# Getting Ready: Stack Overflow

Understand the Stack Overflow problem and learn the questions to further simplify the design problem.

**We'll cover the following**

* [Problem definition](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#Problem-definition)
* [Expectations from the interviewee](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#Expectations-from-the-interviewee)
  + [Discoverability](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#Discoverability)
  + [Reputation](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#Reputation)
  + [Voting](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#Voting)
  + [Bounty](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#Bounty)
* [Design approach](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#Design-approach)
* [Design pattern](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#Design-pattern)

## Problem definition

**Stack Overflow**is a Q&A website for programmers and developers, irrespective of their expertise level in the domain. It has a wide variety of questions on topics related to computer science and programming. Registered users can post new questions and answer questions from other users. Each user can collect reputation points. These points are affected by the upvotes or downvotes received by the user on their questions or answers. More reputation points allow users to perform additional functions like voting to close or delete a question. On achieving reputation milestones, users are awarded badges to highlight their credibility.

A question page on the Stack Overflow

## Expectations from the interviewee

It is important to narrow down the components to be included in your Stack Overflow design. The following section provides an overview of some of the main expectations that the interviewer will want to hear you discuss in more detail during the interview:

### Discoverability

You may want to ask the interviewer the following to get a better understanding of how Stack Overflow's discoverability works:

* How are users able to search for questions?
* Is there a way to filter questions using tags or users?

### Reputation

Reputation points affect user privileges. Therefore, it's important to ask the interviewer the following aspects of reputation points:

* How are reputation points calculated? Do users get points for asking or answering questions?
* How many points are required for users to get a moderator access?

### Voting

Voting is one of the main features of Stack Overflow. It gives insight on which questions or answers are more popular among users. Make sure to ask the following questions to understand how voting works in Stack Overflow:

* What are the different types of voting allowed on Stack Overflow? Are you allowed to upvote and downvote?
* How does voting work when a question has to be closed and deleted? Which user can vote in such circumstances?

### Bounty

**Bounty** is a special reputation placed on a question that is not being noticed or answered. Any user that answers a bounty question receives reputation points equal to the bounty value. You need to clarify the bounty requirements from the interviewer by asking the following questions:

* How are reputation points awarded on bounty questions?
* When do users start a bounty? How long does a bounty last before expiring?

## Design approach

We’ll design Stack Overflow using the bottom-up design approach. For this purpose, we’ll follow the steps below:

* Identify and design the smallest components first, like a question and answer.
* Use these small components to design additional components, for example, comments, bounty, and tags.
* Repeat the steps above until we design the complete Stack Overflow platform.

## Design pattern

It is always a good practice to discuss the design patterns that the Stack Overflow falls under, during the interview. Stating the design patterns will give the interviewer a positive impression and shows that the interviewee is well-versed in the advanced concepts of object-oriented design.

The following design pattern is used to design Stack Overflow:

* Observer design pattern

Let’s explore the requirements of Stack Overflow in the next lesson.

Back

# Requirements for Stack Overflow

Learn about the requirements for the Stack Overflow.

**We'll cover the following**

* [Requirement collection](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#Requirement-collection)

In this lesson, we’ll list the requirements of Stack Overflow. This is a very crucial step as requirements define the scope of a problem, so getting them right from the interviewer and understanding them well will make the design of the rest of the system smooth and easy.

We’ll use the notational convention to identify each requirement with a unique label "Rn", where "R" is short for Requirement and "n" is a natural number.

## Requirement collection

The requirements for Stack Overflow are defined below:

**R1:** Any guest can view questions and search questions by tag, username or words.

**R2:** Users should be able to post new questions and add answers to an open question.

**R3:** Users can flag a question, answer, or comment if anything goes against the community guidelines.

**R4:** A user can upvote, downvote, and add comments to a question or answer, while they can only upvote a comment.

**R5:** Users can vote to delete or vote to close off questions for community-specific reasons. However, they can only vote to delete an answer.

**R6:** Any user can add a bounty to their question to attract more answers.

**R7:** Moderators can close a question or restore an already deleted question. Moderators can also delete answers.

**R8:** The system should send the user a notification whenever there has been an interaction with them, such as the user’s question receiving an answer, earning a badge, or someone upvoting or downvoting their post.

