The Golden Ticket

Software Requirements Document

Brittany Lacy
Drake Lambert
Jeremy LeJeune
Jenna Meadors
Sam Miller

Contents

1 Preface **2**

2 Introduction **3**

3 Glossary **5**

4 User Requirements definition **6**

5 System Architecture **11**

6 System Requirements Specification **12**
6.1 functional Requirements 12
6.2 Non-functional Requirements 15

7 System Models **16**

8 System Evolution **18**

9 Appendices **19**
9.1 Software Dependencies for host 19
9.2 Hardware Dependencies for host 19
9.3 Software Dependencies for client 19
9.4 Hardware Dependencies for client 19

1 Preface

This document is a complete rewrite of the previously submitted version in order to better fit the format required of the requirements document. This document will contain all of the information of the previous document, and will additionally contain the sections as suggested in the textbook.

assumptions:

- The Client will not have access to the ticketing interface
- The boolean isPriority will allow administrators the ability to prioritize tickets

Summary of changes:

- From the first version that was submitted early this semester this document has gone from being a list of tables to a full blown document written in LaTex .

In making this document I intend to layout the master plan for a ticketing system using Microsoft's dot net core and the C# language. This document is intended to be read by anyone intending to create this ticketing system or those intending to use it. Understanding of the C# language will be required in order to create this ticketing system as it was originally designed. The documentation contained at may be particularly helpful in understanding the code and design of this application.

2 Introduction

The ServiceDesk is designed to be a lightweight web based ticketing system written using the c# language. The front end of of the ticketing system will be a responsive designed website so that it is accessible on as many devices as possible. The back-end for the sight will be a micro-services implementation hosted on a debian server featuring sql-lite, the dot net core, identity server 4, lets-encrypt,the apache web server 2.0, and more. The server will feature continuous deployment, when a revision is pushed to the master branch the server will on its check (every minute) download and deploy the change and restart any changes that are needed. In this ticketing system there will be two users of the interface, technicians and admins/managers. This is added text.

1. Technicians will have access to the ticket queue and assigned tickets.
2. Admins will have access to the ticket queue, assigned tickets (if any) and the admin portal.
3. The ticket queue will consist of all tickets that are not marked as complete. The ticket at the top of the queue will be the ticket whose due date is closest to now / farthest past due. Their will be options to filter views on tickets, such as only tickets assigned to you, only tickets at x difficulty level, etc.
4. A ticket will consist of the following items:
    - Title
    - Description
    - Difficulty level
    - A boolean isPriority that can be set by administrators to bring the ticket to the top of the list.
    - Contact
    - Department
    - notes + time entries
    - assigned technician
    - due date/ time
5. A contact will consist of the following items:
    - Name
    - Phone number
    - Email address
    - Title
    - Department name

6. a company will consist of the following items:
    - A list of contacts
    - A list of associated tickets
    - join date
7. A technician will consist of the following items:
    - Join date
    - experience level
    - Email
    - phone
    - list of assigned tickets
8. An administrator will consist of the following items:
    - All items contained within the technician
    - An additional property that tags their account as an administrator

3 Glossary

Glossary

Client The person/entity that manages technicians and client relations, has the power to make tickets priority. 5

Client The person/entity that requests a service from the service company(users of the ticketing system. 2, 5

Entity Framework A framework that allows for .net core applications to automatically create and manage entities such as an SQL-lite database . 5, 19

isPriority A boolean value that sets a ticket as priority, can be set by administrators. 2, 5

Model View Controller A Framework of application development that is used by the Entity Framework in order to generate the SQL-lite database.. 5, 19

