Alabama Liquid Snake

University of Pretoria

Epi-Use

Botic - Privacy aware chatbot
Process and Methodology

Justin Grenfell - u16028440
Peter Msimanga - u13042352
Alicia Mulder - u14283124
Kyle Gaunt - u15330967
Lesego Mabe - u15055214

Contents

1 Introduction

1.1 Agile Unified Methodology

<Provide a Diagram of the AUM>

Using the Agile Unified Methodology to take advantage of agile principles as well as work with a methodology based off the waterfall process, that has a simple linear and uncomplicated progression is what we have chosen to do. We anticipate that the project will not have a high requirements change throughout development, and this makes the waterfall process derivative methodology more fitting.

The Agile Unified Methodology makes use of Test-Driven Development, which makes it easier to develop high quality code. <Provide details of how we would deal with the set backs of the Test-Driven Development> <Additional motivations>

At the moment, phase 3 has high priority.

2 Planning Phase

2.1 Acquiring Requirements

2.1.1 Specific Functional Requirements and Constraints

2.1.1.1 User Interface

R1.1 The system must allow a customer to enter a query and click on a button to send it.

R1.2 The system must warn a customer of personal information included in a query.

R2.2 The system must be able to highlight personally identifying information according to severity index.

R3.3.2.2 The system must be able to highlight personally identifying information according to severity index.

R3.3.2.3 The system must be able to warn the client representative if they have entered identifying information.

R4.1 The system must allow customers to thumbs up a query response.

R4.2 The system must allow customers to thumbs down a query response.

R5 The system must allow queries to be sent to customer support representatives if not answered satisfactorily.

C1 The system must use an Angular Single Page Application for the user interface.

2.1.1.2 Information Scraper

R2.1 The system must be able to attach a severity to the personally identifying information.

R3.1 The system must scrape its customer query responses for personal information.

R3.3.2.1 The system must be able to identify personal information in a customer representative's response.

C3.1 The system must use word2vec for identifying personal information in customer queries.

C5.2 The system must determine if the response contains personally identifying information.

2.1.1.3 Query Classification

R3.2 The system must be able to classify the user queries.

C3.2 The system must use word2vec for classifying customer queries.

2.1.1.4 Response Generation

R3.3.1 The system must generate a response if it certain that it can.

C5.1 The system must generate an automated response based on the query classification.

2.1.1.5 Chatbot

R1.3 The system must be able to recieve customer queries.

R3.3.2 The system must be able to send the query to a customer support representative if it cannot obtain an appropriate response.

R3.4 The system must be able to send a query response back to a customer.

R4.3 The system must be able to recieve customer feedback.

R5 The system must allow queries to be sent to customer support representatives if not answered satisfactorily.

R8 The system must interface with the currently existing ticket system.

C2 The system must provide an API for the SPA to interact with.

2.1.1.6 Chatbot Trainer

R6.1 The system must store previous customer interactions with positive feedback.

R7 The system must must be trained with previous customer queries and responses.

C4 The system must use Machine Learning or Deep Neural Networks in order to be trained with previous customer queries and responses.

2.1.1.7 Data Persistence

R9.1 The system must scrape customer interation data for personal information before storing.

2.1.2 Functional Clusters and Functional Subsystems

Functional Cluster	Functional Description	System Requirements	Function Subsystem Identified
User Interface	This functional cluster allows customers and customer support representatives to interact with the system	R1.1, R1.2, R2.2, R3.3.2.2, R3.3.2.3, R4.1, R4.2, R5, C1	User Interface Subsystem
Personal Information Identification	This functional cluster identifies personal information within messages	R2.1, R3.1, R3.3.2.1, C3.1, C5.2	Message Scraper Subsystem
Classification of Queries	This functional cluster is responsible for classifying queries.	R3.2, C3.2	Query Classification Subsystem
Automatic Query Response Generation	This functional cluster is responsible for generating query responses	R3.3.1, C5.1	Response Generation Subsystem
Main Program	This function cluster is responsible for coordinating query handling subsystems	R1.3, R3.3.2, R3.4, R4.3, R5, R8, C2	Chatbot Subsystem
ChatBot Training	This functional cluster is responsible for training the system using previous interactions	R6.1, R7, C4	Chatbot Trainer Subsystem
Data Persistence	This functional cluster is responsible for persisting customer interactions	R9.1	Database Subsystem

