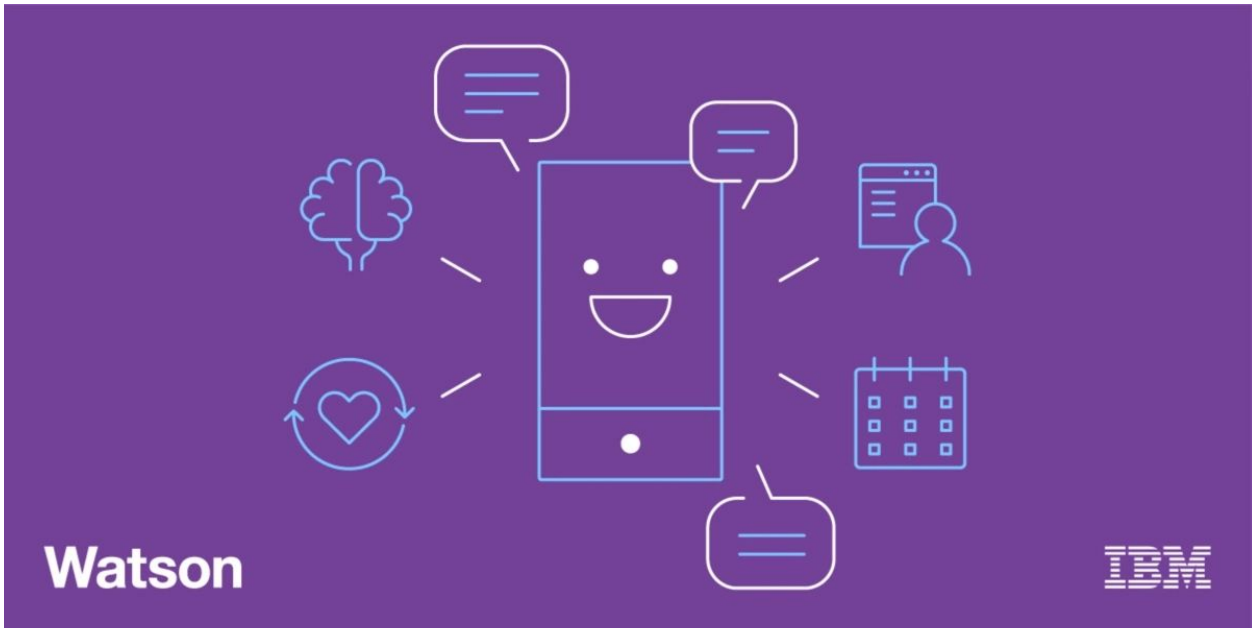


IBM Watson Assistant Object-Oriented ChatBot

CSC 450: Software Engineering

Deliverable #4

Alex Mello, Cameron Detig, Clayton Dugo,
Hannah Moore, Ian Norris, James Fuchs & William Purdy

Table of Contents

1. Introduction

1.1. Project Statement and Objectives

The purpose of our project is to create a chatbot using IBM Watson Assistant to interpret and answer questions appropriating based on a specific topic. The chatbot is designed to be used by students taking an object-oriented class to help answer common questions about the subject.

1.2. Major Functions

The end-user will be able to ask various questions about the concepts that make up object-oriented programming and receive a well explained response. The chatbot will have the ability to define the basic concepts of object-oriented as well as give examples using Java.

1.3. Performance Limitations

There are a few performance issues that may arise when using this chatbot. First, the end-user may be using a different programming language than Java and find the answer that was provided to be unhelpful due to it being in another language. Lastly, there are many ways to answer one question. Therefore, there is the possibility that the AI does not understand the question being asked by the end-user and results in a bad answer or no response.

1.4. System Reference Diagram

The system reference diagram shows the process the chatbot will execute when the user inputs a question. Once the user inputs their question into the chatbot interface, using IBM Watson Assistant, the question is interpreted into an intent that has been created by the developers. The intent is selected based on if the user's question matches up with one of the intents the developer has created. From there, the intent grabs the entity, or response, the chatbot should output. The entity is then formatted into a proper sentence that answers the user's question and is printed out into the chatbot interface.

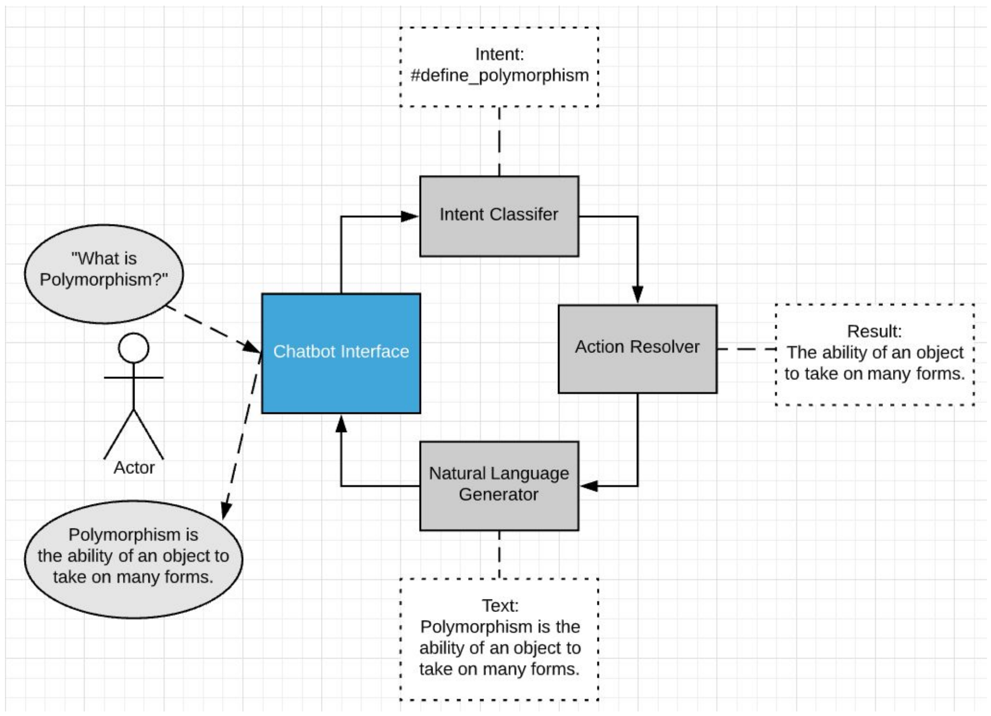

Intent:
#define_polymorphism
Intent Classifer
"What is Polymorphism?"
Chatbot Interface
Action Resolver
Result:
The ability of an object to take on many forms.
Actor
Natural Language Generator
Polymorphism is the ability of an object to take on many forms.
Text:
Polymorphism is the ability of an object to take on many forms.

Figure 1: System Reference Diagram

1.5. System Requirements

Based upon the focus group of who will be using the chatbot, a list of functional and nonfunctional requirements that should be included in the system has been created. The functional requirements address what the system needs to function appropriately and successfully whereas the nonfunctional requirements address usability and maintainability. For features that are essential to have in the system, "The system shall" is stated, whereas for features that are considered nonessential to the system, "The system should" is stated. The requirements are listed starting with the most essential features listed first.

1.5.1 Functional Requirements

	Requirement Description
1	The system shall allow the user to ask questions about the concepts of Object-Oriented Programming and the system give definition-based answers
2	The system shall allow the user to ask basic coding related questions regarding Object-Oriented Programming and the system give example code in Java as an answer
3	The system shall be able to interpret the same question asked in different ways
4	The system should allow the user to input their coding error as a question and the system respond with possible reasons for this error to occur
5	The system should allow the user to ask follow up questions to the answer provided by the system

Table 1: Functional Requirements

In regards to the top three requirements listed, these are considered the most essential to have a successful chatbot designed for answering Object-Oriented related questions. The chatbot will have the ability to answer questions such as, "What is an object?" and "How do I set up a class?". These questions can be asked in various ways, therefore, the chatbot will be able to interpret all of the possible variations and give an appropriate answer.

