Stamford Chat Project Documentation

Project Title: Stamford Chatbot

Course Code: ITE222

Course Name: Programming II

Student (1) Name: Lynn Thant

Student (1) ID: 2406120001

Student (2) Name: Zwe Pyi Phyo

Student (2) ID: 2406120005

Course: ITE:222

Instructor: Dr Nay Myo Sandar

Date: 4 Jun 2025

Table of Contents

Introduction

In today's digitally driven educational landscape, universities are increasingly adopting chatbot technology that helps many of its students complete their respective tasks. Stamford Chatbot is a Java-based chatbot application that aims to assist students at Stamford University by providing quick and easy access to essential information that is related to campus. It is able to offer instant responses to common queries such as course information, the academic calendar, map of a campus, contact details of staff and calculate their budgets. This report outlines the design, functionality, and technical implementation of Stamford Chatbot.

Objectives

The aim of this project is to develop a Java-based chat bot system that automates student support services within a university setting. The chat bot aims to provide necessary information such as course information, academic calendar, map of a campus, contact of respective personnel in the university and a tool that allows students to calculate their finances. The program should be easy to use as a user and consist of maintainable codes that allow smooth collaboration between developers.

Tools and Technologies Used

Category	Tools / Technologies
Programming Language	Java (JDK) – for core development and execution
Development Environment	Eclipse IDE – for writing, debugging, and managing Java code.
Execution Interface	Command Line / Terminal – for compiling and running the application
Java Packages Used	- java.util.Scanner– for reading user input - java.lang – String manipulation.

System Overview

The Stamford Chatbot system is a console program built on Java that helps users with a variety of university-related questions. It offers features like searching for course information, browsing the academic calendar, navigating the campus map, and finding staff contact information. The system is modular with separate classes handling each specific task. Users interact with the chatbot through a simple text interface, entering commands like "help," "course info," or "exit" to navigate the options. The program isn't case sensitive, allowing users to enter correct commands as they prefer.

The system uses a layered and organized structure to handle user commands. When a user types a command, the main class checks the input and passes it to the system class. This system class then creates and uses the right feature class to do the task. For example, if the user wants campus map details, the system calls showMap(), which creates a Stamford_Chatbot_CampusMap object to handle that request. This layered approach ensures proper separation of concerns, with each class handling specific responsibilities. The benefit of developing with this design pattern allows centralized control over a program that standardize request handling and managing error. Additionally, when adding new features, the existing work will be not be disrupted, encouraging scalability to the application. Overall, the chatbot achieves to be a clean, organized structure while maintaining flexibility for future enhancements.

Spiral Modal

The development of this project follow the Spiral Modal that provides a systematic and iterative approach to a software development.

Requirement

In this phase we identified the need for a chat bot to assist Stamford University students and allows them to access important campus-related information quickly. Main function of the chat bot includes:

- Providing Course Information
- Displaying the Academic Calendar

- Offering Campus Map details
- Providing contact details for staff
- Calculating student budgets.

During this phase, we also considered potential challenges, such as user input errors and the complexity of the logic, aiming to mitigate these issues early on to enhance user experience. The decision was made to use a command-line interface due to its simplicity and our current knowledge limitations with graphical user interfaces.

Design

Based on the requirements, we planned the architecture of the Stamford Chatbot as a console program. The system was designed to be modular, with separate classes handling specific tasks.

- Stamford_Chatbot.java (Main class) : designed as an entry point
- Stamford_Chatbot_System.java: designed as a central control class.
- Stamford_Chatbot_CourseFinder.java: Designed to provide detailed course information.
- Stamford_Chatbot_Schedule.java: Designed to display academic calendar.
- Stamford_Chatbot_Contact.java: Design to provide contact information.
- Stamford_Chatbot_CampusMap.java: Designed to display location information.
- Stamford_Chatbot_Budget.java: Designed to help students manage their finance.

Develop

In the product development phase, the chatbot was built according to the planned design. This involved coding the main chatbot structure first, followed by the development of each feature module (course information, academic calendar, campus map, contact details, and budget calculation) incrementally.

- Java code was written for each class and their respective methods, implementing the logic for user interaction, data processing, and output display.
- Looping constructs like while loops were used to maintain program execution , and for loops were used for iterations, such as calendar generation.
- Conditional statements (if/else if, nested if) were extensively used for menu navigation and handling hierarchical selections.
- Input and output were managed using java.util.Scanner. Object instantiation was used extensively to access functionality across classes, for instance,
- Stamford_Chatbot_System creating objects of specialized classes to handle specific requests.
- Syntax errors were addressed during this coding phase.

Test

The final phase involved rigorous review and evaluation to ensure the chatbot met all project requirements and functioned as expected.