2.1.3 Traceability Matrix

R	UI	Info Scraper	Query Classification	Resp Generation	Chatbot	Chatbot Trainer	Data Persistence
R1.1	X						
R1.2	X						
R1.3				X			
R2.1		X					
R2.2	X						
R3.1		X					
R3.2			X				
R3.3.1				X			
R3.3.2					X		
R3.3.2.1		X					
R3.3.2.2	X						
R3.3.2.3	X						
R3.4					X		
R4.1	X						
R4.2	X						
R4.3	X						
R5	X				X		
R6.1						X	
R7						X	
R8					X		
R9.1							X
C1	X						
C2					X		
C3.1	X						
C3.2			X				
C4						X	
C5.1				X			
C5.2		X					

Key: UI = User Interface, Info Scraper = Information Scraper, Resp Generation = Response Generation.

2.1.4 Nonfunctional Requirements/Quality Attributes

2.1.4.1 Security Requirements

R1.1. The system must be able to authenticate users and authorize them to access system features.

- R1.1.1. The system must be able to identify and authenticate customers.
- R1.1.2. The system must be able to identify and authenticate customer support representatives.
- R1.1.3. The system must be able to deny users who haven't been authenticated to access system features

R1.2. The system must be able to allow new users to register for user profiles for authentication.

R1.3. The system must be able to allow users to update their password.

R1.4. The system must ensure that confidentiality of customer and customer support representative interactions are ensured and maintained across the system.

R1.4.1. The system must ensure that customers can interact with the system in a secured manner.

R1.4.2. The system must ensure that customer queries are sent in a secured manner.

R1.4.3. The system must ensure that customer support representatives care interact with the system in a secured manner.

R1.4.4. The system must ensure that customer support representative response are sent in a secured manner.

R1.4.5. The system must ensure that all queries and responses are processed in a secured manner.

R1.5. The system must ensure that information disclosed during error management is not revealing of internal architecture, design, and configuration information.

2.1.4.2 Availability

R1.1. The system must have high availability to handle customer queries.

R1.1.1. The system should be available at least 99 percent of the time.

R1.2. The system must ensure that errors that occur throughout the system are handled appropriately and provide sufficient information.

R1.2.1. The system must provide error messages when errors occur.

R1.2.2. The system must ensure to keep a traces that show what led to errors.

R1.3. The system must ensure that errors are localized and that their effect is minimized throughout the system.

2.1.4.3 Reliability

R1.1. The system must ensure that responses to customer queries are done in a reliable manner.

R1.1.1. The system must ensure that customer support representative are authorized to respond to customer queries.

R1.1.2. The system must ensure that queries are responses sent throughout the system are complete and consistent.

R1.2. The system must ensure that it is at least 80 percent certain that an autogenerated response is correct before responding to a query.

2.1.4.4 Performance

R1.1. The system must ensure that personal information is highlighted according to severity in real-time.

R1.1.1. The system must ensure that a severity of a word is recieved within a second of it being typed.

R1.1.2. The system must ensure that a word or set of words containing personal information are highlighted in less than a second after recieving the severity.

2.1.4.5 Maintainability

R1.1. The system must allow for system changes and modifications to the user interface to be made as seemlessly as possible.

R1.2. The system must allow for system changes to the database to be made as seemlessly as possible.

2.2 Deriving Use Cases from Requirements

modeling and anaylsis of misuse cases - role based access rights - nonfunctional requirement associations i.e. security requirements, should be associated using the requirement-use case traceability matrix

Using the steps as defined in [?] page 176 to 192, namely: identifying use cases, specifying use case scopes, visualizing use case contexts, reviewing the use cases and diagrams, and finally allocating the use cases to iterations.