**R9:** Users can earn badges for their helpful answers or comments.

**R10:** The system should also be able to determine the most popular tags used in questions.

**R11:** Users can add tags to their questions. A **tag** is a word or phrase that describes the topic of the question.

We've identified our requirements for the problem, and in the next lesson, we will define different use cases of Stack Overflow.

# Use Case Diagram for Stack Overflow

Learn how to define use cases and create the corresponding use case diagram for the Stack Overflow problem.

**We'll cover the following**

* [System](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#System)
* [Actors](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#Actors)
  + [Primary actors](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#Primary-actors)
  + [Secondary actors](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#Secondary-actors)
* [Use cases](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#Use-cases)
  + [User](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#User)
  + [Guest](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#Guest)
  + [Admin](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#Admin)
  + [Moderator](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#Moderator)
  + [System](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#System)
* [Relationships](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#Relationships)
  + [Generalization](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#Generalization)
  + [Associations](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#Associations)
  + [Include](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#Include)
  + [Extend](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#Extend)
* [Use case diagram](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#Use-case-diagram)

Let’s build the use case diagram of Stack Overflow and understand the relationship between its different components.

First, we’ll define the different elements of our Stack Overflow system, followed by the complete use case diagram of the system.

## System

Our system is "Stack Overflow."

## Actors

Now, we’ll define the main actors of Stack Overflow.

### Primary actors

* **User:**This actor can create, answer, and flag questions. This actor can add bounty, tags, and comments to questions, and vote to close or delete questions or answers.

### Secondary actors

* **Guest:** The guest can only search and view questions and their answers but needs to register an account to ask and answer questions.
* **Admin:** The admin is in charge of performing numerous operations, such as blocking or unblocking users.
* **Moderator:** The moderator is in charge of performing numerous operations, such as closing, reopening, deleting, and restoring questions.
* **System:** This is responsible for awarding badges to users and sending out notifications whenever a new answer is added to a question asked by a user.

## Use cases

In this section, we’ll define the use cases for Stack Overflow. We have listed the use cases according to their respective interactions with a particular actor.

**Note:**You will see some use cases occurring multiple times because they are shared among different actors in the system.

### User

* **Login/Logout:** To log in or log out from the Stack Overflow account
* **Reset password:** To reset the password of the Stack Overflow account
* **Add/modify/flag question:**To create a new question or modify or flag an existing question
* **Add/modify/flag answer:**To create a new answer or modify or flag an existing answer
* **Add comment:**To add a comment to a question or answer
* **Vote to close/delete question:** To vote to either close or delete an existing question
* **Upvote/downvote:**To either offer support or register disapproval for a post

### Guest

* **Search/view question**: To search for a question from Stack Overflow and view its contents
* **Register account:**To register an account

### Admin

* **Block/unblock user:**To block or unblock a user

### Moderator

* **Close/reopen/delete/restore question:**To close or delete an existing question, reopen a closed question or restore a deleted question
* **Delete answer:**To delete an existing answer

### System

* **Award badge to a user:**To award an earned badge to a user
* **Send notification:**To send a notification if any new answer or comment is added to a question followed by a user, and if there is an upvote or downvote on a question/answer

## Relationships

We describe the relationships between and among actors and their use cases in this section.

### Generalization

* "Moderator" has a generalization relationship with "User" as the moderator can perform all those tasks that a normal user can perform.
* "User" has a generalization relationship with "Guest" as the normal user can perform all those tasks that a guest user can perform.

### Associations

The below table shows the association relationship between actors and their use cases.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **User** | **Guest** | **Admin** | **Moderator** | **System** |
| Search/view question | Register account | Block/unblock user | Search/view question | Award badge to user |
| Login/Logout | Search/view question |  | Login/Logout | Send notification |
| Reset password |  | Reset password |  |
| Add/modify/flag question | Add/modify/flag question |
| Add/modify/flag answer | Add/modify/flag answer |
| Add comment | Add comment |
| Upvote/downvote | Upvote/downvote |
| Vote to close/delete question | Close/reopen/delete/restore question |
|  | Delete answer |

### Include

* The “Send notification” use case also has an include relationship with the “Add answer,” “Add comment,” and “Add question” use cases, since a notification is sent whenever an interaction is made for any of these.