Requirements 4 and 5 are non essential to the success of the chatbot. Depending on the common errors that occur while using Object-Oriented programming, the chatbot should be able to recognize the error and respond with multiple possible reasons this error could occur. Finally, if the user has a follow up question to an answer the chatbot has just provided, then the user will be able to continue the conversation by asking their follow up question.

1.5.2 Nonfunctional Requirements

	Requirement Description
1	The system shall be easy to navigate and utilize.
2	The system shall be easy for developers to add missing answers.
3	The system should be able to handle multiple follow up questions.

Table 2: Nonfunctional Requirements

1.6. Software Project Constraints

The constraints for this project have been formatted in the table below. The table lists the constraints with a description of how they related to the chatbot.

Constraint	**Description**
Scope	The chatbot must be able to answer basic concept and coding questions related to object-oriented programming
Time	The chatbot must be completed within the length of the semester.

Quality	The chatbot must give appropriate/reasonable answers that are viewed as helpful to the user.
Resources	The chatbot must be created using IBM Watson Assistant.

Table 3: Project Constraints

2. Project Estimates

2.1. Historical Data Used for Estimation

In order to properly access the amount of time required for designing a chatbot with the IBM Watson platform, it was important for the team to reference historical data based on the time taken to implement a project of a similar scope.

In the case of our team, the main commonality was the end of semester project for the Object Oriented Programming and Design class. While the actual project wasn't the same for everyone, it did require us to work in groups over a substantial amount of time. Using this past experience and a Top-Down approach we were able to estimate the time required to accomplish our goals for the IBM Watson chatbot project.

PHASE	ESTIMATED HOURS	ACTUAL HOURS
Project Planning	15	20
Environment Setup	10	-
Design	30	-
Implementation	105	-
Testing & Edge Cases	50	-
Documentation	20	-

Table 4: Estimated hours vs. actual hours spent on each phase of the project.

Function	Estimated Lines of code
General Questions	100
Variable Definitions	75
Error Message Debugging	150
Edge Case Patching	50

Table 5: Estimated lines of code for each section of project calculated using historical data

3. Project Risks

When looking at the entirety of a project's timeline, risks or issues that could be detrimental to the project's completion time and overall health must be evaluated. If risks are not properly addressed, assessed, and dealt with, issues could arise further down the project's projected path. Below is a risk table associated with our group's project. Each row details the risk, the category the risk falls under, the probability the risk would occur, the impact on the project if the issue came to fruition (on a scale from 1- negligible, 2- marginal, 3- critical, 4- catastrophic), the risk resolution, and any additional notes associated with the risk.

One noticeable difference between most project risk assessments and this one is the consideration of cost. With this being a school project, project extensions or additions only result in a cost of time, rather than a monetary value. Additionally, the IBM team and our team members have discussed ways to navigate around the costs brought on by working with Watson Assistant and other tools needed to complete the project.

Risk Assessment Table

Risk	Category	Probability	Impact	Resolution	Notes
Team member(s) is forced to remain in quarantine and miss in-person meetings	Schedule	75%	1	Team can hold virtual meetings to accommodate until quarantine window is over	High probability due to COVID-19
Team member(s) become sick	Schedule	75%	1 to 2	Team can hold virtual meetings to accommodate unless more severely sick, whereas a team member's task would be delegated to the remaining members.	High probability due to COVID-19
Lack of sample data to feed to AI	Performance and Support	60%	3	Additional methods to gather sample data would be implemented	
Team members lacking experience to finish required tasks	Support and Schedule	40%	2	Team member would have to go through training/tutorials to gain familiarity to project if time permits; if deadline is approaching the team member can seek assistance from other members on the team	This % will lower as members gain experience throughout the project
Project scope is too large	Performance and Schedule	50%	3	Project's scope and feature priority would have to be reevaluated and reassessed	

Project scope is too small	Performance	20%	2	Project scope, if enough time permits, would be expanded upon and additional features could be added	
Project isn't completed by deadline	Performance and Schedule	30%	4	Project could continue after deadline but with significant loss to classroom grade and risk of further complications	This risk assessment would have to be reevaluated as deadline approaches
Project scope/requirements change	Performance and Schedule	90%	2	Additional requirements would be assessed and delegated to team members; loss of requirements would cause team members on that task to be reassigned	With project still early in its conceptualization, project details are sure to change
Project is behind schedule	Schedule	40%	2 to 3	Remaining time until deadline would be evaluated and team members could be added to the tasks that are causing difficulty	This risk assessment would have to be reevaluated as deadline approaches
Project cost and cost of associated tools increase	Cost	20%	3 to 4	Attempt to find less expensive tools and request access from IBM team to costly features	High impact due to small budget restrictions

Table 6: Risk Assessment

4. Schedule

4.1. Project Work Breakdown Structure

The following breakdown structure was formed based on the five deliverables needed to be completed within a 14 week period. Each deliverable requires several tasks that should be done to complete that deliverable. These items make up the sub-tasks listed within the five main tasks.

- 1.1 Project planning
  - 1.1.1 Meet with IBM clients
  - 1.1.2 Identify needed information to use IBM Watson
  - 1.1.3 Become familiar with IBM Watson
  - 1.1.4 Establish project statement
  - 1.1.5 Identify project risks, estimates and resources
- 1.2 Software requirements specification
  - 1.2.1 Establish products functionality
  - 1.2.2 Identify products requirements and constraints
  - 1.2.3 Determine products use cases
- 1.3 Software design specification
  - 1.3.1 Define desired output/control/input
  - 1.3.2 Establish Data and Architectural design
  - 1.3.3 Define Interface Design/Specification
  - 1.3.4 Review output/control/input with IBM clients
  - 1.3.5 Begin Implementation
- 1.4 Test plan with test specifications
  - 1.4.1 Define test plan
  - 1.4.2 Perform functional, structural and integration testing
  - 1.4.3 Evaluate test results

1.4.4 Revise as needed

1.5 Source code and documentation

1.5.1 Define description of implemented system

1.5.2 Summarize programming experience

1.5.3 Source code listing

4.2. Task Network (CPM)

The Task Network diagram visualizes the project work breakdown structure. This visualization helps determine the critical path of the project as well as visualize tasks that are done in sync.