- Manual Testing: The program was run multiple times with various inputs to check the responsiveness and accuracy of each module.
- Input Testing: Different inputs were tested to verify how well the system responded, including handling valid and potentially invalid user entries.
- Output Testing: The program's outputs were checked for correctness and clarity, ensuring messages were informative and summaries were accurate.
- Code Quality Review: The overall structure and design were reviewed to confirm that the code was clean, modular, and easy to understand, contributing to the chatbot's

reliability and user-friendliness. Small bugs were identified and fixed, and message clarity was improved during this phase.

Class Description

Stamford_Chatbot.java (Main Class)

The Stamford_Chatbot.java class serves as the entry point for the Stamford Chatbot system. It acts as a bootstrapper that initializes the application and manages user input. The class utilizes a Scanner package to read user commands and while-loop to keep the program running until the user chooses to exit. Furthermore, the class consists of a main method, and it creates an instance of the Stamford_Chatbot_System class, which provides access to other specialized classes. Based on the user's input, conditional checks, if-else statements determine the appropriate action by invoking the corresponding method from the system class to handle the request.

Method	Purpose	Example Output
main(String[] args)	Entry point that initializes and runs the chatbot system	(No direct output - launches system)
Key Feature: Act as an entry point and instantiate system class.		

Stamford_Chatbot_System.java

The Stamford_Chatbot_System.java function as a central control class. The class consists of a constructor that display a welcoming message and several methods that invoke an instance of a specialized tasks.

Method	Purpose	Example Output
Constructor	Initializes the chatbot system and displays welcome message	"STAMFORD CHATBOT" Type 'help' for options"
Help()	Displays the main help menu	"1. Course Info 2. Academic Calendar

		3. Campus Map 4. Contact Staff 5. Exit"
findCourse()	Launches course information module	"What area are you interested in? - Computing - Art - Business"
showSchedule()	Starts the academic calendar system	"Enter the year (2025 or 2026) to view the academic calendar…"
showMap()	Initiates campus location services	“Where? (Library/Gym/Cafeteria):”
contactStaff()	Provides department contact information	"Department: (Admissions/IT/Finance):"
Key Feature: Act as a central command between main class and module classes		

Stamford_Chatbot_CourseFinder.java

The Stamford_Chatbot_CourseFinder.java class is a specialized class designed to provide detailed course information based on user queries. It interacts with users through a structured menu system, guiding them to select an academic area and then a specific subject within that area. Its findCourse() is a main method that orchestrate the user inteaction and logic behind the course lookup. Lastly, its modular design ensures clean separation from other chatbot functionalities, promoting maintainability and scalability.

Method	Purpose	Example Output
findCourse()	Interacts with the user via console to suggest relevant courses based on input area and subject	User enters “Computing” → then “network” → gets info about ITE 554: Computer Networks.
Key Feature: The Stamford_Chatbot_CourseFinder class interactively guides users to discover course information based on their area of interest and subject preferences in Computing, Art, or Business.		

Stamford_Chatbot_Schedule.java

The Stamford_Chatbot_Schedule class is one of core components of the Stamford Chatbot system, designed to display academic calendars and key dates for specified years (2025 or 2026). It generates a monthly calendar view with annotations for important events and supports user interaction through a console interface. The class consists of constructor and several methods that allow users to select year, view a formatted calendar, check important dates and get detailed event summaries. It is built with a 6x7 2D array to represent weeks and days.

Method	Purpose	Example Output
Constructor	Starts user input loop to display the calendar for year 2025 or 2026.	"Enter the year (2025 or 2026) to view the academic calendar (or 0 to exit):"
displayCalendar(int year)	Displays the full 12-month calendar for the given year, marking holidays and exam dates.	Calendar grid with symbols (e.g., 15X, 01*) + summary at the end.
resetCalendarGrid()	Resets the 2D calendar array before rendering each month's calendar.	Internal operation; clears the grid. No direct output.
printEventSummary(int year)	Prints a summary of holidays and fake exam dates for the selected year.	Shows: 2025-01-01 - New Year's Day (Holiday), etc.
isLeapYear(int year)	Checks whether a given year is a leap year.	isLeapYear(2024) → true, isLeapYear(2025) → false
getDaysInMonth(int year, int month)	Returns the number of days in a specific month of a year.	getDaysInMonth(2025, 2) → 28, getDaysInMonth(2024, 2) → 29

getMonthName(int month)	Converts a numeric month to its full name.	getMonthName(3) → "March"
getDayOfWeek(int year, int month, int day)	Calculates day of the week (0 = Sunday, ..., 6 = Saturday).	getDayOfWeek(2025, 1, 1) → 3 (Wednesday)
isFakeHoliday(int year, int month, int day)	Determines if a given date is marked as a fake holiday.	isFakeHoliday(2025, 2, 12) → true (Fake Winter Break)
isFakeExamDate(int year, int month, int day)	Determines if a given date falls in a fake exam period.	isFakeExamDate(2025, 5, 18) → true (Midterm)
Key features: The Stamford_Chatbot_Schedule class dynamically generates and displays a year-long academic calendar for 2025 or 2026, highlighting holidays and exam periods within a formatted monthly grid.		