2.2.1 Identifying Use Cases

2.2.1.1 Deriving Use Cases, Actors, and Subsystems

Verb-Noun Phrase	Is it a business process?	Does it begin with an actor?	Does it end with the actor?	Does it accomplish a business task for the actor?	Is it a use case?	Actor	Subsystem
Startup System	Y	Y	Y	Y	Y	Admin	Botic
Shutdown System	Y	Y	Y	Y	Y	Admin	Botic
Ask Bot Assistance	Y	Y	Y	Y	Y	Customer	Botic
Ask Human Assistance	Y	Y	Y	Y	Y	Customer	Botic
Respond to Query	Y	Y	Y	Y	Y	Customer Support Rep	Botic
Provide Feedback	Y	Y	Y	Y	Y	Customer	Botic
Train with Interactions	Y	Y	Y	Y	Y	Admin	Botic
Provide Historic Interactions	Y	Y	Y	Y	Y	Admin	Botic
Login	Y	Y	Y	Y	Y	Customer	Botic
Logout	Y	Y	Y	Y	Y	Customer	Botic
Login	Y	Y	Y	Y	Y	Customer Support Rep	Botic
Logout	Y	Y	Y	Y	Y	Customer Support Rep	Botic
Register	Y	Y	Y	Y	Y	Customer	Botic
Register	Y	Y	Y	Y	Y	Customer Support Rep	Botic
Deregister	Y	Y	Y	Y	Y	Customer	Botic
Register User	Y	Y	Y	Y	Y	Admin	Botic
Deregister User	Y	Y	Y	Y	Y	Admin	Botic
Update Password	Y	Y	Y	Y	Y	Customer	Botic
Update Password	Y	Y	Y	Y	Y	Customer Support Rep	Botic

2.2.1.2 Rearranging Use Cases Among Subsystems

Using role-based partitioning we produce the following use case groupings:

Botic/Admin	Botic/Customer	Botic/Customer Support Rep
UC1: Startup System	UC3: Ask Bot Assistance	UC5: Respond to Query
UC2: Shutdown System	UC4: Ask Human Assistance	UC11: Login
UC7: Train with Interactions	UC6: Provide Feedback	UC12: Logout
UC8: Provide Historic Interactions	UC9: Login	UC14: Register
UC16: Register User	UC10: Logout	UC19: Update Password
UC17: Deregister User	UC13: Register	
	UC15: Deregister	
	UC18: Update Password	

2.2.1.3 Constructing A Traceability Matrix

A requirements use case traceability matrix is useful for a number of reasons, one them being associating the none functional quality attributes to the use cases and another being to see which use cases are not required as well as to see which requirements are not being satisfied. Here is the Requirements Use Case Traceability Matrix for the system, below:

Requirement	Priority	UC1	UC2	UC3	UC4	UC5	UC6	UC7	UC8	UC9	UC10
R1.1	5			X	X						
R1.2	5			X	X						
R1.3	5			X	X						
R2.1	5			X	X	X			X		
R2.2	4			X	X						
R3.1	5			X	X	X					
R3.2	4			X	X	X			X		
R3.3.1	4			X							
R3.3.2	5			X							
R3.3.2.1	5				X						
R3.3.2.2	5					X					
R3.3.2.3	5					X					
R3.4	5					X					
R4.1	3			X			X				
R4.2	3			X			X				
R4.3	3			X			X				
R5	3			X							
R6.1	3			X	X						
R7	4							X			
R8	3										
R9.1	4								X		
UC Priority		5	5	5	5	5	4	4	4	5	5

Requirement	Priority	UC11	UC12	UC13	UC14	UC15	UC16	UC17	UC18	UC19
R1.1	5									
R1.2	5									
R1.3	5									
R2.1	5									
R2.2	4									
R3.1	5									
R3.2	4									
R3.3.1	4									
R3.3.2	5									
R3.3.2.1	5									
R3.3.2.2	5									
R3.3.2.3	5									
R3.4	5									
R4.1	3									
R4.2	3									
R4.3	3									
R5	3									
R6.1	3									
R7	4									
R8	3									
R9.1	4									
UC Priority		4	4	4	4	4	4	3	4	4

In order to associate the quality attributes to the use cases, these below are the non-functional use case traceability matrixes for the system, below:

Security	Priority	UC1	UC2	UC3	UC4	UC5	UC6	UC7	UC8	UC9	UC10
R1.1.1	5									X	
R1.1.2	5									X	
R1.1.3	5	X	X	X	X	X	X	X	X	X	X
R1.2	5										
R1.3	5										
R1.4.1	5			X	X					X	X
R1.4.2	5			X	X						
R1.4.3	5					X				X	X
R1.4.4	5					X					
R1.4.5	5			X	X	X					
R1.5	5	X	X	X	X	X	X	X	X	X	X
Availability											
R1.1.1	5	X	X	X	X	X	X	X	X	X	X
R1.2.1	5	X	X	X	X	X	X	X	X	X	X
R1.2.2	5	X	X	X	X	X	X	X	X	X	X
R1.3	5	X	X	X	X	X	X	X	X	X	X
Reliability											
R1.1.1	4					X					
R1.1.2	4			X	X	X			X		
R1.2	4			X							
Performance											
R1.1.1	3			X	X	X					
R1.1.2	3			X	X	X					
Maintainability											
R1.1	2	X	X	X	X	X	X	X	X	X	X
R1.2	2								X	X	X
UC Priority		5	5	5	5	5	4	4	4	5	5

Security	Priority	UC11	UC12	UC13	UC14	UC15	UC16	UC17	UC18	UC19
R1.1.1	5			X		X			X	
R1.1.2	5	X	X		X					X
R1.1.3	5					X		X	X	X
R1.2	5			X	X		X			
R1.3	5								X	X
R1.4.1	5			X		X			X	
R1.4.2	5									
R1.4.3	5	X	X		X					X
R1.4.4	5									
R1.4.5	5									
R1.5	5	X	X	X	X	X	X	X	X	X
Availability										
R1.1.1	5	X	X	X	X	X	X	X	X	X
R1.2.1	5	X	X	X	X	X	X	X	X	X
R1.2.2	5	X	X	X	X	X	X	X	X	X
R1.3	5	X	X	X	X	X	X	X	X	X
Reliability										
R1.1.1	4									
R1.1.2	4									
R1.2	4									
Performance										
R1.1.1	3									
R1.1.2	3									
Maintainability										
R1.1	2	X	X	X	X	X	X	X	X	X
R1.2	2	X	X	X	X	X	X	X	X	X
UC Priority		4	4	4	4	4	4	3	4	4

2.2.2 Specifying Use Case Scopes

The specification of use case scopes will help us define where the use cases start and where they end. This specification results high-level use cases from the abstract use cases that were previously defined. Below are the high-level use cases for this system:

UC1. Startup System
TUCBW an administrator user clicking on the "Startup System" button (or enters a command) on the Botic Admin dashboard.
TUCEW the administrator seeing a confirmation message that the system is up along with the configuration of the running system.

UC2. Shutdown System
TUCBW an administrator user clicking on the "Shutdown System" button (or enters a command) on the Botic Admin page.
TUCEW an administrator seeing a confirmation message that the system has been shut down.

UC3. Ask Bot Assistance
TUCBW a customer typing a message into the input box in the customer chat page.
TUCEW a customer recieving a "Session has ended" message.

UC4. Ask Human Assistance
TUCBW a customer clicks on the "Ask Human" button in the customer chat page.
TUCEW a customer recieving a "Session has ended" message.

UC5. Respond to Query
TUCBW a customer support representative clicking on a customer query in the customer support chat page.
TUCEW a customer support representative recieves a "Query Resolved" message.

UC6. Provide Feedback
TUCBW a customer clicking either the thumbs up or thumbs down buttons in the customer chat page.
TUCEW when the feedback button (either thumbs up or thumbs down) changes color; in the case that it is thumbs down, the "Ask Human" button should appear.

UC7. Train with Interactions
TUCBW an administrator clicks the "Train AI" button in the Botic Admin page.
TUCEW the administrator sees the message "AI Has Been Trained" accompanied with the report.

UC8. Provide Historic Interactions
TUCBW an administrator clicks the "Load Historic Data" button in the Botic Admin page.
TUCEW the administrator sees the message "Data has been Recieved" accompanied with the report.

UC9. Login
TUCBW a customer clicks the "Sign In" button on the home page.
TUCEW a customer gets redirected to the customer chat page (and it is opened).