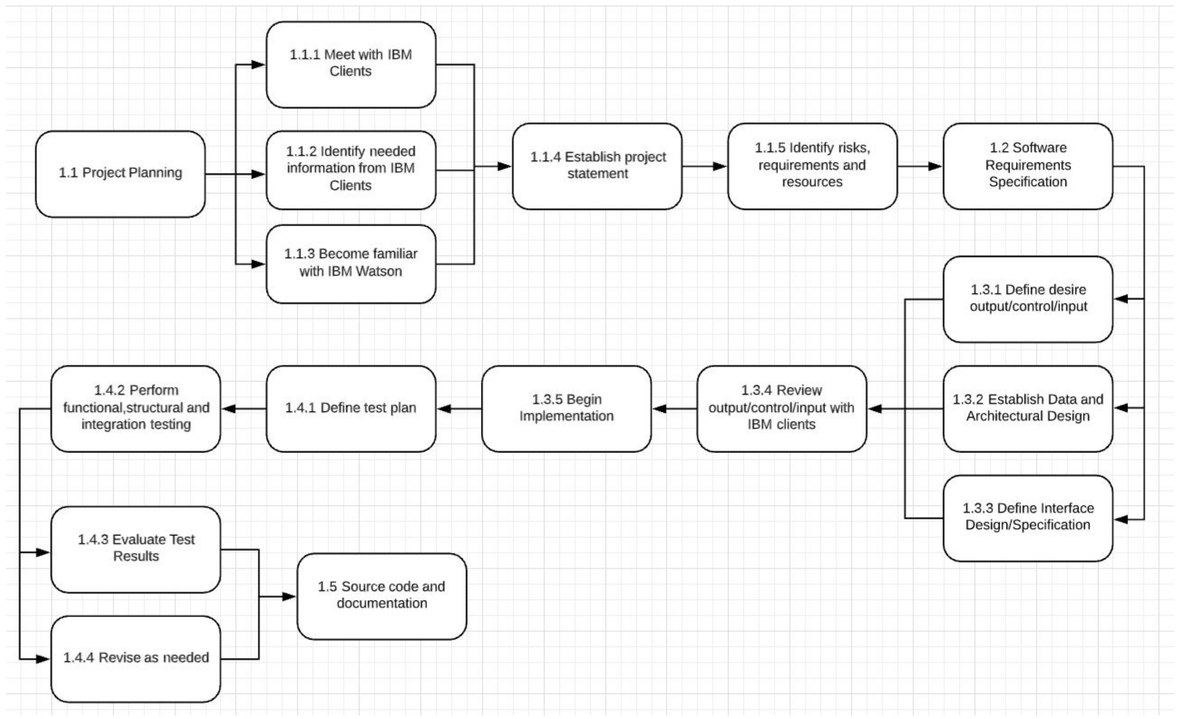

1.1 Project Planning
1.1.1 Meet with IBM Clients
1.1.2 Identify needed information from IBM Clients
1.1.3 Become familiar with IBM Watson
1.1.4 Establish project statement
1.1.5 Identify risks, requirements and resources
1.2 Software Requirements Specification
1.3.1 Define desire output/control/input
1.3.2 Establish Data and Architectural Design
1.3.3 Define Interface Design/Specification
1.3.4 Review output/control/input with IBM clients
1.3.5 Begin Implementation
1.4.1 Define test plan
1.4.2 Perform functional,structural and integration testing
1.4.3 Evaluate Test Results
1.4.4 Revise as needed
1.5 Source code and documentation

Figure 2: Task Network diagram

4.3. Time-line Chart

A time-line chart was formed based on the task network diagram. This chart breaks down the length of time each task should take. The chart is subject for change throughout the project but will stay within the 14 week limit. Multiple tasks will have to be implemented at the same time to ensure the project is completed on time.

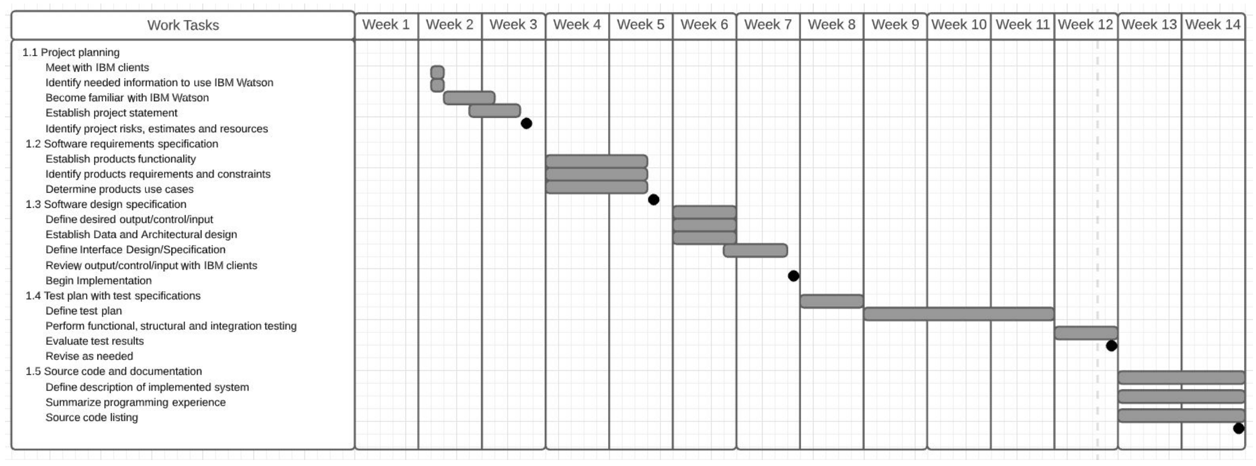


Figure 3: Timeline chart

5. Project Resources

5.1. People

- Project Team Members - Alex Mello, Cameron Detig, Clayton Dugo, Hannah Moore, Ian Norris, James Fuchs, William Purdy
- IBM Employees - Rebecca James, Ryan Brink
- UNCW Instructors - Ronald Vetter, Shauna White

5.2. Hardware

- Web server (Satoshi server)

5.3. Software

- IBM Watson Assistant

6. Functional Description (Data Flow Diagram)

The Data Flow Diagram shows the flow of information between the user and the chatbot. Using the web interface, the user inputs their question. That is then sent to Watson, where the custom dialog nodes use the databases of intents and entities to generate a response. The response is then sent back through the pipeline and displayed for the user who can then continue with another question.

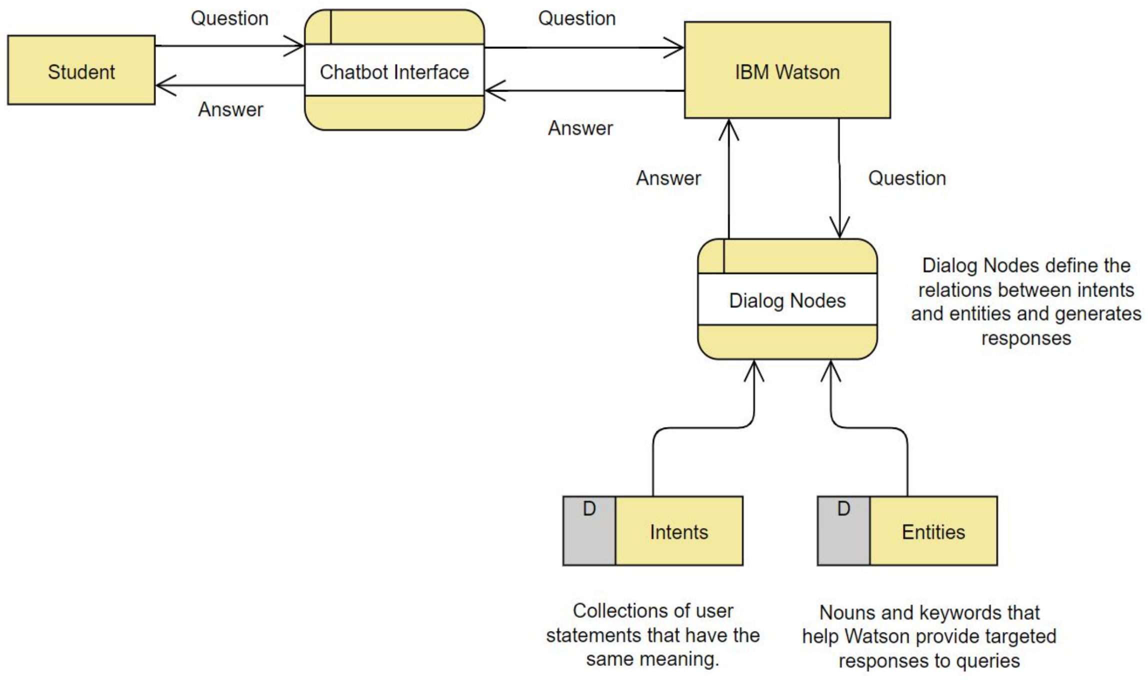

Question
Student
Answer
Chatbot Interface
Question
Answer
IBM Watson
Answer
Question
Dialog Nodes
Dialog Nodes define the relations between intents and entities and generates responses
D
Intents
D
Entities
Collections of user statements that have the same meaning.
Nouns and keywords that help Watson provide targeted responses to queries

Figure 4: Data Flow Diagram

6.1 Functional Partitioning

- Chatbot Interface: A website with the Watson Assistant integrated into it where the user can input questions to the system and receive output.
- IBM Watson Assistant: All of the processing will occur here. The base functionality and natural language processing of Watson is abstracted away, but the team will define the dialog nodes, intents, and entities that will give functionality to the system.
- Dialog Nodes: This is where the relations between intents and entities are defined, and where responses are generated.
- Intents Database: A collection of possible user questions and variations of how they could be phrased that help Watson identify what the user is asking.
- Entities Database: A list of possible subjects or "nouns" that the intents could be referring to.