Stamford_Chatbot_Contact.java

The Stamford_Chatbot_Contact class is a specialized module within the Stamford Chatbot system that handles contact information requests for university departments. It provides users with email addresses and phone numbers for key departments such as Admissions, IT and Finance departments through an interactive console menu.

Method	Purpose	Example Output
Constructor Stamford_Chatbot_Contact()	Initializes the contact system and prompts the user for a department choice.	"Department: (Admissions/IT/Finance) : "
displayAdmissions()	Shows Admissions contact info.	"✉ admission@stamford.edu 📞 02-765-4321 (9AM-5PM)"

displayIT()	Shows IT Helpdesk contact info.	"✉ it-support@stamford.edu 📞 02-123-4567 (9AM-5PM)"
displayFinance()	Shows Finance Office contact info.	"✉ cashier_rm9@stamford.edu 📞 02-769-4000 (8:30AM-5:30PM)"
Key Feature: The Stamford_Chatbot_Contact class allows users to quickly retrieve contact information for specific university departments such as Admissions, IT, and Finance.		

Stamford_Chatbot_CampusMap.java

The Stamford_Chatbot_CampusMap class is also one of the specialized module within the Stamford Chatbot system that provides location-based information about key campus facilities. It offers users quick access to details such as building locations, operating hours, and amenities for predefined campus spots.

Method	Purpose	Example Output
Constructor Stamford_Chatbot_CampusMap()	Initializes the campus map system and prompts the user for a location of choice.	"Where? (Library/Gym/Cafeteria): "
displayLibrary()	Shows Library location and hours.	"Building A, 1st Floor Open 7AM – 7PM"
displayGym()	Informs users about gym availability (currently not at Rama 9 campus).	"Sorry, Not Available in Rama 9 Campus"
displayCafeteria()	Displays Cafeteria details, including today's menu.	"Building B, 1st Floor Open 7AM-7PM Today's Menu: Pad Thai, Salad"
Key Feature: The Stamford_Chatbot_CampusMap class provides users with location and availability details for key campus facilities such as the library, gym, and cafeteria based on their input.		

Student_Chatbot_Budget.java

The Student_Chatbot_Budget class is an interactive Java tool that helps students understand their monthly financial situation. Its key methods focus on collecting data, calculating totals, and clearly presenting the results.

Method	Purpose	Example Output
Student_Chatbot_Budget()	Initializes chatbot and runs the budget tool	Runs the budget interaction
StudentBudgetCalculator()	Initializes calculator with income	Creates a calculator object
setTuitionFees(double)	Set tuition fee value	void (no output)
setAccommodation(double)	Set accommodation expense	void (no output)
setFood(double)	Set food expense	void (no output)
setTransportation(double)	Set transportation cost	void (no output)
setPersonalExpenses(double)	Set personal expenses	void (no output)
setOtherExpenses(double)	Set miscellaneous/other expenses	void (no output)
calculateRemainingBudget()	Perform remaining budget calculation	double (e.g., 450.00)
displayBudgetSummary()	Display a breakdown of income and expenses	Prints budget summary
Key Feature: It provides an interactive console-based tool for students to input their income and expenses, calculate their remaining budget, and display a detailed financial summary.		

Core Concepts Demonstrated

1. Loops

while loop: Used in Stamford_Chatbot.java to keep the program running until the user exits.

for loop: In Stamford_Chatbot_Schedule.java to iterate through months/days for calendar generation.

if/else if: For menu navigation (e.g., checking user input for "course info" or "calendar").

nested if: In Stamford_Chatbot_CourseFinder.java to handle hierarchical selections (area → subject).

2. Constructors

Constructors: Each class initializes its own state.

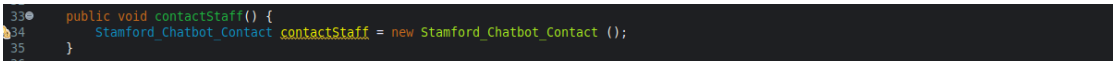
Method Overloading: Not explicitly used.

3. Inheritance

Not Applied: The project uses composition instead of inheritance.

4. Object Instantiation

Used extensively to access functionality



5. this and Dot Operator

This: Used in StudentBudgetCalculator to distinguish instance variables from parameters.

Dot Operator: Frequently used to access methods (eg. chatBot.findCourse()).

6. Java Packages

Java.util.Scanner: For user input in all interactive classes.