Technician The person/entity that works a ticket. 5

4 User Requirements definition

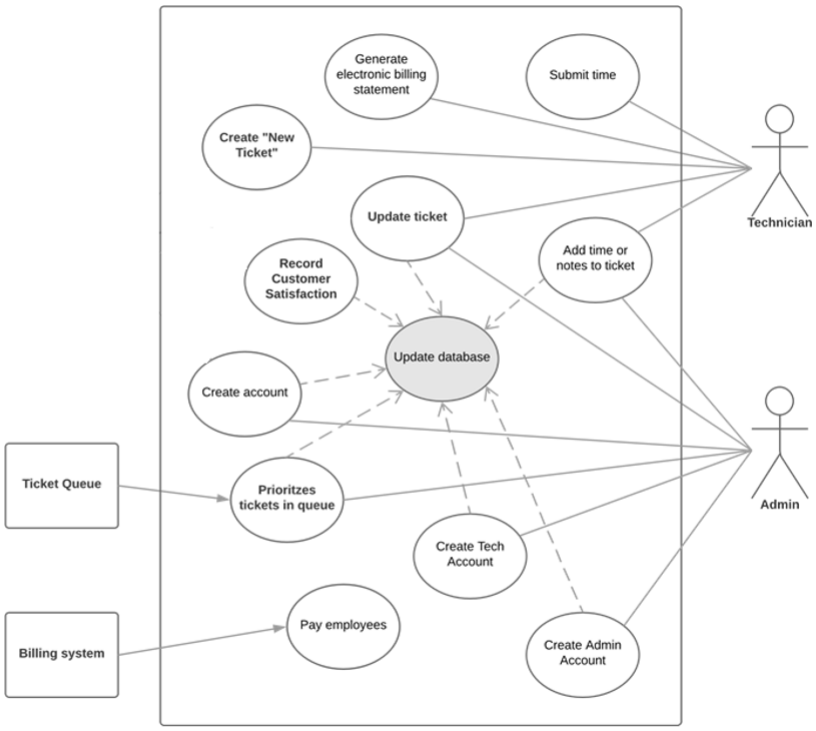

Figure 1: Use Case Diagram
[The above Diagram describes the use cases of the ticketing systems.]

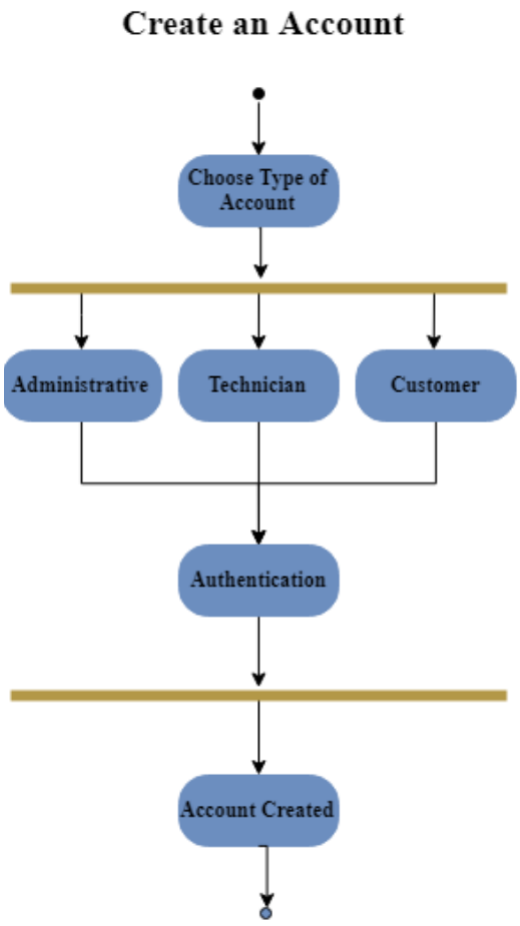

Figure 2: Activity Diagram: Create Account
[The above Diagram describes the process of creating an account]