6.2 Functional Description

Our research indicates that object oriented programming is the most difficult class for computer science students at the University of North Carolina Wilmington. A combination of the new concept of objects and the deeper dive into the language of java proves difficult for most students. We set out to improve this experience for students by creating a chatbot that can answer common questions and solve common errors for them. Using the IBM Watson Cloud platform, we expect to be able to identify and address all common questions that result from the UNCW 331 Object Oriented Programming class.

6.2.1 Processing Narrative

Version	Date	Authors	Description
0.01	9/27/20	Alex Mello, Cameron Detig, Clayton Dugo, Hannah Moore, Ian Norris, James Fuchs, William Purdy	Original outline of OOP Chatbot

Table 7: Processing Narrative

6.2.2 Restrictions/Limitations

The main restriction or limitation that we expect to run into during the development of this project is the constraint applied to free accounts of the IBM cloud platform. A strictly free 'Lite' account has a limit of 10,000 messages a month, a maximum of 5 dialog skills, and only web chat and service desk integration. However the 'Plus' accounts have unlimited messages per month, up to 50 dialog skills and 10 versions per skill, and an additional 50 dialog skills. After a discussion with our IBM contacts Ryan Brink and Rebecca James we were able to identify a solution as a group. Each month a user creates an account and registers for the 'Plus' free trial version. That account shares permissions with the rest of the group and is used for the month. At the end of the month we export the project as a JSON file and another user starts a new free trial and imports the project.

6.2.3 Design Constraints

After an internal discussion with the members of our group and our contacts at IBM we were not able to identify any design constraints within the scope of our project. We expect to be able to deliver a chatbot that improves the learning process of the average student taking Object Oriented Programming.

6.3 Control Description

We know our fundamental user requirements thoroughly. The user will enter in a question. The chatbot then will search a database for the exact or similar question. If these parameters are satisfied the search will return an answer. If not it will return a query to ask the question in a different way. This way the user's interaction with the software is simple.

6.3.1 Control Specification

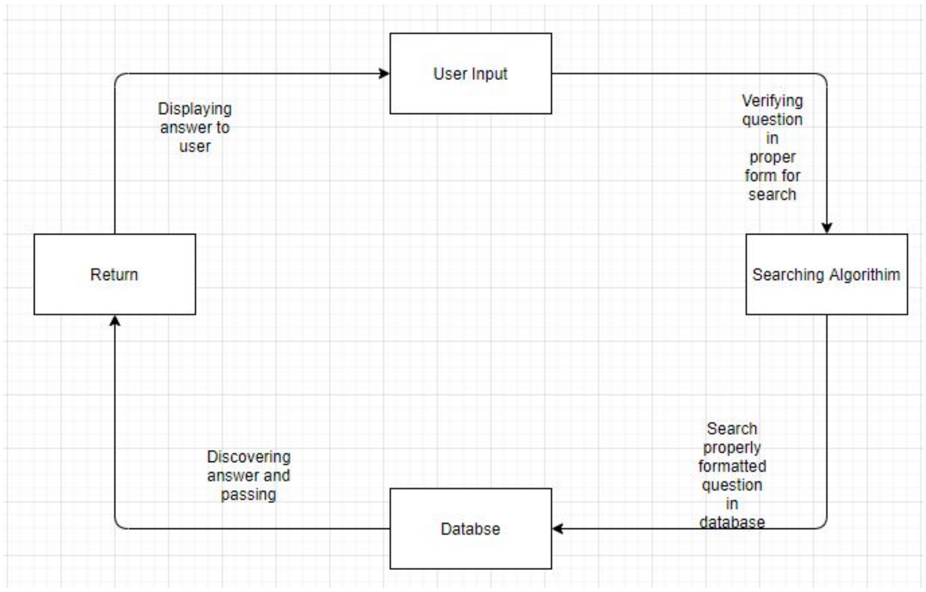

User Input
Verifying question in proper form for search
Searching Algorithim
Search properly formatted question in database
Databse
Discovering answer and passing
Return
Displaying answer to user

Figure 5: Control Specification

6.3.2 Design Constraints

The biggest constraint we will have is accepting the user's input. We will need to prompt the user for correctly formatted questions for the search to properly work. Once that is accomplished we can actively and efficiently search the database.

7. Behavioral Description

The behavioral description of a project shows the states of the machine and the actions taken to get to those states. As the functionality of most chatbots, the main behavior of the project is to handle queries and deliver an answer to the user. Once the answer is given, additional functionality will allow the user to request an example (if applicable) or request an additional explanation to said query. If the answer to the user's query cannot be found, the user will be prompted to either reword the initial question or to simply ask another question. Once the user is satisfied with their answer or the question they are asking cannot be found, they can end communication and exit the chatbot. The behavioral description, with machine states and actions, of our chatbot can be simplified into a state transition diagram (see below).

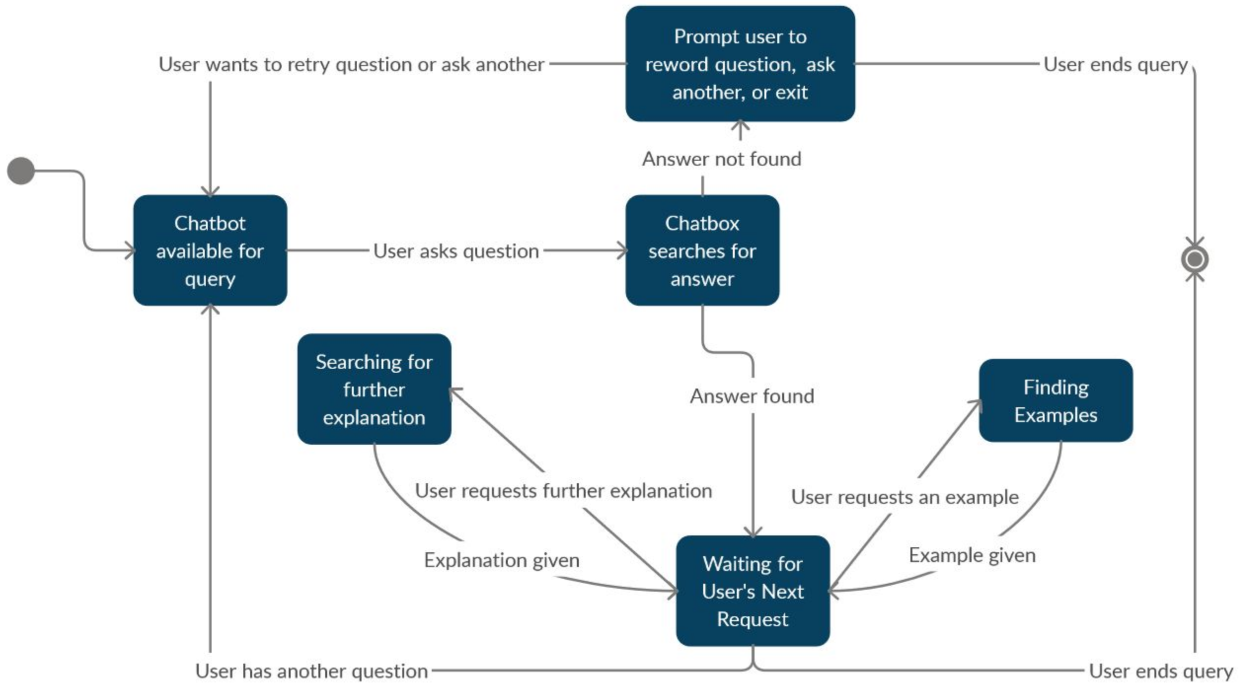

Prompt user to reword question, ask another, or exit
User wants to retry question or ask another
User ends query
Answer not found
Chatbot available for query
User asks question
Chatbox searches for answer
Searching for further explanation
Answer found
Finding Examples
User requests further explanation
User requests an example
Explanation given
Example given
Waiting for User's Next Request
User has another question
User ends query

Figure 6: State Transition Diagram

8. Use Cases

8.1 Stakeholders

A stakeholder is an individual or business that has an interest in using a given product.