7. Encapsulation

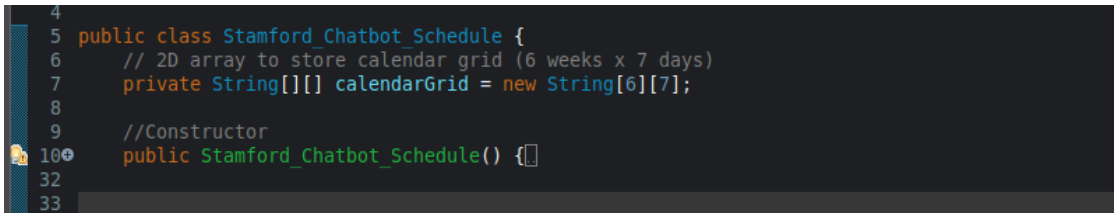
Private Fields: Used in StudentBudgetCalculator class.

Getters/Setters: Implemented in StudentBudgetCalculator for all expense categories.

8. Object Interaction

Collaboration: Stamford_Chatbot_System.java coordinates interactions between the main class and specalized classes (e.g., CourseFinder.java, CampusMap.java).

9. 2D Arrays



6x7: 6x7 2D Array is implemented in Calendar Logic to present weeks and days.

10. Static vs. Non-Static Methods

Static: Helper methods like getDayOfWeek() in Stamford_Chatbot_Schedule.

Non-Static: Most methods (e.g., findCourse(), displayLibrary()) are instance-based.

11. Commenting & Naming

Comments: Used to explain complex logic.

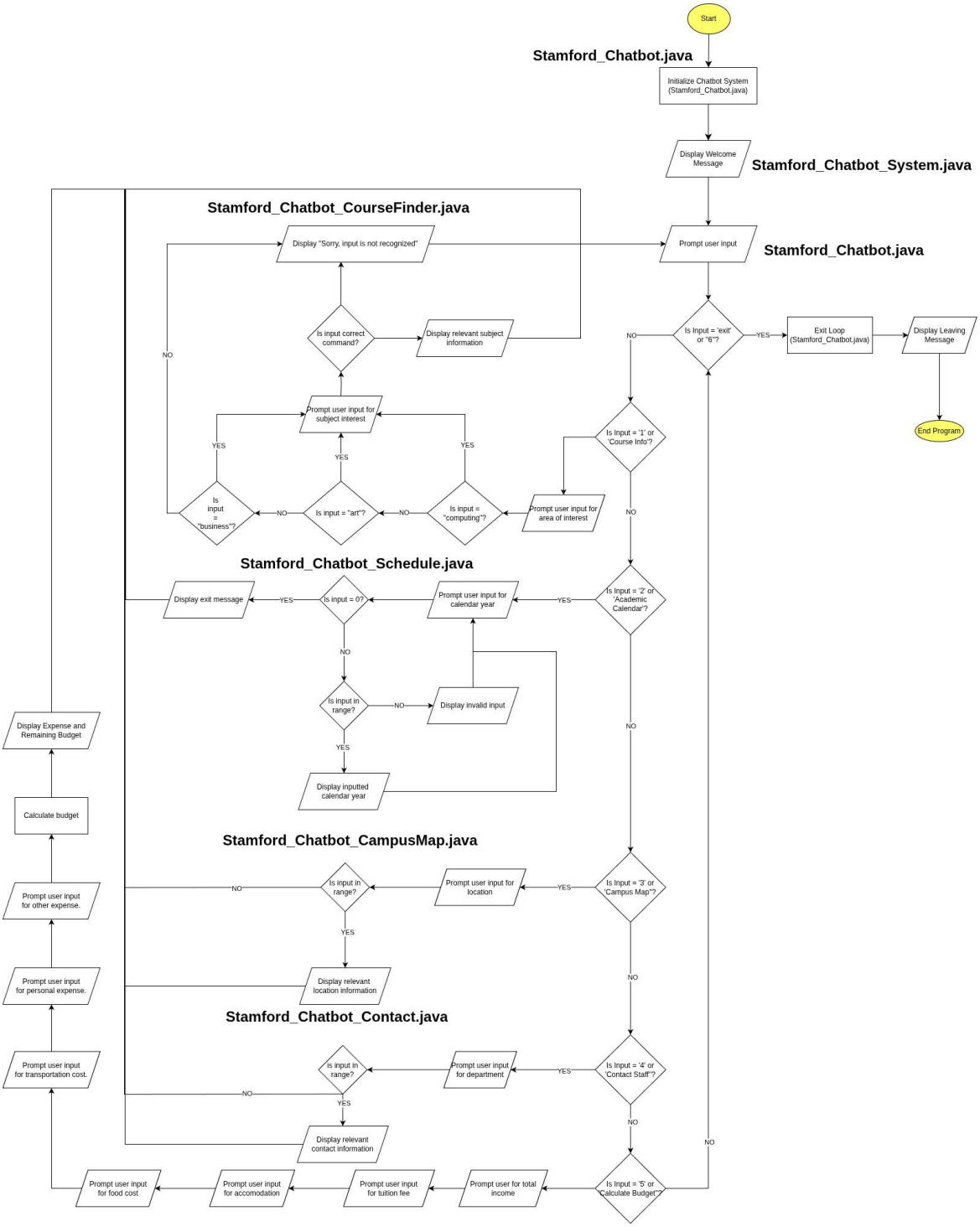
Naming: Follows Java conventions (e.g., displayCafeteria(), isFakeExamDate).