UC10. Logout
TUCBW a customer clicks the "Sign Out" button on the customer chat page.
TUCEW a customer gets redirected to the home page.

UC11. Login
TUCBW a customer support representative clicks the "Sign In" button on the home page.
TUCEW a customer support representative gets redirected to the customer support chat page.

UC12. Logout
TUCBW a customer support representative clicks the "Sigh Out" button on the customer support chat page.
TUCEW a customer support representative gets redirected to the customer support chat page.

UC13. Register
TUCBW a customer clicks on the "Register" button on the home page.
TUCEW a customer gets the message "You have been Registered" and gets redirected to the home page.

U14. Register
TUCBW a customer support representative clicks on the "Register" button on the home page.
TUCEW a customer support representative recieves a notification with the message "You have been Approved."

U15. Deregister
TUCBW a customer clicks on the "Deregister" button on customer chat page.
TUCEW a customer recieves "Customer has been successfully deregistered" message and is redirected to the home page.

U16. Register User
TUCBW an Admin clicks on the "Register User" button on the Botic Admin page.
TUCEW the Admin receives the message "User Has Been Registered."

U17. Deregister User
TUCBW an Admin clicks on the "Deregister User" button on the Botic Admin page.
TUCEW the Admin recieves the message "User Has Been Deregistered."

U18. Update Password
TUCBW the customer clicks the "Update Password" link on the customer chat page.
TUCEW the customer recieving the message "Password Updated" and being redirected to the home page.

U19. Update Password
TUCBW the customer support representative clicks the "Update Password" link on the customer support chat page.
TUCEW the customer support representative recieves the message "Password Updated" and being redirected to the home page.

2.2.3 Visualizing Use Case Contexts

Here a "User" encompasses: Administrator, Customer and Customer Support Representative.