There are few stakeholders for this chatbot due to it being designed for a niche user base.

Examples of stakeholders include:

- Users: (interested in a source to answer object-oriented related questions)
  - Students
  - Teacher Assistants
  - Professors
  - Tutors
- Universities (interested in a source where students can access 24/7)
- Businesses (interested in embedding the chatbot into their platform)

8.2 Actors

- Student
  - The goal is to provide students enrolled in Object Oriented Java classes with a chatbot assistant that will answer Object Oriented questions about terminology, error and coding as well as be able to provide example code snippets.
- Tutor/Teacher Assistant
  - The goal is to provide tutors with a chatbot assistant that will help them give accurate answers to common questions asked by students regarding Object Oriented questions.

8.3 Use Case Diagram

The figure below shows the interactions between the actors and use cases.

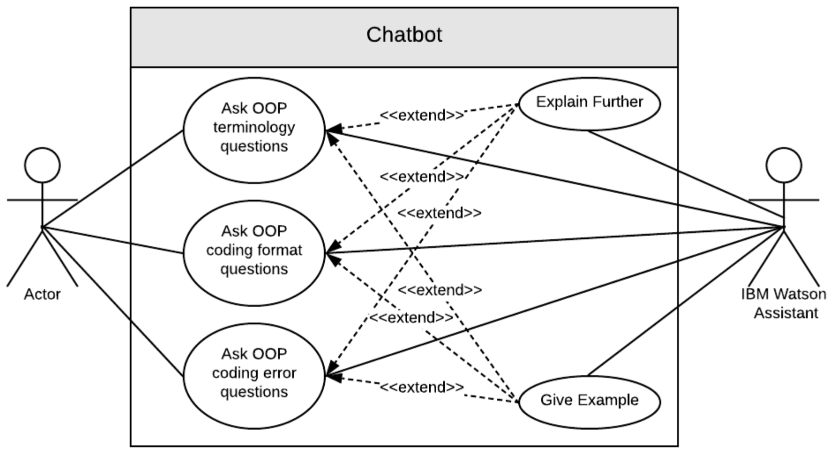

Chatbot
Ask OOP terminology questions
Ask OOP coding format questions
Ask OOP coding error questions
Explain Further
Give Example
<<extend>>
Actor
IBM Watson Assistant

Figure 7: Use Case Diagram

8.4 Use Case Description

The chart below describes each possible use case.

Use Case	**Description**
Ask OOP terminology questions	There will be various Object Oriented terminology questions that can be asked within the chatbot. The actor will be able to ask the same question in different formats and receive the same response.
Ask OOP coding format questions	The actors will be able to ask syntax questions regarding Object Oriented Design and Java code and the chatbot will provide related code snippets.
Ask OOP coding error questions	The actors will be able to input the type of Java system error found in their code and the chatbot will respond with more information about what causes the error and how to resolve the error.

Table 8: Use cases

9. Scope

9.1. System Objectives

The Object-oriented programming chatbot is able to take user input in the form of questions and process them into intents and then output a response to answer that question.

9.2. Major Software Requirements

The Object-oriented programming chatbot requires an IBM Cloud account under the Lite payment plan. This account needs access to the IBM Watson Assistant resource. The chatbot also needs to be hosted on a website.

9.3. Design Constraints/Limitations

The Lite payment plan on the IBM Cloud has limitations that impact the Object-oriented programming chatbot. The bot is limited to sending 10,000 messages per month. The bot has an analytics dashboard to use for debugging but the Lite plan restricts the storage of that dashboard to a window of 7 days. The Lite plan also restricts the amount of Dialog Skills that can be created to 5. Each Skill has 100 Dialog Nodes.

10. Data Design

10.1 Entity Relationship Diagram

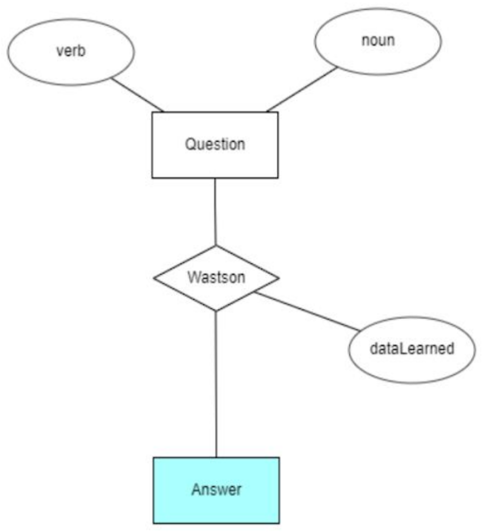


Figure 8: ER Diagram

10.2 Data Objects

The entities are just the questions and answers. The users pass a question which relates to the answer. The questions have a few attributes. The current attributes it consists of are nouns and verbs. These are key fundamentals of the question and will help develop the understanding of the question.

10.3 Relationships

The relationship is completely based on IBM Watson. We don't have a full understanding of how Watson learns. We do understand though the more data passed through

Watson the smarter it becomes. The relationship has an attribute of dataLearned. This is what we can teach Watson in the given timeframe.

11. Architectural Design

11.1. Review of Data and Control Flow

- Control: Within the system, the intents and entities are at the bottom and are controlled by the dialog nodes that define how they should work. Above that however, is the Watson Assistant, which has the most control over how the internals of the chatbot function, and uses the lower layers to generate responses. What controls the Watson Assistant will be the web interface, which the student uses to enter questions to the chatbot.
- Data: The data flow is not continuous, but happens sporadically whenever the user inputs a question. Questions are passed to the chatbot, where functional components within Watson Assistant, like dialog nodes, use the data components of intents and entities to create responses.

11.2. Derived Program Structure

The system will use a data-flow architectural style, and as such, the diagram structure below is similar to that of the data flow diagram. The questions are entered into the interface, where they are sent to the Watson Assistant and then processed using the dialog nodes. The dialog nodes use the intents and entities databases to generate a response and then send it back up the pipeline to display for the user.