12. Access Specifiers

public: Methods called across classes (e.g., showSchedule()).

private: Fields in StudentBudgetCalculator and helper methods.

Flowchart


Start
Stamford_Chatbot.java
Initialize Chatbot System (Stamford_Chatbot.java)
Display Welcome Message
Stamford_Chatbot_System.java
Stamford_Chatbot_CourseFinder.java
Display "Sorry, input is not recognized"
Prompt user input
Stamford_Chatbot.java
Is input correct command?
Display relevant subject information
Is Input = 'exit' or "6"?
Exit Loop (Stamford_Chatbot.java)
Display Leaving Message
End Program
Prompt user input for subject interest
Is Input = '1' or 'Course Info'?
Is input = "business"?
Is input = "art"?
Is input = "computing"?
Prompt user input for area of interest
Stamford_Chatbot_Schedule.java
Display exit message
Is input = 0?
Prompt user input for calendar year
Is Input = '2' or 'Academic Calendar'?
Is input in range?
Display invalid input
Display inputted calendar year
Display Expense and Remaining Budget
Calculate budget
Stamford_Chatbot_CampusMap.java
Prompt user input for other expense.
Is input in range?
Prompt user input for location
Is Input = '3' or 'Campus Map"?
Prompt user input for personal expense.
Display relevant location information
Stamford_Chatbot_Contact.java
Prompt user input for transportation cost.
is input in range?
Prompt user input for department
Is Input = '4' or 'Contact Staff'?
Display relevant contact information
Prompt user input for food cost
Prompt user input for accomodation
Prompt user input for tuition fee
Prompt user for total income
Is Input = '5' or 'Calculate Budget"?
YES
NO

Algorithm

Step 1: Start

Step 2: Create an instance of Stamford_Chatbot_System

Step 3: Display Welcome Message.

Step 4: Prompt user for input

Step 5: Read the input using a Scanner.

Step 6: Check input against commands (help, course info, academic calendar, campus map, contact staff and exit).

Step 6.1: If user enter course info.

Step 6.1.1: Prompt user to select an area of study.

Step 6.1.2: Read user's area of choice.

Step 6.1.2: Prompt for a subject keyword.

Step 6.1.3: If Match, display course details.

Step 6.1.4 Return to main loop (Step 4)

Step 6.2: If user enter academic calendar

Step 6.2.1: Prompt user for a year

Step 6.2.2: If year is valid, print calendar grind.

Step 6.2.3: Else if year = 0, return to the main loop (Step 4)

Step 6.2.4: Else print "Invalid Year"

Step 6.3: If user enter campus map

Step 6.3.1: Prompt user for location (Library, Gym, Cafeteria).

Step 6.3.2: If valid location, display appropriate result.

Step 6.3.3: Return to main loop (Step 4)

Step 6.4: If user enter contact staff

Step 6.4.1: Prompt user for department

Step 6.4.2: Check department, if correct → display appropriate result.

Step 6.4.3: Return to main loop (Step 4)

Step 6.5: If user enter calculate budget

Step 6.5.1: Prompt user for total income and wait for input.

Step 6.5.2: Prompt user for tuition fee and wait for input.

Step 6.5.3: Prompt user for accommodation and wait for input.

Step 6.5.4: Prompt user for estimated food cost and wait for input.

Step 6.5.5: Prompt user for estimated transportation cost.

Step 6.5.6: Prompt user for personal expenses and wait for input.

Step 6.5.7: Prompt user for other expenses and wait for input.

Step 6.5.8: Calculate Expense.

Step 6.5.9: Display expenses and remaining budget.

Step 6.5.10: Return to main loop (Step 4)

Step 7: Repeat from Step 4 until “exit” is entered.