### Extend

* The “Modify question” use case has an extend relationship with the “Modify tag” and “Modify bounty” use cases, since the modification of a question can involve the steps to modify the tags and bounties.
* The “Add question” use case has an extend relationship with the “Add bounty” and “Add tag” use cases, since a user can either add a bounty or tag when creating a question.
* The “Modify question” use case also has an extend relationship with the “Add tag” and “Add bounty” use cases, since a user can decide to add tags and bounties after creating a question as well.

## Use case diagram

A diagram of a diagram

Description automatically generated with medium confidence

# Class Diagram for Stack Overflow

Learn to create a class diagram for Stack Overflow using the bottom-up approach.

**We'll cover the following**

* [Components of Stack Overflow](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#Components-of-Stack-Overflow)
  + [Account](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#Account)
  + [Guest](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#Guest)
  + [Question](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#Question)
  + [Answer](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#Answer)
  + [Comment](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#Comment)
  + [Bounty](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#Bounty)
  + [Badge](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#Badge)
  + [Tag and tag list](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#Tag-and-tag-list)
  + [User](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#User)
    - [Admin](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#Admin)
    - [Moderator](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#Moderator)
  + [Notification](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#Notification)
  + [Search interface](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#Search-interface)
  + [Search catalog](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#Search-catalog)
  + [Enumerations](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#Enumerations)
* [Relationship between the classes](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#Relationship-between-the-classes)
  + [Association](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#Association)
  + [Composition](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#Composition)
  + [Generalization](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#Generalization)
  + [Inheritance](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#Inheritance)
* [Class diagram of Stack Overflow](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#Class-diagram-of-Stack-Overflow)
* [Design pattern](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#Design-pattern)
* [Additional requirements](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#Additional-requirements)

In this lesson, we’ll identify and design the classes, abstract classes, and interfaces based on the requirements we have previously gathered from the interviewer in our Stack Overflow system.

## Components of Stack Overflow

As mentioned earlier, we will design the Stack Overflow system using a bottom-up approach.

### Account

The Account class is used to identify a Stack Overflow user using their username, email, and id. Users with an account will have the option to create questions and add answers and comments to questions.

Here's what the class definition looks like:

A screenshot of a computer

Description automatically generated

A screenshot of a computer screen

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer screen

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a search catalog

Description automatically generated

### Enumerations

The enumerations required in the Stack Overflow design are provided below:

* AccountStatus**:**The account status tells us about the status of a user's account whether it is active, disabled, or blocked.
* QuestionStatus**:** The question status describes the status of an existing question, whether it is still active, closed, flagged, or bountied.
* ClosingDetails**:**The closing details enumeration outlines the reason that a question was closed.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**Note:** We have already discussed the inheritance relationship between classes in the component section above one by one.

## Class diagram of Stack Overflow

Here’s the complete class diagram for Stack Overflow:

A screenshot of a computer

Description automatically generated

A screenshot of a computer screen

Description automatically generated

A screenshot of a computer screen

Description automatically generated

# Sequence Diagram for Stack Overflow

Visualize the sequence diagram for closing a question and practice the concepts with a challenge.

**We'll cover the following**

* [Close question](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#Close-question)
* [Sequence challenge: Create question](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#Sequence-challenge:-Create-question)

Sequence diagrams are a great way to understand the interactions between different entities and objects in the system. There can be different sequence diagrams that we can create for Stack Overflow. In this lesson, we will create sequence diagrams for the following two interactions:

* **Close question:** A user votes to close a question authored by a different user.
* **Sequence challenge:** A user creates a question and adds a bounty.