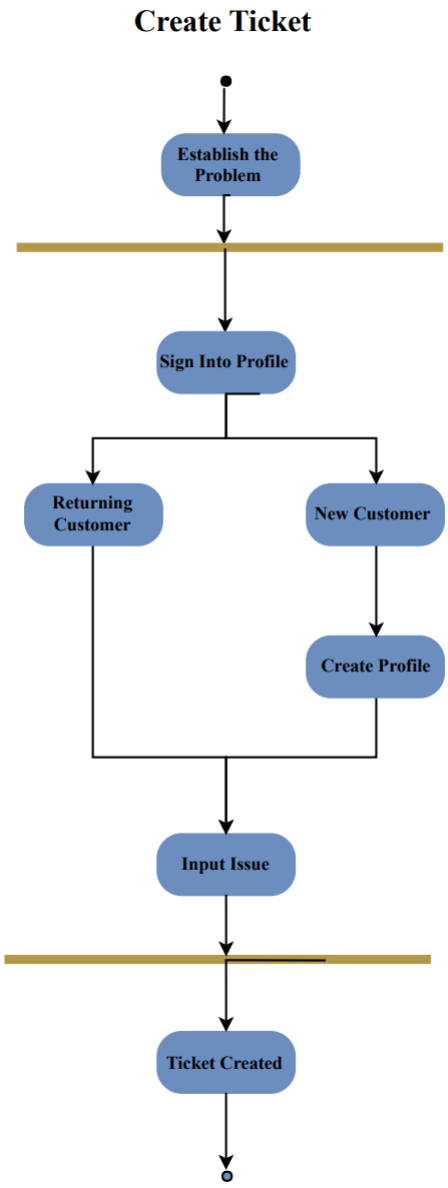

Figure 3: Activity Diagram: Crate Ticket
[The above Diagram describes the process of creating a ticket]

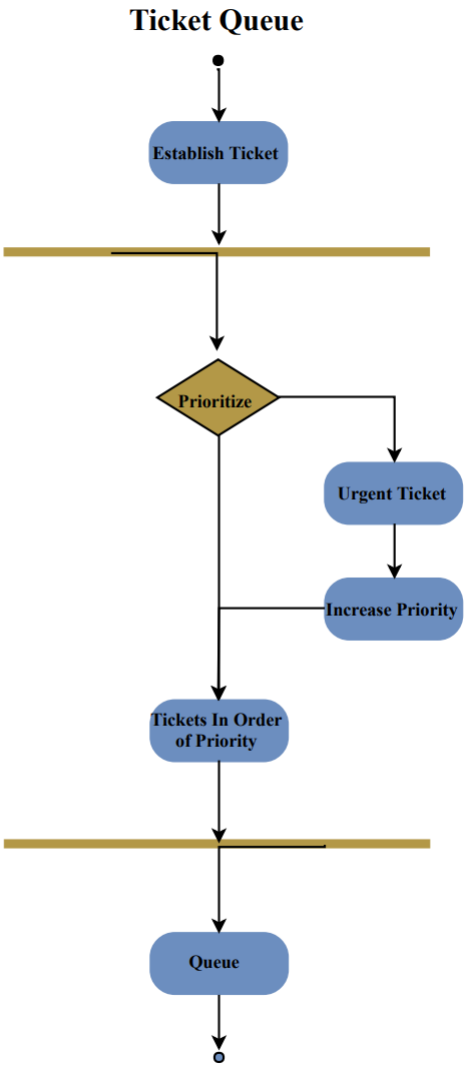

Figure 4: Activity Diagram: Ticket Queue
[The above Diagram describes the process of generating and sorting the ticket queue.]

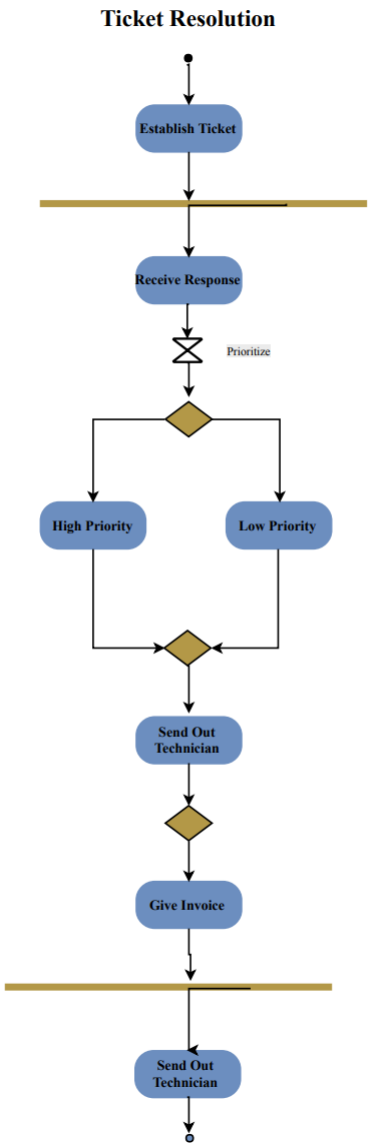

Figure 5: Activity Diagram: Ticket Resolution
[The above Diagram describes the process of ticket resolution.]