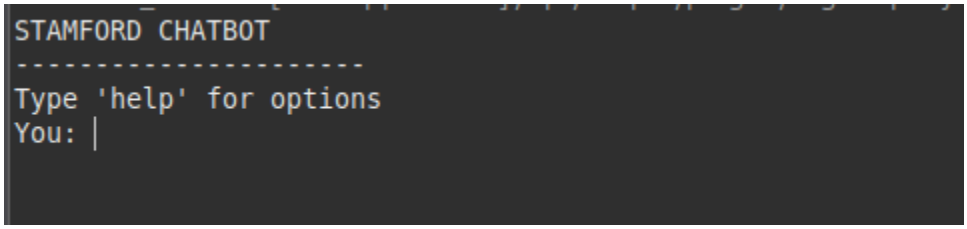
Step 8: Break While Loop

Step 9: Close Scanner.

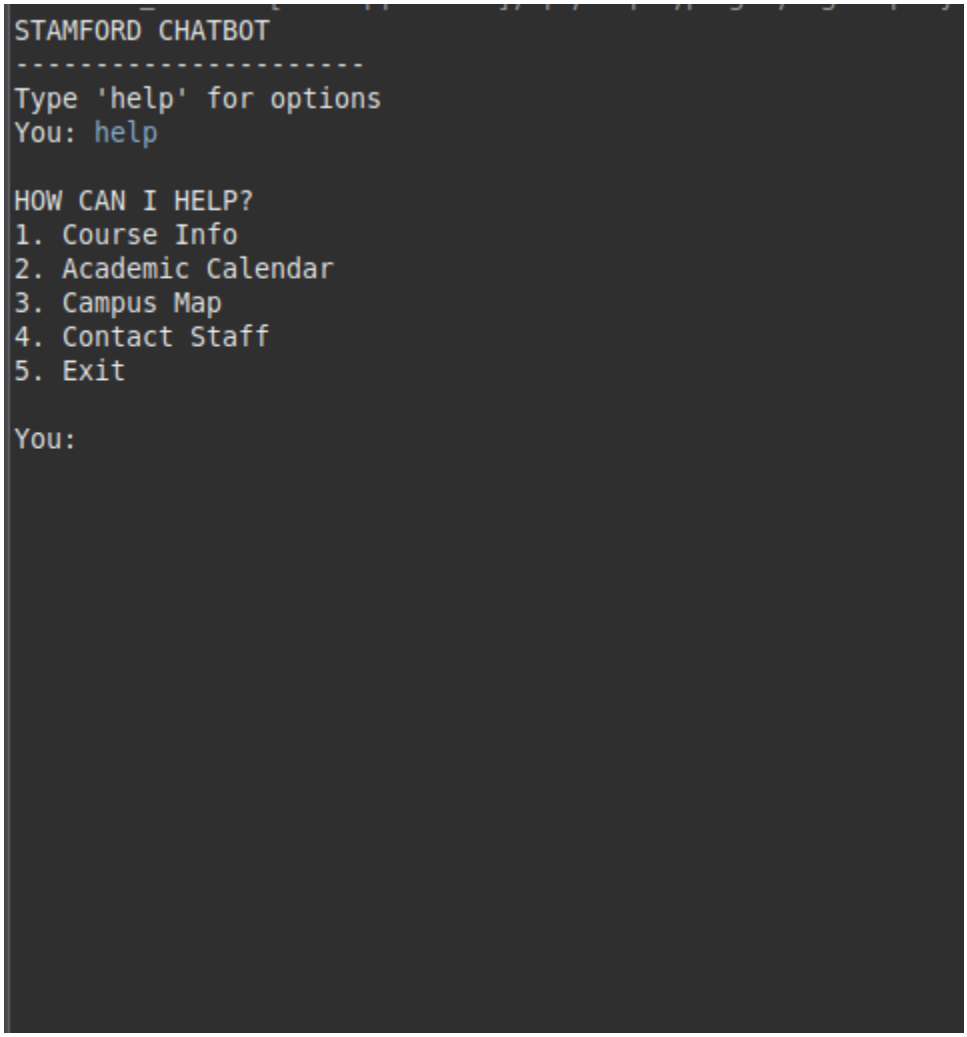
Step 10: Display “Bye! See you again”

Screenshots / Sample Outputs

1. Start



2. Users enter ‘help’



3. Users enter ‘course info’.

```
STAMFORD CHATBOT
---------------------
Type 'help' for options
[main]You: help

HOW CAN I HELP?
1. Course Info
2. Academic Calendar
3. Campus Map
4. Contact Staff
5. Calculate Budget
6. Exit

[main]You: Course Info

Welcome to the Course Information System!
What area are you interested in?
1. Computing
2. Art
3. Business
Enter your area of interest:
```

3.1 User chose ‘computing’ as an area of interest.

```
5. Calculate Budget
6. Exit

[main]You: Course Info

Welcome to the Course Information System!
What area are you interested in?
1. Computing
2. Art
3. Business
Enter your area of interest: Computing

Great! You're interested in Computing.
What kind of subject within Computing are you inte

1. Database
2. System Analysis
3. Network
4. Data and Algorithm

Enter a keyword related to the subject you like:
```

