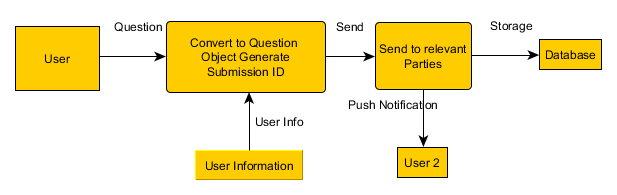
Users can generate two kinds of submissions: answer submissions, and question submissions.

**QuestionSubmission**

A question takes text and categories from the user and creates a **QuestionSubmission** object. The user enters the title for their and their questions body then presses submit. The system requests categories for the question such as course information through an alert. After the user confirms there categories a variety of information is passed to the backend of the system.

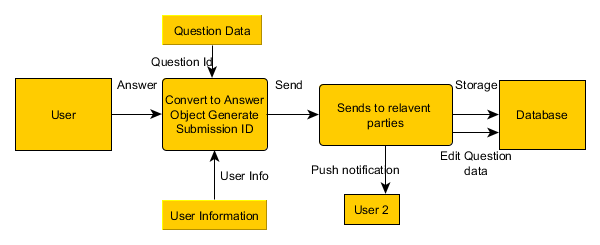
The system creates a **QuestionSubmission** object by setting **Title**, a string equal to the title textbox contents, setting **CategoryID,** an integer, to the selected category, initializing **QuestionAnswers**, an array, for **AnswerIDs** as an empty array, and generating the remainder of a **Submission** object which it inherits from. This involves setting a **Body**, a string, equal to the question textbox data, setting an integer **AuthorID** equal to **UserID**, initializing **Score**, an integer, to zero and its **Status** to Active, setting **Date**, aJSdate object**,** equal to the current date and generating an **ID**, an integer, for itself. The information is then placed in the database and sent to other users based on the **CategoryID**. The following dataflow diagram illustrates this process:



**AnswerSubmission**

An answer takes text from the user and creates an answer object. The user enters their response to a given question and presses submit. The **ParentQuestionID**, **UserID** and answer text are then sent to the back end of the system. The backend takes this information and generates an **AnswerSubmission** object.

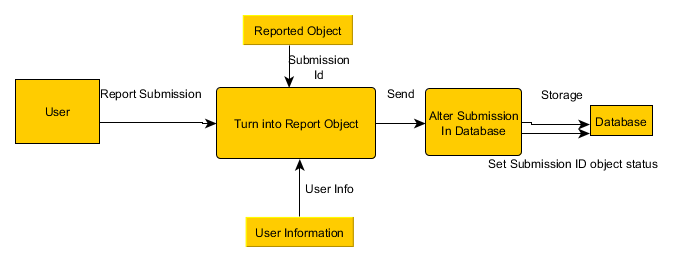
The answer object is created by setting **ParentQuestionID**, an integer, equal to the **AuthorID** of the question the user is answering and generating the remainder of a **Submission** object which **AnswerSubmission** objects inherit from. This involves setting **Body,** a string, equal to the answer text, setting **AuthorID**, an integer, equal to **UserID**, initializing **Score**, an integer, to zero and initializing **Status** to Active, setting **Date**, a JS Date object, equal to the current date, and generating an **ID**, an integer, for itself. The backend then updates the **QuestionSubmission** object in the database by placing its own **ID** in the **QuestionAnswers** array found using the **ParentQuestionID**. Finally the backend pushes a notification to the user who asked the question by referencing the **AuthorId** on the **QuestionSubmission** object. The following dataflow diagram illustrates this process:



**AbuseReport**

A report takes a report type and a brief description of why they are reporting a submission to make an **AbuseReport**. The user selects either spam or language for the reason of the report. The backend then creates an **AbuseReport** object.

The backend generates **AbuseReportID**, **a**n integer, for the object, setting **Reason**, a string, to either spam or language, setting **Description**, a string, to the description entered by the user, setting **SubmissionID,** an integer, to the reported objects submission id, setting **Date**, a JS Date object, equal to the current date, setting its **Status** to active and setting **ReporterID**, an integer, to the reporters **UserID**. Once the report has been generated the **Status** of the flagged item is set to under review. The **AbuseReport** object is then place in storage to be reviewed by an admin later. The following dataflow diagram illustrates this process:



The following Data Dictionary lists all the data elements used in the above descriptions:

|  |  |  |
| --- | --- | --- |
| Name of Data Element | Description | Narative |
| AbuseReport | Record comprising fields:  AbuseReportID  Reason  Description  SubmissionID  Status  ReporterID  Date | The fields contain all details of a report |
| AbuseReportId | An integer | A unique number generated by generate\_report\_ID |
| AnswerID | An integer | A unique number generated by submissionID |
| AnswerSubmission | Record Comprising fields:  AuthorID  Body  Date  ParentQuestionID  ID  Score  Status | The fields contain all details of an AnswerSubmission |
| AuthorID | An integer | The UserID from the user who generates a submission |
| Body | A string | contains the bulk of a question gathered from the description textbox in the UI when a user submits a question |
| CategoryID | An integer | The CategoryID generated when the user selects one of several pre-set categories |
| Date | A JS date Object | The current date generated using built in JS functions |
| Description | A string | contains a description of a report |
| ID | An integer | An ID for a submission generated by a function |
| ParentQuestionID | An integer | An ID in an answer object retrieved from the question it is answering |
| QuestionAnswers | An array | An array initialized to empty to contain the answers to the question |
| QuestionSubmission | A field containing:  Category ID  Body  Title  Date  QuestionAnswers  ID  Status  AuthorID  Date  Score | The fields contain all details of a QuestionSubmission |
| ReporterId | An integer | The User ID of the user submitting a AbuseReport object |
| Score | An integer | An integer attached to a submission. The score is altered by users rating questions and answers and is used to decide what to display first |
| Status | A status enum | A Status Enum with 3 state  Active  UnderReview  Deleted |
| Submission | A field containing:  Body  Date  ID  Status  AuthorID  Date  Score | The fields contain all details of a Submission |
| SubmissionID | An integer | The ID of a reported submission gathered by a report |
| Title | A string | A brief description of a question gathered from the title textbox in the UI |
| UserID | An integer | The ID of a user generated upon profile creation |