5 System Architecture

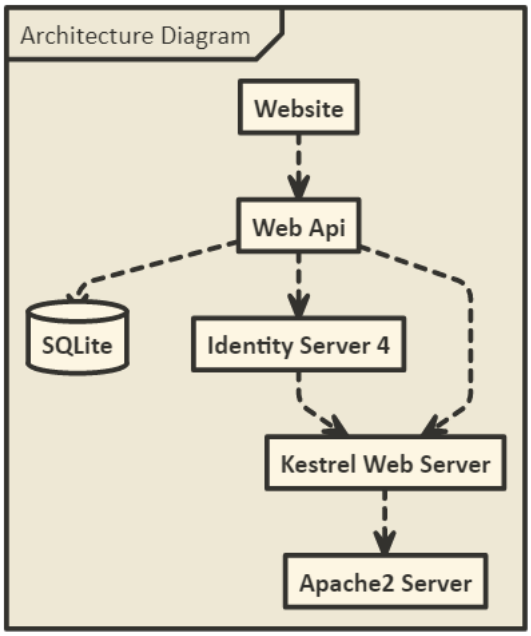

Figure 6: Architecture Diagram
[The diagram describing the architecture of the system]

The above figure describes the main processes and applications that will be used to run this system.

6 System Requirements Specification

6.1 functional Requirements

Function	Create Admin
Description	Create an admin account
Inputs	Name, number and email
Source	An Admin
Outputs	A profile that can be used to manage the ticketing system
Destination	Database
Action	Type in the information and submit
Requires	The customer to input the required information
Precondition	The Admin creating the account must have the necessary info
Postcondition	An account will be created with Admin privileges

Figure 7: Create Admin
[The above table describes the requirements of implementing the process of creating an administrator account.]

Function	Create Tech Account
Description	The Admin will input information for a technician employee account.
Inputs	Name, number and email
Source	The Admin
Outputs	A profile that can be used to create ticket, work a ticket, print an invoice
Destination	Database
Action	Type in the information and submit
Requires	The admin to input the proper information
Precondition	The admin must possess the proper information
Postcondition	The technician will have an account to work in the ticketing system

Figure 8: Create Technician
[The above table describes the requirements of implementing the process of creating a technician account.]

Function	Create Client
Description	Ability to create client entities
Inputs	Company name Contact(s) • Name • Email • Phone • Title
Source	Technician/Admin
Outputs	Client entry
Destination	Client DB
Requires	Company & 1 or more contacts

Figure 9: Architecture Diagram
[The above table describes the requirements of implementing the process of creating a client entry.]

Function	Create a Ticket
Description	A new ticket entry is created
Inputs	Technician Client Ticket description Complexity: (1 -3)
Source	Technician, Admin
Outputs	Ticket: • Automatically Prioritized based on the system outline
Destination	DB
Requires	Client Description
Postcondition	A ticket is stored in the queue

Figure 10: Create Ticket
[The above table describes the requirements of implementing the process of creating a ticket.]

Function	Prioritize Tickets
Description	A prioritization scheme in order to determine in what order to handle customer requests
Inputs	Customer service records
Source	DB
Outputs	Prioritized list of tickets
Destination	Ticket Queue
Action	Tickets ranked by: 1. Customers who have received services from MODEL at least 5 times 2. Customers who have received services from MODEL at least 3 times but less than 5 3. Customers who have received services from MODEL only once 4. New customers

Figure 11: Prioritize Tickets
[The above table describes the requirements of implementing the process of having the ticket queue automatically prioritized based on the number of times that a client has been served and prioritization by administrators.]