3.2 User chose ‘Database’ as a subject.

```
Welcome to the Course Information System!
What area are you interested in?
1. Computing
2. Art
3. Business
Enter your area of interest: Computing

Great! You're interested in Computing.
What kind of subject within Computing are you interested in?

1. Database
2. System Analysis
3. Network
4. Data and Algorithm

Enter a keyword related to the subject you like: Database

Information for: ITE 101 database
ITE 101: Introduction to Database Systems - Covers fundamental database concepts, relational models, and SQL.

[main]You:
```

4. Users enter ‘Academic Calendar’.

```
You: Academic Calendar

Enter the year (2025 or 2026) to view the academic calendar and holidays (or 0 to exit):
```

4.1 User enters year ‘2025’.

```
Enter the year (2025 or 2026) to view the academic calendar and holidays (or 0 to exit): 2025

--- Academic Calendar for 2025 ---

----- January 2025 -----
Sun Mon Tue Wed Thu Fri Sat
             1*  2   3   4
  5   6   7   8   9  10  11
 12  13  14  15  16  17  18
 19  20  21  22  23  24  25
 26  27  28  29  30  31

----- February 2025 -----
Sun Mon Tue Wed Thu Fri Sat
                         1
  2   3   4   5   6   7   8
  9  10* 11* 12* 13* 14* 15*
 16* 17* 18  19  20  21  22
 23  24  25  26  27  28

----- March 2025 -----
Sun Mon Tue Wed Thu Fri Sat
                         1
  2   3   4   5   6   7   8
  9  10  11  12  13  14  15
 16  17  18  19  20  21  22
 23  24  25  26  27  28  29
 30  31

----- April 2025 -----
Sun Mon Tue Wed Thu Fri Sat
          1   2   3   4   5
  6   7   8   9  10  11  12
 13* 14* 15* 16* 17* 18* 19*
 20* 21  22  23  24  25  26
 27  28  29  30

----- May 2025 -----
Sun Mon Tue Wed Thu Fri Sat
                1   2   3
  4   5   6   7   8   9  10
 11  12  13  14  15X 16X 17X
 18X 19X 20X 21X 22X 23  24
 25  26  27  28  29  30  31
```

```
----- June 2025 -----
Sun Mon Tue Wed Thu Fri Sat
  1   2   3   4   5   6   7
  8   9  10  11  12  13  14
 15  16  17  18  19  20  21
 22  23  24  25  26  27  28
 29  30

----- July 2025 -----
Sun Mon Tue Wed Thu Fri Sat
          1   2   3   4*  5*
  6*  7*  8*  9* 10* 11* 12
 13  14  15  16  17  18  19
 20  21  22  23  24  25  26
 27  28  29  30  31

----- August 2025 -----
Sun Mon Tue Wed Thu Fri Sat
                      1   2
  3   4   5   6   7   8   9
 10  11  12  13  14  15  16
 17  18  19  20  21  22  23
 24  25  26  27  28  29  30
 31

----- September 2025 -----
Sun Mon Tue Wed Thu Fri Sat
      1   2   3   4   5   6
  7   8   9  10  11  12  13
 14  15  16  17  18  19  20
 21  22  23  24  25  26  27
 28  29  30

----- October 2025 -----
Sun Mon Tue Wed Thu Fri Sat
              1   2   3   4
  5   6   7   8   9  10  11
 12  13  14  15  16  17  18
 19  20  21  22  23  24  25
 26  27  28  29  30  31
```

```
----- November 2025 -----
Sun Mon Tue Wed Thu Fri Sat
                      1
 2   3   4   5   6   7   8
 9  10  11  12  13  14  15
16  17  18  19  20X 21X 22X
23X 24X 25X 26X 27X 28  29
30

----- December 2025 -----
Sun Mon Tue Wed Thu Fri Sat
     1   2   3   4   5   6
 7   8   9  10  11  12  13
14  15  16  17  18  19  20
21  22  23  24  25* 26  27
28  29  30  31


* indicates a holiday.
X indicates a exam date.
--- End of Calendar for 2025 ---

--- Holidays and Exam Dates for 2025 ---
2025-01-01 - New Year's Day (Holiday)
2025-02-10 to 2025-02-17 - Winter Break (Holiday)
2025-05-15 to 2025-05-22 - Mid-term Exam Period (Exam Date)
--- End of Holidays and Exam Dates ---

Enter the year (2025 or 2026) to view the academic calendar (or 0 to exit): 0
Exiting Academic Calendar.
```