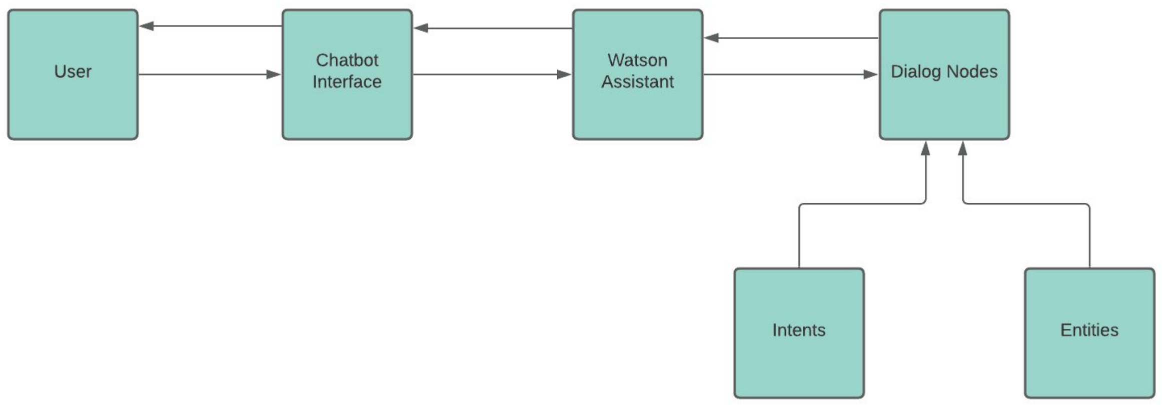


Figure 9: Derived Program Structure

12. Interface Design

Interface design expresses how information and data will flow in and out of our system. Our chatbot will be embedded onto a webpage in order for users to interact with it. Since we're using IBM's Watson Assistant to create our chatbot, there are many ways our chatbot will interact externally and internally that we are unable to fully express, as that information is unavailable to us at our level of access. With that being said, in the next following sections, the topic of interface design will be discussed generally with the information available to our team.

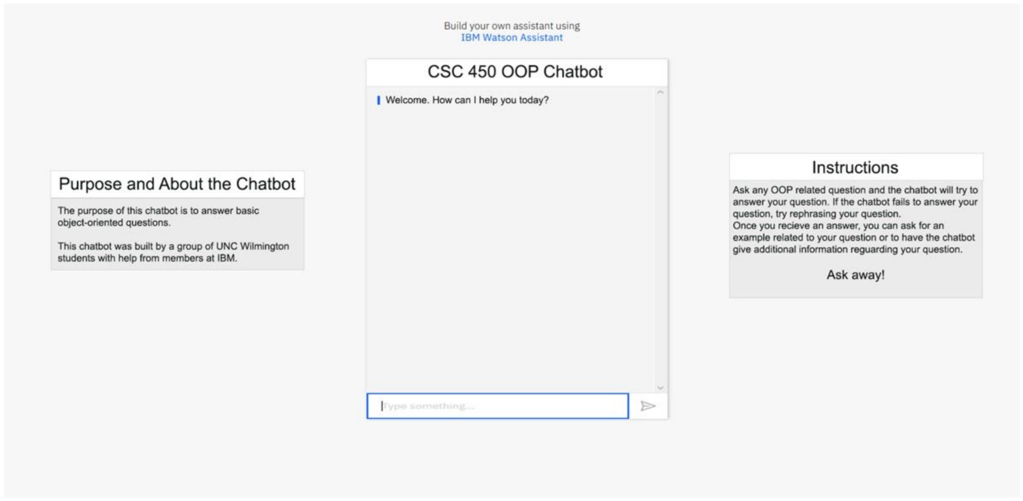
12.1. Human-Machine Interface Specification

With our chatbot being embedded on a simple webpage, the required memory and processing power falls on IBM's Watson Assistant tool rather than that of the user. The only requirements needed from the user is an internet connection in order to access the web page itself. Of course, from a hardware perspective, the user will need a keyboard and mouse to interact with the chatbot, as well as, a monitor to display the information on the screen.

12.2. Human-Machine Interface Design Rules

The webpage where our chatbot will be embedded will be the only way for users to interact with our chatbot. The webpage itself will be simple, with the main focus of the webpage being on that of the chatbot. Additionally, the webpage will include information about the chatbot, like the intended use and credit to the creators of the chatbot (including the team from IBM). Additionally, a simple explanation of how to use the chatbot and the options available to the user will help in the chance that the user gets lost or doesn't know what to do next.

Below is a simple mockup for how the webpage will look. This mockup doesn't include the final color scheme or other aesthetic elements; rather, the purpose of this mockup is to include the elements mentioned in the paragraph above and help display the foundations for the UI to be expanded upon later.



CSC 450 OOP Chatbot

Welcome. How can I help you today?

Purpose and About the Chatbot

The purpose of this chatbot is to answer basic object-oriented questions.

This chatbot was built by a group of UNC Wilmington students with help from members at IBM.

Instructions

Ask any OOP related question and the chatbot will try to answer your question. If the chatbot fails to answer your question, try rephrasing your question.
Once you recieve an answer, you can ask for an example related to your question or to have the chatbot give additional information reguarding your question.

Ask away!

Figure 10: Interface Design

12.3. Internal Interface Design

Once a user inputs a question into the chatbot, the question is sent to the Watson assistant's dialogue skill. This dialogue skill takes the user's question and breaks down it into entities. These entities are objects, terms, or keywords that provide context to intents. Intents are essentially an anticipated reason or goal the user has when asking a question. These intents are incorporated into a dialogue flow which determine how the chatbot will further interact with the user, either asking the user for additional information required to answer the user's question, or, how the answer to the user's request will be delivered back to the user.

12.4. External Interface Design

If the answer to a user's question cannot be answered internally with the use of Watson's dialogue skill, the chatbot will utilize Watson's search skill. The search skill utilizes external sources and databases to deliver an answer back to the user. We plan to utilize common computer science and object-oriented help websites and their databases, such as W3Schools, Geeks4Geeks, and other open-source documentation databases, to assist in delivering a response to a user's request that would otherwise not be answered with the dialogue skill.

13. Test Plan

13.1. What to Test

Testing the chatbot is spread between functional testing and integration testing. Functional testing is integral to understanding the scope of the chatbots current performance. This includes the chatbots capacity to evaluate and answer questions. Integration testing in the context of the chatbot is how the chatbot operates when integrated into a webpage.

13.2. When to Test

Testing occurs each and every time changes are made to the chatbot. This is in order to avoid a bug progressing through multiple iterations of the chatbot. Intents, entities, and dialog need to operate seamlessly for a pleasant user experience and frequent testing is required to ensure this.

13.3. How to Test

Functional testing is done both manually by one of the team members or automatically using the WA-Testing-Tool from IBM. The testing tool performs k-fold cross-validation on the chatbots training data. (Intents, Entities, and Dialog) The testing is able to find where the chatbot is getting confused and by what behind the scenes with a level of attention to detail not feasible by one of our team members. However while the tool may return a high cross-validation score, this does not guarantee it will translate well to live users. This is why it is important for equal scrutiny from our team members.

14. Functional Testing

14.1. Test Data Design Techniques

While testing the chatbots functionality there are five categories of design that will be evaluated. The following categories are:

1. Onboarding - Are users understanding what the chatbots purpose is and how to interact with it when the chatbot is opened?
2. Understanding - What is the chatbot able to understand?
3. Answering - How well is the chatbot responding? Are the responses relevant to the users input?
4. Navigation - How easy is it to go through the chatbot conversation?
5. Error Management - How good is the chatbot when dealing with an unrecognized input?

14.2. Test Cases

A test case will be executed for each of the five categories to determine if the chatbot meets the requirements for each one. The test case's expected result will be compared to the actual result in order to determine if the chatbot fulfills the category.

Category	Test Case	Expected Result	Actual Result
Onboarding	Open chatbot from website	Greeting from chatbot stating what type of questions it can answer	User is prompted with, "Hello, you can ask me questions about OOP"
Understanding/Answerings	*See Table 10*		
Navigation	Ask a series of questions	Answer each question with no errors	Chatbot answered each question correctly with no confusion with user input
Error Management	Ask a question unrelated to OOP	Prompt the user to try asking another question	"I didn't understand. Can you try rewording your question?"