## Close question

The sequence diagram for closing a question should have the following actors and objects that will interact with each other:

* **Actors:** User and Author
* **Object:**Question

Here are the steps in the close question interaction:

1. A user votes to close a question with some remark.
2. If the user is a moderator:
   1. The question is closed.
   2. The author of the question is notified about the question.
3. Else if they are a regular user:
   1. If the user's reputation is greater than or equal to 3000:
      1. A close vote is added.
      2. If there are 3 close votes, the question closes, and the author is notified.
   2. Else if the user's reputation is less than 3000:
      1. The vote is considered invalid.

Based on the order above, the sequence diagram for closing a question in Stack Overflow is provided below:

A screenshot of a computer screen

Description automatically generated

## Sequence challenge: Create question

You’ll help us complete a sequence diagram for a new question created by a user where the user adds a bounty to the question. A skeleton of the create question sequence diagram is given below:

A diagram of a diagram

Description automatically generated

**Note:** If you get stuck, just click the “Show Solution” button to check out the correct answer.

A screenshot of a diagram

Description automatically generated

Next, let's look at the activity diagrams for Stack Overflow to understand the control flow of the system.

# Activity Diagram for Stack Overflow

Create some activity diagrams for the Stack Overflow problem.

**We'll cover the following**

* [A member posting a question](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#A-member-posting-a-question)
  + [States](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#States)
  + [Actions](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#Actions)
* [Activity challenge: A member closes a question](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#Activity-challenge:-A-member-closes-a-question)

Activities diagrams are a great way to visualize the flow of messages from one activity to the other in the system. There can be different activity diagrams that we can create for our Stack Overflow system. In this lesson, we will create activity diagrams for the following two activities:

* A member posting a question
* **Activity challenge:** A member closes a question.

## A member posting a question

The following are the states and actions that will be involved in this activity diagram.

### States

**Initial state:**The member clicks on the “Ask Question” button.

**Final state:**There are two final states present in this activity diagram, shown below:

* The question was successfully posted.
* The question is not posted.

### Actions

The member clicks on the “Ask Question” button and fills in the required fields. The member is then asked if they want to add a tag. After the post is composed, the system checks if the terms of service are followed. The question is posted only if the terms of service are followed.

A diagram of a question

Description automatically generated

## Activity challenge: A member closes a question

You’ll help us create an activity diagram of a member closing a question.

A skeleton of the activity diagram is provided below:

A diagram of a group of people

Description automatically generated

Notice that the actions in the diagram above are numbered from 1 to 10. The slots below represent the activities, and the arrows represent the flow from one activity to the other.

Can you rearrange the slots below in the correct order they should appear in the activity diagram given above?

**Note:** If you get stuck, just click the “Show Solution” button to check out the correct answer.

Alternatively, you can click the "Show complete diagram" button below to see the complete activity diagram.

A screenshot of a diagram

Description automatically generated

# Code for Stack Overflow

Write the object-oriented code to implement the design of the Stack Overflow problem.

**We'll cover the following**

* [Stack Overflow classes](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#Stack-Overflow-classes)
  + [Constants](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#Constants)
  + [Account](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#Account)
  + [User, admin, moderator, and guest](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#User,-admin,-moderator,-and-guest)
  + [Question, answer, comment, and bounty](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#Question,-answer,-comment,-and-bounty)
  + [Badge, tag, and tag list](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#Badge,-tag,-and-tag-list)
  + [Notification](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#Notification)
  + [Search catalog and interface](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#Search-catalog-and-interface)
* [Wrapping up](https://www.educative.io/courses/grokking-the-low-level-design-interview-using-ood-principles/getting-ready-the-amazon-online-shopping-system#Wrapping-up)

We've gone over the different aspects of Stack Overflow and observed the attributes attached to the problem using various UML diagrams. Let us now explore the more practical side of things, where we will work on implementing the Stack Overflow network using multiple languages. This is usually the last step in an object-oriented design interview process.

We have chosen the following languages to write the skeleton code of the different classes present in Stack Overflow:

* Java
* C#
* Python
* C++
* JavaScript

## Stack Overflow classes

In this section, we will provide the skeleton code of the classes designed in the class diagram lesson.

**Note:**For simplicity, we are not defining getter and setter functions. The reader can assume that all class attributes are private and accessed through their respective public getter methods and modified only through their public method functions.