4.2 User enters ‘exit’ to get back to menu.

```
Enter the year (2025 or 2026) to view the academic calendar and holidays (or 0 to exit): 0
Exiting Academic Calendar.
You:
```

5. Users enter ‘Campus Map’.

```
Enter the year (2025 or 2026) to view the academic calendar and holidays (or 0 to exit): 0
Exiting Academic Calendar.
You: Campus Map

Where? (Library/Gym/Cafeteria):
```

5.1 Users find ‘Library’.

```
You: Campus Map

Where? (Library/Gym/Cafeteria): Library

**************************
Library
- Building A, 1st Floor
- Open 7AM - 7PM

You:
```

6. Users enter ‘Contact Staff’.

```
HOW CAN I HELP?
1. Course Info
2. Academic Calendar
3. Campus Map
4. Contact Staff
5. Exit

You: contact staff

CONTACT REQUEST
Department: (Admissions/IT/Finance) :
```

6.1 Users find ‘IT Department’.

```
You: contact staff

CONTACT REQUEST
Department: (Admissions/IT/Finance) : it

****************************
IT Helpdesk:
✉️ it-support@stamford.edu
📞 02-123-4567 (9AM-5PM)

You:
```

7.0 User want to calculate budget

```
HOW CAN I HELP?
1. Course Info
2. Academic Calendar
3. Campus Map
4. Contact Staff
5. Calculate Budget
6. Exit

[main]You: 5

--- Student Budget Calculator ---
Enter your total income for the period: $
```

7.1 Prompt user income.

```
--- Student Budget Calculator ---
Enter your total income for the period: $5000
Enter your tuition fees: $
```

7.2 Prompt user for tuition fee.

```
--- Student Budget Calculator ---
Enter your total income for the period: $5000
Enter your tuition fees: $2500
Enter your accommodation costs: $
```

7.3 Prompt user accommodation.

```
--- Student Budget Calculator ---
Enter your total income for the period: $5000
Enter your tuition fees: $2500
Enter your accommodation costs: $500
Enter your estimated food costs: $
```

7.4 Prompt user for estimated food costs.

```
--- Student Budget Calculator ---
Enter your total income for the period: $5000
Enter your tuition fees: $2500
Enter your accommodation costs: $500
Enter your estimated food costs: $200
Enter your estimated transportation costs: $
```

7.5 Prompt user for transportation costs.

```
Enter your total income for the period: $5000
Enter your tuition fees: $2500
Enter your accommodation costs: $500
Enter your estimated food costs: $200
Enter your estimated transportation costs: $100
Enter your estimated personal expenses: $
```

7.6 Prompt user for transportation costs.

```
--- Student Budget Calculator ---
Enter your total income for the period: $5000
Enter your tuition fees: $2500
Enter your accommodation costs: $500
Enter your estimated food costs: $200
Enter your estimated transportation costs: $100
Enter your estimated personal expenses: $80
Enter any other expenses: $
```

7.7 Prompt user for transportation costs.

```
--- Student Budget Calculator ---
Enter your total income for the period: $5000
Enter your tuition fees: $2500
Enter your accommodation costs: $500
Enter your estimated food costs: $200
Enter your estimated transportation costs: $100
Enter your estimated personal expenses: $80
Enter any other expenses: $0
```

7.8 Print Budget Summary and Remaining Budget.

```
--- Budget Summary ---
Income: $5000.00
Tuition Fees: $2500.00
Accommodation: $500.00
Food: $200.00
Transportation: $100.00
Personal Expenses: $80.00
Other Expenses: $0.00
----------------------
Remaining Budget: $1620.00
```

8. User Exit the Program

```
You: exit

Bye! See you again
```

Limitation and Future Development

While the current Stamford Chatbot project has successfully demonstrated core Java concepts and a modular system, it has several limitations. For instance, due to lack of advanced natural language processing, users have to follow rigid input formats and keywords. Which ultimately reduces flexibility and engagement from a user. Furthermore, the functions are isolated in their modules that can limit the flow of context and memory between each class. These constraints highlight areas for future enhancement that will make the chatbot more interactive.

Conclusion

In conclusion, the Stamford Chatbot project stands as a successful demonstration of a Java-based console application. The adoption of a modular and layered design approach ensured a clear separation of concerns that foster better scalability. Additionally, by following a Spiral Model, the development process has been comprehensive in planning, analysis, building and testing at each phase. Furthermore, the project effectively showcased the practical application of core Java programming concepts such as object instantiation, conditional logic, 2D arrays, and usage of both static and non-static methods. Likewise, Object Oriented Programming concepts such as encapsulation have been applied with the use of private and get-set methods. Overall, the Stamford Chatbot not only achieved its intended goals of providing essential campus information and support but also served as a powerful practical exercise in applying object-oriented programming principles to address a real-world problem.

References

List any resources or references you used during the development.