Table 9: Test Cases

Test Case	Input	Expected Result	Actual Result
Input "definition based" question	"Can you tell me what encapsulation means?"	Definition of Encapsulation	"Encapsulation is used to hide the values or state of a structured data object inside a class, preventing unauthorized parties' direct access to them"
Input "common error message" question	"What does the error incompatible types mean?"	Explanation of error	"This error message indicates you are attempting to perform an operation on a data type that is not compatible with that operation. Here is a StackOverFlow thread that may help your understanding. *url*"

Input “oop concept” question	“What are the basic concepts of oop?”	List of 4 main concepts of OOP	“The four main concepts of OOP are Abstraction, Encapsulation, Inheritance and Polymorphism”
Input “how to use” question	“What is the use of finalize method”	Explanation of finalize method	“Finalize method helps to perform cleanup operations on the resources which are not currently used. Finalize method is protected, and it is accessible only through this class or by a derived class.”
Input “difference between terms” question	“What is the difference between early binding and late binding?”	Definition of both early and late binding	“Early binding refers to the assignment of values to variables during design time, whereas, late binding refers to the assignment of values to variables during run time”
Input “java syntax” question	“What is the syntax of an array”	Example of creating an array	“Type [] variable = {item1, item2, item3};”

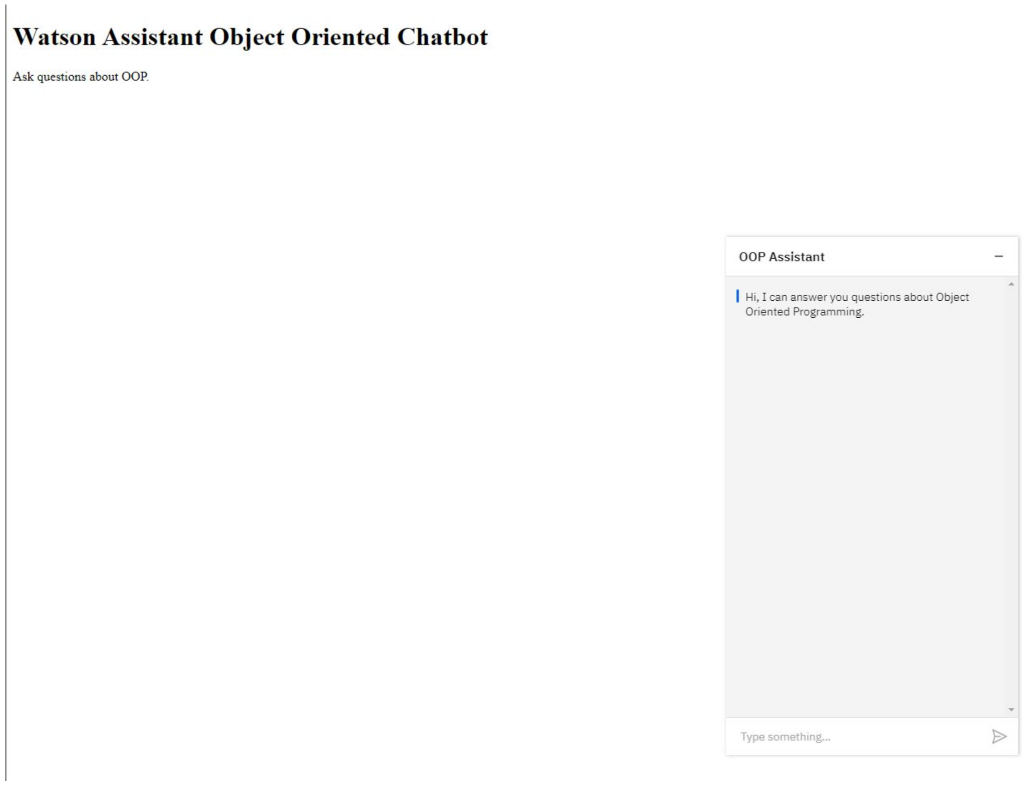
Table 10: Understanding/Answering Test Cases

After testing each category with a test case, it is concluded that the chatbot fits the requirements for each category. This shows that the chatbots functionality is working properly and is easy to use.

15. Integration Testing

15.1. Integration Strategies

Due to the relative simplicity of the integration we are using continuous integration testing. We have created a basic website with the Watson Assistant embedded into it to be able to test the interaction between the webpage and the chatbot. The website can be viewed here: http://student.uncw.edu/cad2272/Watson.html (url subject to change).


Watson Assistant Object Oriented Chatbot

Ask questions about OOP.

OOP Assistant
Hi, I can answer you questions about Object Oriented Programming.
Type something...

Figure 11: Chatbot Interface

The layout of the page is in its early stages but exhibits the basic functionality that is required. The webpage functions independently and the chatbot component functions independently on the IBM Cloud. Integrating the webpage and chatbot together allows the user to view the webpage while also asking the chatbot questions.

15.2. Test Cases

Test Case	Expected Result	Actual Result
Opening the chatbot	An icon appears on the website that can be clicked to open the chatbot	After opening the webpage a chat icon appears in the bottom right corner that can be clicked to open the chatbot
Minimizing the chatbot	A minimize button can be clicked to minimize the chatbot	While the chatbot is open you can click the minimize icon in the top right corner of the chatbot interface that minimizes the chatbot back to the chat icon
Asking the chatbot one question	The chatbot will return an answer to the question	User types "What is a class?" and is returned the answer "A class is a representation of a type of object. It is a blueprint/plan/template that describes the details of an object."
Asking the chatbot multiple questions	The chatbot will return an answer to the first question before allowing more questions to be asked which will also return answers	User types the first question and is returned an answer and then types "What are the four main concepts of OOP?" and is returned the answer "The four main concepts of OOP are Abstraction, Encapsulation, Inheritance and Polymorphism."

Table 11: Test Cases

16. Test Evaluation

16.1. Test Cases and Actual Test Results

Referring to the test cases above in 15.2 the expected actual results for most cases were similar. Opening and minimizing the chatbot work as expected. Asking the chatbot one question and multiple questions is working as expected so far. It is still a work in progress because of its limited answering ability. We expect it to be more complex in the future but currently it is still simple.

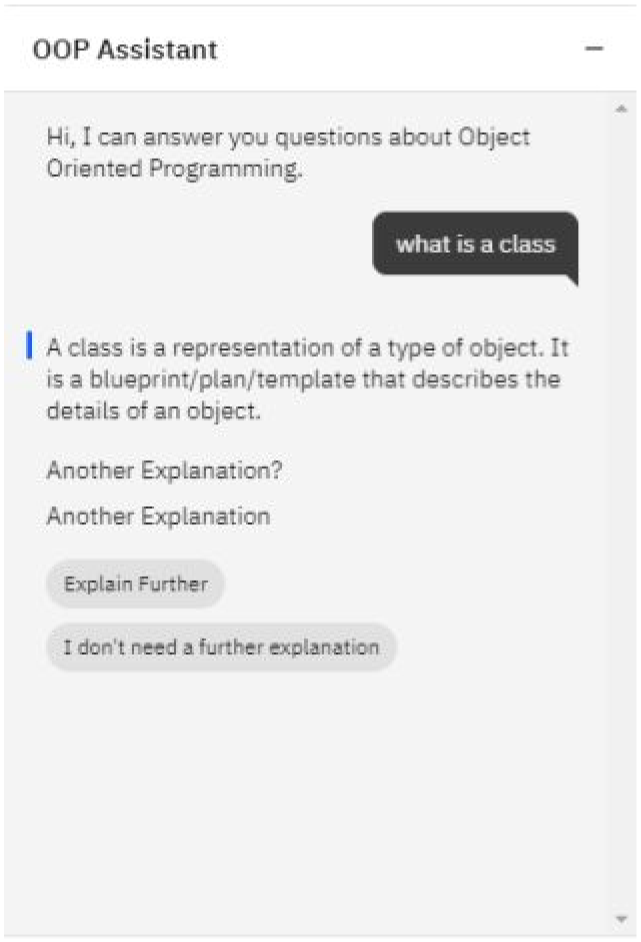
16.2. Structural Coverage Measurement

Currently all of our intents are useful and needed. They each go into teaching the assistant about object oriented programming. Theoretically our assistant will use all of the information we input. We could later find out that certain questions are rarely or never asked. We could then consider removing them. However, this is unlikely because in this case the more our assistant knows the stronger it is.

