# My Project

Generated by Doxygen 1.8.11

# Contents

# Chapter 1

# Namespace Index

## 1.1 Namespace List

Here is a list of all documented namespaces with brief descriptions:

# Chapter 2

# Hierarchical Index

## 2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

# Chapter 3

# Class Index

## 3.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

# Chapter 4

# Namespace Documentation

## 4.1 models Namespace Reference

Different tables to be stored in the db.

### 4.1.1 Detailed Description

Different tables to be stored in the db.

## 4.2 views Namespace Reference

Actual operations on the db are performed in this package.

### 4.2.1 Detailed Description

Actual operations on the db are performed in this package.

# Chapter 5

# Class Documentation

## 5.1 api.models.Answer Class Reference

Model for Answer.

Inheritance diagram for api.models.Answer:

Collaboration diagram for api.models.Answer:

### Public Member Functions

- def **__str__** (self)

### Static Public Attributes

- **user** = models.ForeignKey(User)
- **username** = models.CharField(max_length=100, blank=True)
- **userdepartment** = models.CharField(max_length=4, choices=department_choices, default='CSE')
- **userbio** = models.CharField(max_length=1000, blank=True)
- **userdegree** = models.CharField(max_length=9, choices=degree_choices, default='NA')
- **userspecialization** = models.CharField(max_length=100, blank=True)
- **content** = models.CharField(max_length=10000)
- **question** = models.ForeignKey(Question)
- **votes** = models.IntegerField(default=0)
- **timestamp** = models.DateTimeField(auto_now=True)

**Parameters**

### 5.1.1 Detailed Description

Model for Answer.

**Parameters**

| | |
|---|---|
| *user* | each answer will be answered by a user but a user can give many answers hence the foreignkey |
| *username* | the name of the user who gave the answer |
| *userdepartment* | the department of the user who gave the answer |
| *userbio* | bio of the user who gave the answer |
| *userdegree* | the degree of the user who gave the answer |
| *userspecialization* | the specialization of the user who gave the answer |
| *content* | the actual answer |
| *question* | each answer must be in response to a question but a question can have many answers hence the one-to-many relationship between these models |
| *votes* | an integer specifying the cumulative votes of an answer by combining upvotes with downvotes |
| *timestamp* | last modified timestamp for the answer |

The documentation for this class was generated from the following file:

- models.py

## 5.2 api.serializers.AnswerSerializer Class Reference

Inheritance diagram for api.serializers.AnswerSerializer:

Collaboration diagram for api.serializers.AnswerSerializer:

### Classes

- class Meta

The documentation for this class was generated from the following file:

- serializers.py

## 5.3 api.apps.ApiConfig Class Reference

Inheritance diagram for api.apps.ApiConfig:

## 5.4 api.serializers.TopicSerializer.Meta Class Reference

### Static Public Attributes

- **model** = Topic
- tuple **fields** = ('id', 'name')

The documentation for this class was generated from the following file:

- serializers.py

## 5.5 api.serializers.VoteSerializer.Meta Class Reference

### Static Public Attributes

- **model** = Vote
- tuple **fields** = ('id', 'upvote_downvote', 'user', 'answer')

The documentation for this class was generated from the following file:

- serializers.py

## 5.6 api.serializers.NotificationSerializer.Meta Class Reference

### Static Public Attributes

- **model** = Notification
- tuple **fields** = ('id', 'receiver', 'sender', 'sendername', 'question', 'code', 'timestamp')

The documentation for this class was generated from the following file:

- serializers.py

## 5.7 api.serializers.UserSerializer.Meta Class Reference

### Static Public Attributes

- **model** = User
- tuple **fields** = ('id', 'name', 'ldapid', 'gender', 'department', 'bio', 'student_professor', 'degree', 'year', 'specialization', 'password', 'subscribed_topics', 'totalvotes')

The documentation for this class was generated from the following file:

- serializers.py

## 5.8 api.serializers.QuestionSerializer.Meta Class Reference

### Static Public Attributes

- **model** = Question
- tuple **fields** = ('id', 'user', 'username', 'userdepartment', 'userbio', 'userdegree', 'userspecialization', 'topics', 'question', 'description', 'numAnswers', 'timestamp')

The documentation for this class was generated from the following file:

- serializers.py

## 5.9 api.serializers.AnswerSerializer.Meta Class Reference

### Static Public Attributes

- **model** = Answer
- tuple **fields** = ('id', 'user', 'username', 'userdepartment', 'userbio', 'userdegree', 'userspecialization', 'content', 'question', 'votes', 'timestamp')

The documentation for this class was generated from the following file:

- serializers.py

## 5.10 api.models.Notification Class Reference

Model for Notification.