Function	Ticket Queue
Description	A queue containing a list of customers, ordered based on the above priorities
Inputs	Ranked list of tickets
Source	Ticket prioritization algorithm
Outputs	A live updating list with ticket name and complexity
Destination	UI
Action	Ordered list is displayed on screen; Admin can move tickets to the top of the list by flagging them as emergency
Requires	Ordered list of tickets

Figure 12: Ticket Queue
[The above table describes the requirements of implementing the ticket queue.]

Function	Payscale Algorithm
Description	Gives automatic raises based on years with the company to technicians
Inputs	Years worked
Source	DB
Outputs	Pay rate
Destination	Billing system
Action	Calculate pay rate $30 base plus $10 per year worked over 1
Requires	Employee profile

Figure 13: Pay-scale Algorithm
[The above table describes the requirements of implementing the process of automatically scaling the billable rate for the technician based on the number of years of service that they have with the company.]

Function	Create a Ticket
Description	A new ticket entry is created
Inputs	Technician Client Ticket description Complexity: (1 -3)
Source	Technician, Admin
Outputs	Ticket: • Automatically Prioritized based on the system outline
Destination	DB
Requires	Client Description
Postcondition	A ticket is stored in the queue

Figure 14: Create Ticket
[The above table describes the requirements of implementing the process of creating a ticket.]

6.2 Non-functional Requirements

- Will be developed using a platform that allows for the project to be demonstrated to the class (Web based)
- Will use UML as the primary modeling tool
- Will keep a record the date, time, length and members present for each face-to-face group meeting
- Use GitHub as a code collaboration environment, to share documents, to stay up to date on due dates, and to communicate about meeting times and share ideas.

7 System Models

The Below model gives a broad overview of the relationships of the different components of the software system and how they interconnect and work together to provide the necessary functions of the ticketing system.