### Constants

The following code provides the definition of the various enums and custom data types being used in the Stack Overflow design:

**Note:**JavaScript does not support enumerations so we will be using the Object.freeze() method as an alternative that freezes an object and prevents further modifications.

enum AccountStatus {

ACTIVE,

BLOCKED,

DISABLED

}

enum QuestionStatus {

ACTIVE,

CLOSED,

FLAGGED,

BOUNTIED

}

enum ClosingDetail {

COMMUNITY\_SPECIFIC\_REASON,

DUPLICATE,

NEEDS\_CLARITY,

NEEDS\_MORE\_FOCUS,

OPINION\_BASED

}

A screenshot of a computer

Description automatically generated

public class User {

private int reputationPoints;

private Account account;

private List<Badge> badges;

public boolean createQuestion(Question question);

public boolean addAnswer(Question, question, Answer answer);

public boolean createComment(Comment comment);

public boolean createTag(Tag tag);

public void flagQuestion(Question question);

public void flagAnswer(Answer answer);

public void upvote(int id);

public void downvote(int id);

public void voteToCloseQuestion(Question question);

public void voteToDeleteQuestion(Question question);

public void acceptAnswer(Answer answer);

}

public class Admin extends User {

public boolean blockUser(User user);

public boolean unblockUser(User user);

public void awardBadge(User user, Badge badge);

}

public class Moderator extends User {

public void closeQuestion(Question question);

public void reopenQuestion(Question question);

public void deleteQuestion(Question question);

public void restoreQuestion(Question question);

public void deleteAnswer(Answer answer);

}

public class Guest {

public void registerAccount();

}

### Question, answer, comment, and bounty

Stack Overflow users can create and answer questions, upvote and downvote them, and add bounties and comments to questions. The definition of these classes is provided below:

Java

public class Question {

private int id;

private String title;

private String content;

private User createdBy;

private int upvotes;

private int downvotes;

private int viewCount;

private int score;

private int voteCount;

private Date creationDate;

private Date modificationDate;

private QuestionStatus status;

private ClosingDetails closingReason;

private Bounty bounty;

private List<Tag> tags;

private List<Comment> comments;

private List<Answer> answers;

private List<User> followers;

public void addComment(Comment comment);

public void addBounty(Bounty bounty);

}

public class Comment {

private int id;

private String content;

private int flagCount;

private int upvotes;

private Date creationDate;

private User postedBy;

}

public class Answer {

private int id;

private String content;

private int flagCount;

private int voteCount;

private int upvotes;

private int downvotes;

private boolean isAccepted;

private Date creationTime;

private User postedBy;

private List<Comment> comments;

private List<User> followers;

public void addComment(Comment comment);

}

public class Bounty {

private int reputationPoints;

private Date expiryDate;

public boolean updateReputationPoints(int reputation);

}

A screenshot of a computer program

Description automatically generated

### Notification

The Notification class is responsible for sending notifications to users about any new messages, comments, posts, or friend requests via either a phone number, or an email. Its definition is provided below:

Java

C#

Python

C++

JavaScript

public class Notification {

private int notificationId;

private Date createdOn;

private String content;

public boolean sendNotification(Account account);

}

### Search catalog and interface

The SearchCatalog class contains information on existing questions and answers. It also implements the Search interface class to enable the search functionality based on the given criteria (tags, usernames, and searched keywords). The definition of these two classes is provided below:

public interface Search {

public List<Question> searchByTags(String name);

public List<Question> searchByUsers(String name);

public List<Question> searchByWords(String words);

}

public class SearchCatalog implements Search {

private HashMap<String, List<Tag>> questionsUsingTags;

private HashMap<String, List<User>> questionsUsingUsers;

private HashMap<String, List<String>> questionsUsingWords;

public List<Question> searchByTags(String name) {

// functionality

}

public List<Question> searchByUsers(String name) {

// functionality

}

public List<Question> searchByWords(String words) {

// functionality

}

}

**Wrapping up**

We've explored the complete design of Stack Overflow in this chapter. We've looked at how Stack Overflow can be visualized using various UML diagrams and designed using object-oriented principles and design patterns.