16.3. Errors Detected and Corrected

Through testing, our team found little errors with the OOP Chatbot. This is to be expected, as all coding-based issues or errors from the chatbot have been refined out by IBM before we ever got our hands on the Watson Assistant. The only issues that could arise would be from our intent, entity, and dialogue inputs, as well as, the structuring of our project's chatbot itself.

One issue that arose was a simple repeated phrase once the chatbot delivered the answer to the user's question [see figure below]. Once identified, this error was very easy to remedy and the issue was resolved.


OOP Assistant

Hi, I can answer you questions about Object Oriented Programming.

what is a class

A class is a representation of a type of object. It is a blueprint/plan/template that describes the details of an object.

Another Explanation?

Another Explanation

Figure 12: Chatbot Interface

One other issue our team had was with the testing process itself. This issue came about when using the WA-Testing-Tool from IBM that was mentioned above in section 13.3. The problem itself is the product of having little experience with the testing tool and not fully understanding how to fix or adjust the tools available to fit our needs. We couldn’t utilize many of the tools the testing script gave to us, which could’ve been extremely helpful to decrease testing time and find out, in more detail, how to improve our chatbot moving forward. Although we couldn’t remedy this exact issue, we did move to manually testing our chatbot, which worked perfectly fine. In the future, we plan on discussing this issue with the IBM team in order to be able to utilize the many tools the testing script gives to us.

16.4. Summary of Testing Experience

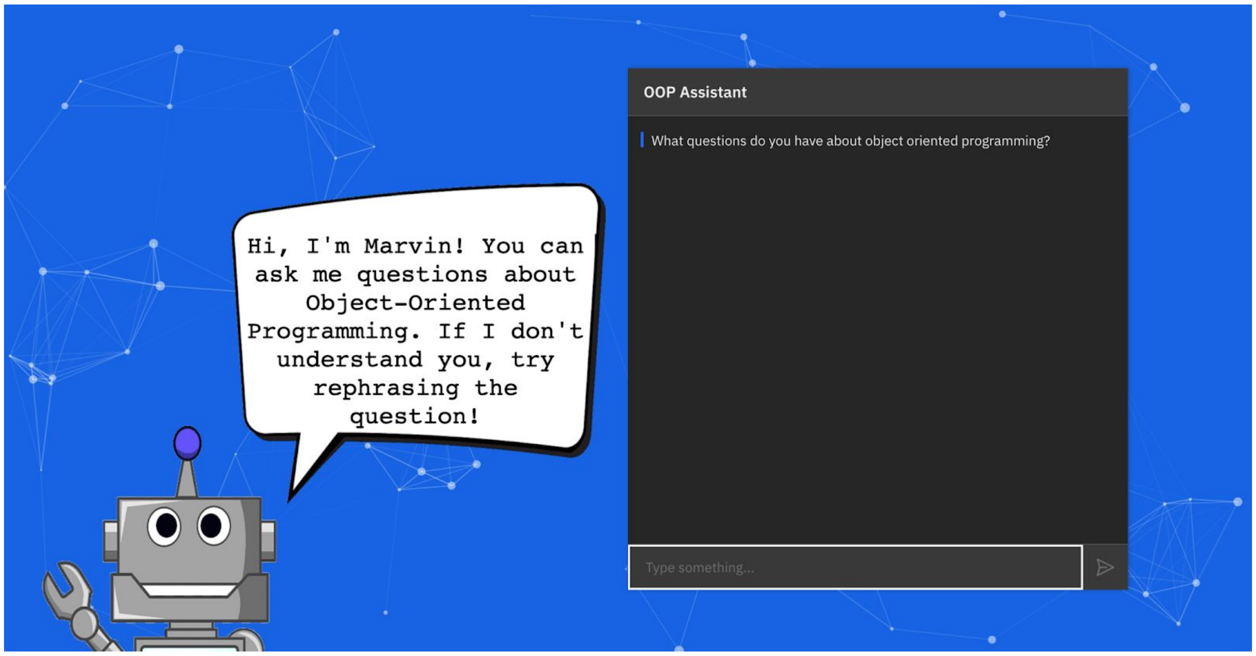
The testing experience, as a whole, went extremely well. As mentioned in the section above, our team ran into little errors and our chatbot functioned as expected. Even the issues we did run into either resulted in a quick fix or finding alternative ways to reach our goal. Through functional testing, we were able to better understand the capacity of our chatbot in regards to evaluating and answering user's questions. Through integration testing, we saw how the chatbot will interact with the user and how exactly our integrated chatbot operates on a webpage.

Generally speaking, our chatbot is meeting our expectations and operating as intended. With this being said, as the project continues down its timeline, our team will continue to add more and more to the chatbot. This involves adding more intents, entities, and dialogue nodes to add more functionality and depth which will expand the usefulness of the chatbot. These additions, of course, will require further testing to make sure the chatbot functions correctly and meets our expectations.

17. Summary

17.1. Project Description

We have created a chatbot using IBM Watson Assistant to interpret and answer object-oriented programming questions. Students taking an object-oriented class can use the chatbot to help answer common questions about the subject. The chatbot will return to the end-user a well explained response to questions regarding the concepts that make up object-oriented programming. The chatbot has the ability to define the basic concepts of object-oriented programming as well as give examples using Java.


Hi, I'm Marvin! You can ask me questions about Object-Oriented Programming. If I don't understand you, try rephrasing the question!
OOP Assistant
Type something...

What questions do you have about object oriented programming?

Figure 13: Final Product

17.2. Programming Experience

As the bulk of the work is done using IBM's interface for building the chatbot, there wasn't very much programming required. The intents and entities would be entered and then the dialog node tree could be created. However, in order to make a web interface for the user to interact with the chatbot, programming in HTML and JavaScript was required. A snippet of code from the IBM website was used to embed the chatbot into the webpage. From there it was formatted to create the UI and layout of the page.

18. Codebase

- Github Repository
  - https://github.com/hannahmre/OOP-Chatbot

19. Appendices

- "Getting started with Watson Assistant"
  - A helpful resource for learning how Watson works and how to use it.
  - https://cloud.ibm.com/docs/assistant?topic=assistant-getting-started
- A similar project that has largely served as the inspiration for this one. The "Jill Watson" teaching assistant developed by Ashok K. Goel at the Georgia Institute of Technology.
  - www.youtube.com/watch?v=WbCguICyfTA
- Overview of conversational AI and why it is an important tool for Lakisha Hall of IBM.

- https://www.ibm.com/blogs/watson/2018/08/beyond-the-chatbot-why-ai-in-customer-service-will-be-crucial-to-your-business/

19.1 Agenda and Meeting Minutes

- 09/03/2020 (9:30 - 10:00) - Initial team Zoom meeting
    - Discussing initial ideas for the project and setting up future meetings.
- 09/07/2020 (5 - 5:30) - Discussing topic
    - Discussing project topic before meeting with employees from IBM.
- 09/08/2020 (9:30 - 10:10) - First Meeting with Ryan Brink and Rebecca James from IBM.
    - Discussing the project and how to use Watson. Talked about the topic for the chatbot and plans for the semester.
- 09/24/2020 (9:30 - 10:15) Discussing Deliverable #2
    - Figuring out plan for deliverable #2 and planning out the project and requirements.
- 09/29/2020 (9:30 - 10:10) Discussing Deliverable #3
    - Talking about the plan for the project, and discussing setting up a meeting with the IBM employees and getting student questions from Mrs. White.
- 09/13/2020 (2:30 - 3:30) Meeting again with Ryan and Rebecca from IBM
    - Discussing where we are at and plans for the project.
- 09/14/2020 (11:30 - 11:50) Discussing class presentation and deliverable #3.