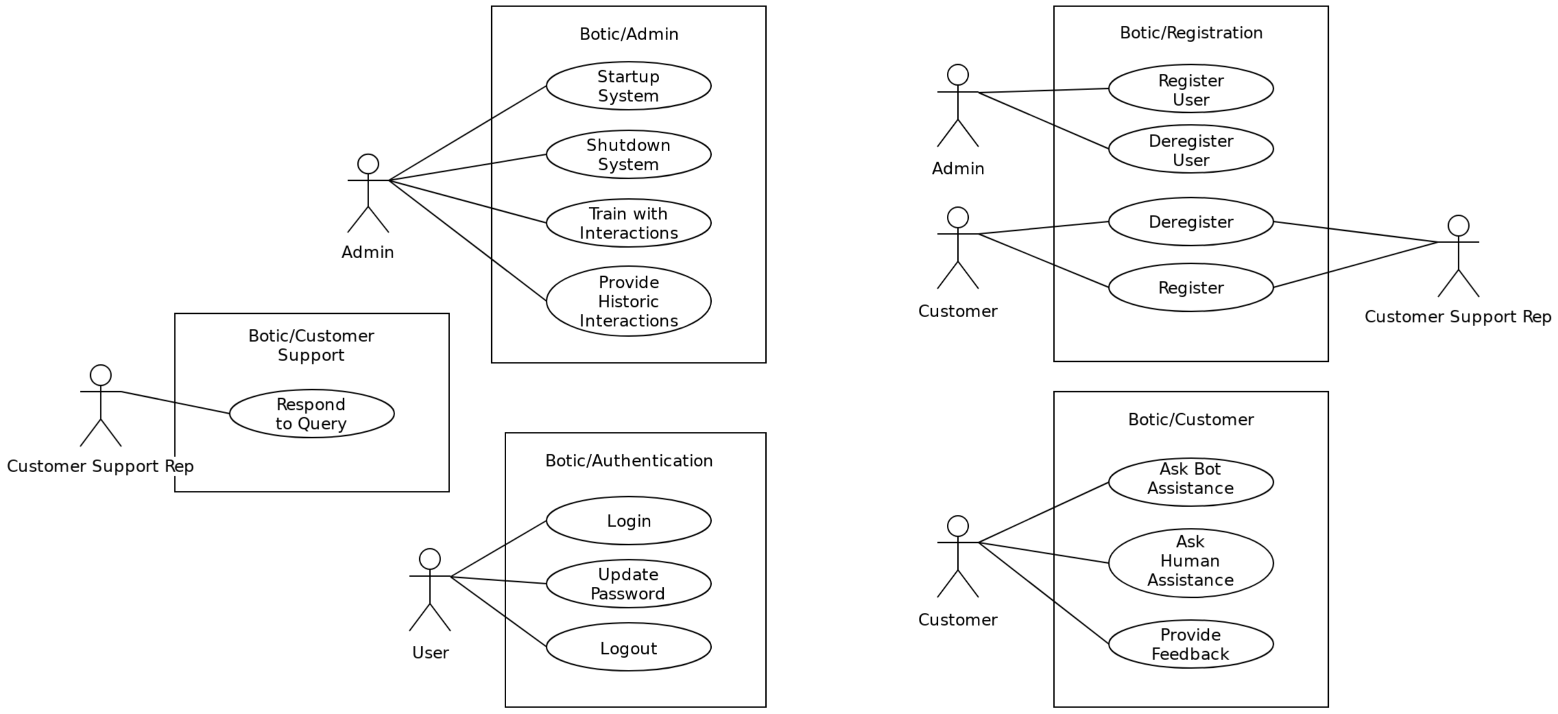

Figure 1: Use Case Contexts

2.3 Allocating Use Cases and Subsystems to Iterations

2.4 Producing an Architecture Design

- Links to relevent documentation - Special consideration should be made during modeling and analysis to show resources that need protection and entities that access those resources - Apply security patterns and security design principles to ensure that security requirements are satisfied. - security test plan to guide the security test process

Planning Phase: Succeeded. - First meeting with client - Use Case-iteration allocation matrix here.

3 Iterative Phase

Interative Phase: - All artifacts are listed and the changes can be checked out by cross referencing appropriate artifacts. - Each phase should logically include changes to all the implementation documentation. - Special consideration should be made during modeling and analysis to show resources that need protection and entities that access those resources - Practice secure coding to ensure that security principles and security patterns are applied to produce secure code during the implementation phase– use quality assurance reviewing to make sure of this. - Also test for security– test for each quality requirement really; apply static and dynamic security testing. Functional testing, security test - Domain modeling should identify and capture security related domain concepts and relationships like the roles and resources accessed by the roles as well as related access priviledges - Design for security and other non functional requirements important during actor-system interaction modeling

3.1 Phase 1:

- Demo 1 happened here. - Phase 2 artifacts to be listed here.

3.1.1 Accomodating Requirements Change

3.1.2 Domain Modeling

3.1.3 Actor-System Iteraction Modeling and User Interface Design

3.1.4 Behavior Modeling and Responsibility Assignment

3.1.5 Deriving Design Class Diagram

3.1.6 Test-Driven Development

3.1.7 Integration

3.1.8 Deployment

3.2 Phase 2:

- Demo 2 happened here - Phase 2 artifacts to be listed here.

3.2.1 Accomodating Requirements Change

3.2.2 Domain Modeling

3.2.3 Actor-System Iteraction Modeling and User Interface Design

3.2.4 Behavior Modeling and Responsibility Assignment

3.2.5 Deriving Design Class Diagram

3.2.6 Test-Driven Development

3.2.7 Integration

3.2.8 Deployment

3.3 Phase 3:

3.3.1 Accomodating Requirements Change

- Another meeting with clients (include proper date). - Refined the requirements according to proper rules thus we must refine the use cases. Link to the new SRS. - Updated architectural desgin. - Link the updated specific requirement and the use cases. - List of use cases to implementated (before and after) - No actionable customer feedback

P: Iteration Use Cases - Haven't produce new ones out of necessity. - Placing emphasis on the Chatbot component that is meant to use all subsystem. - Using updated Architecture

3.3.2 Domain Modeling

- Domain Model has been updated.

3.3.3 Actor-System Iteraction Modeling and User Interface Design

3.3.4 Behavior Modeling and Responsibility Assignment

3.3.5 Deriving Design Class Diagram

3.3.6 Test-Driven Development

3.3.7 Integration

3.3.8 Deployment

4 References