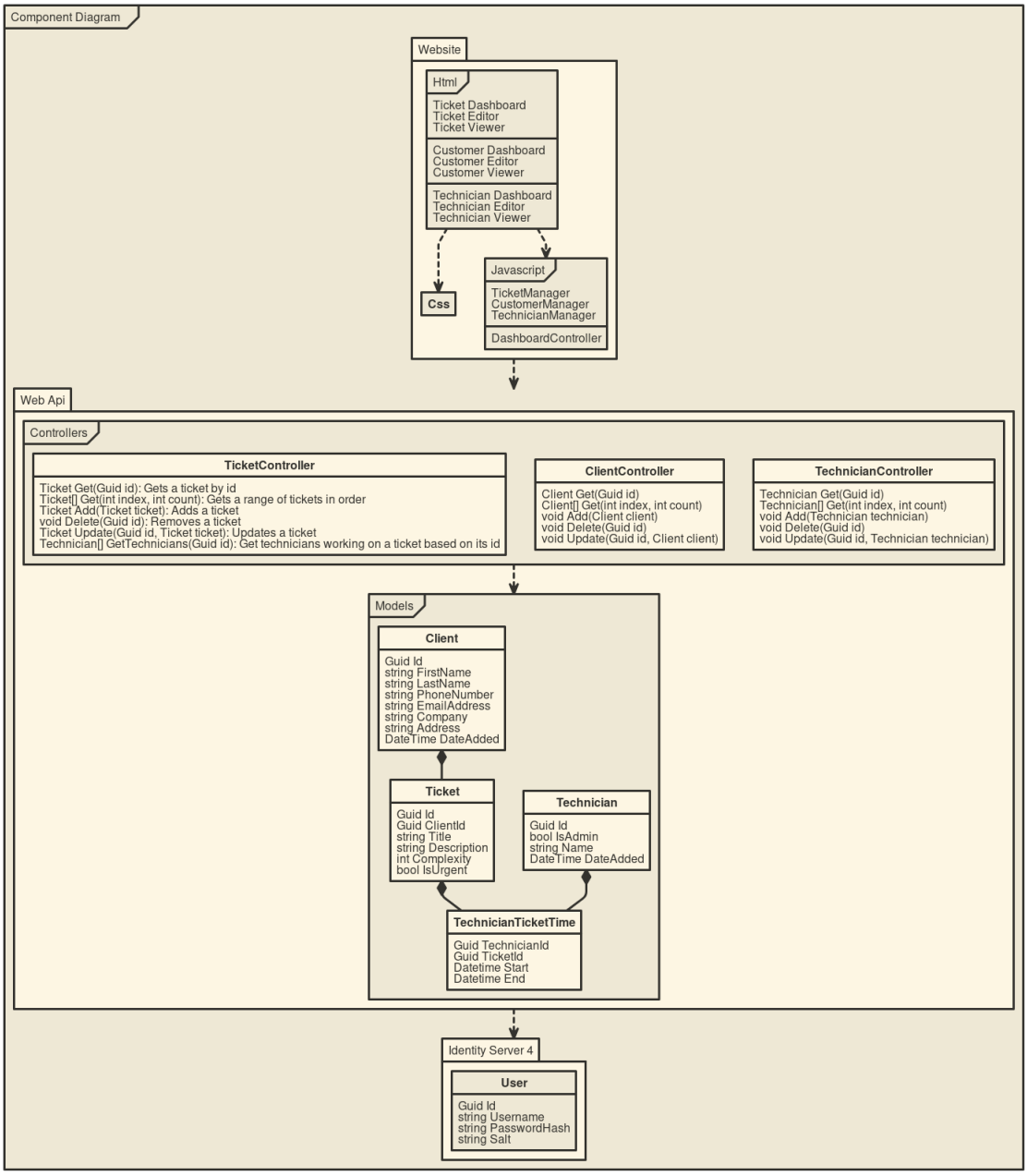

Figure 15: Component Diagram
[The diagram describing the components and classes of the system]

8 System Evolution

One of the biggest assumptions I are making with this system is that the end Employee will not access the system in any capacity. In making this choice I am making the system more secure and the ability for the technicians to speak freely (within the companies guidelines of course)

We assume that the Tickets will be input from emails or phone calls received by administrators and or technicians.

We assume that any and all tickets will be viewable to the pool of technicians, that administrators will ensure qualified technicians take the appropriate tickets.

We assume that admins are the only ones that can create technicians and other administrators.

We assume that the minimum billable hour rate means that no matter how little time is worked on a ticket the customer will be charged at minimum $30.

9 Appendices

9.1 Software Dependencies for host

- Dotnet Runtime 2.1.104
- Apache2 (latest)
- SQL-lite (latest)
- Any modern operating system that can run the above.

Apache2 will function as a reverse proxy, instead of exposing the application directly to the web. Apache2 will receive traffic on port (if port 80 it is automatically redirected to 443) 80 or 443 and then route that traffic to localhost:5000. The application itself builds and manages the SQL-lite databases using the Model View Controller and Entity Framework builtin frameworks.

9.2 Hardware Dependencies for host

- A network connection
- A dual core+ CPU
- 3GB of free-space on the main storage space.

9.3 Software Dependencies for client

- A modern web browser (edge / firefox / chrome)

9.4 Hardware Dependencies for client

- any device capable of running the above web browsers

Alphabetical Index

administrator, 4

company, 4
contact, 3

Hardware Dependencies for client, 19
Hardware Dependencies for host, 19

Software Dependencies for client, 19
Software Dependencies for host, 19

Technician, 4
ticket, 3

List of Figures

1 Use Case Diagram 6
2 Activity Diagram: Create Account 7
3 Activity Diagram: Crate Ticket 8
4 Activity Diagram: Ticket Queue 9
5 Activity Diagram: Ticket Resolution 10
6 Architecture Diagram 11
7 Create Admin 12
8 Create Technician 12
9 Architecture Diagram 13
10 Create Ticket 13
11 Prioritize Tickets 14
12 Ticket Queue 14
13 Pay-scale Algorithm 14
14 Create Ticket 15
15 Component Diagram 17