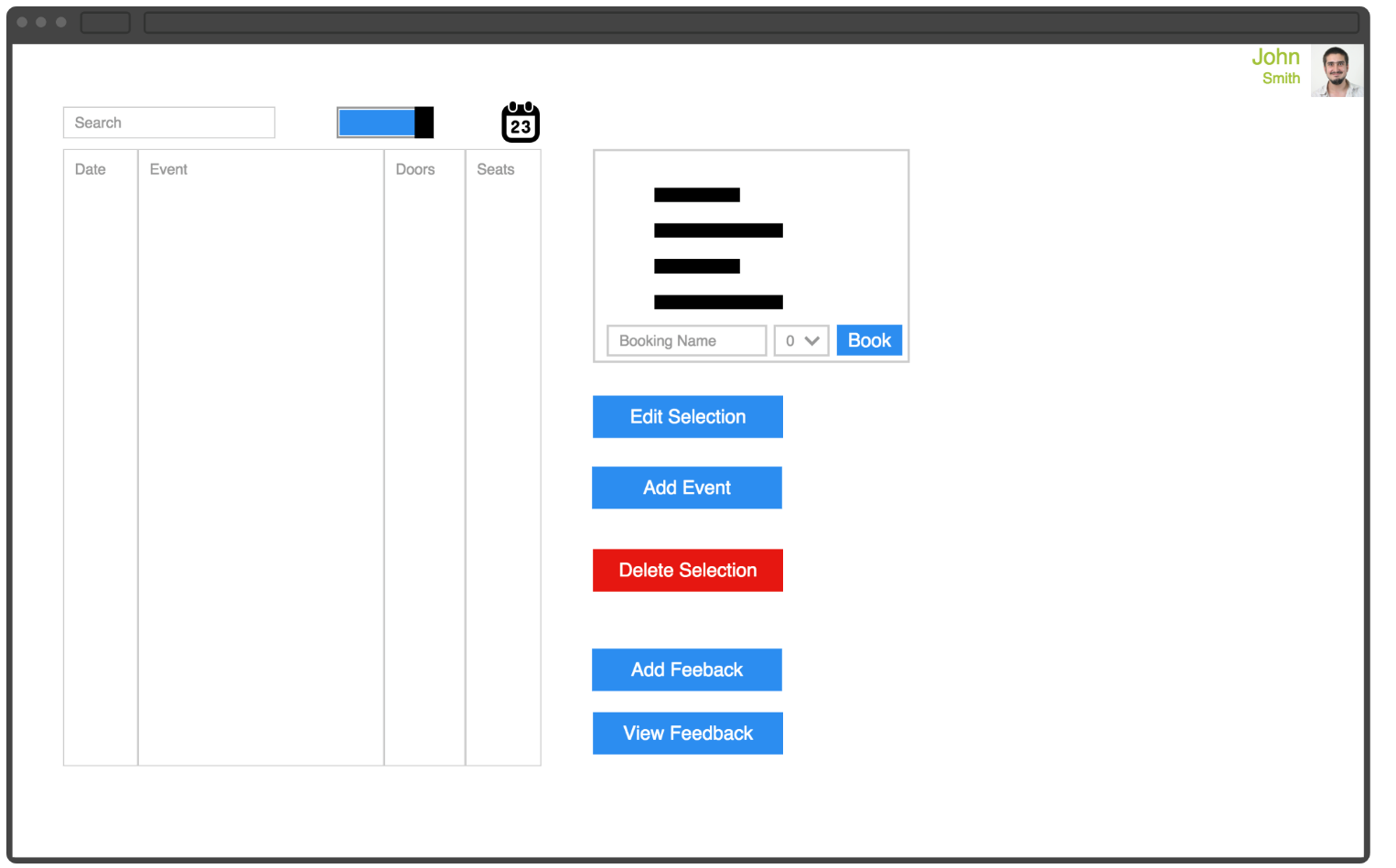
## Appendix J – Sequence Diagrams





## Appendix K – GUI Wireframe Designs







## Appendix L – Effort Matrix