Inheritance diagram for api.models.Notification:

Collaboration diagram for api.models.Notification:

### Public Member Functions

- def **__str__** (self)

### Static Public Attributes

- **receiver** = models.ForeignKey(User, related_name='receiver')
- **sender** = models.ForeignKey(User, related_name='sender')
- **sendername** = models.CharField(max_length=100, blank=True)
- **question** = models.ForeignKey(Question)
- **code** = models.IntegerField(default=1)
- **timestamp** = models.DateTimeField(auto_now=True)

### 5.10.1 Detailed Description

Model for Notification.

**Parameters**

| | |
|---|---|
| *receiver* | the user for which the notification needs to be shown; either an answer has been submitted to his question or his answer has been upvoted/downvoted |
| *sender* | the user who has answered or voted |
| *sendername* | name of the user who answered or voted |
| *question* | the question for which the change has happened |
| *code* | this is an integer field with 1 denoting upvote, 2 denoting downvote and 3 denoting that an answer was posted |
| *timestamp* | last modified timestamp for the notification |

The documentation for this class was generated from the following file:

- models.py

## 5.11 api.serializers.NotificationSerializer Class Reference

Inheritance diagram for api.serializers.NotificationSerializer:

Collaboration diagram for api.serializers.NotificationSerializer:

### Classes

- class Meta

The documentation for this class was generated from the following file:

- serializers.py

## 5.12 api.models.Question Class Reference

Model for Question.

Inheritance diagram for api.models.Question:

Collaboration diagram for api.models.Question:

### Public Member Functions

- def **__str__** (self)

### Static Public Attributes

- **user** = models.ForeignKey(User)
- **username** = models.CharField(max_length=100, blank=True)
- **userdepartment** = models.CharField(max_length=4, choices=department_choices, default='CSE')
- **userbio** = models.CharField(max_length=1000, blank=True)
- **userdegree** = models.CharField(max_length=9, choices=degree_choices, default='NA')
- **userspecialization** = models.CharField(max_length=100, blank=True)
- **topics** = models.ManyToManyField(Topic)
- **question** = models.CharField(max_length=500, unique=True)
- **description** = models.CharField(max_length=5000)
- **numAnswers** = models.IntegerField(default=0)
- **timestamp** = models.DateTimeField(auto_now=True)

### 5.12.1 Detailed Description

Model for Question.

**Parameters**

| | |
|---|---|
| *user* | each question can be asked by a unique user but a user can ask multiple questions thus implying a one-to-many relationship between User and Question which is represented by using User as foreignkey in Question |
| *username* | the name of the user who asked the question |
| *userdepartment* | the department of the user who asked the question |
| *userbio* | bio of the user who asked the question |
| *userdegree* | the degree of the user who asked the question |
| *userspecialization* | the specialization of the user who asked the question |
| *topics* | each question can be tagged by multiple topics and each topic may have multiple questions associated with it |
| *question* | the main problem statement |
| *description* | the detailed description of the problem statement |
| *timestamp* | the timestamp when the question was last modified. By setting auto_now paramter to be True we automatically ensure that the timestamp will be updated whenever the question entry is saved |

The documentation for this class was generated from the following file:

- models.py

## 5.13 api.serializers.QuestionSerializer Class Reference

Inheritance diagram for api.serializers.QuestionSerializer:

Collaboration diagram for api.serializers.QuestionSerializer:

### Classes

- class Meta

The documentation for this class was generated from the following file:

- serializers.py

## 5.14 api.models.Topic Class Reference

Model for Topic.

Inheritance diagram for api.models.Topic:

Collaboration diagram for api.models.Topic:

### Public Member Functions

- def **__str__** (self)

### Static Public Attributes

- **name** = models.CharField(max_length=100, unique=True)

### 5.14.1 Detailed Description

Model for Topic.

**Parameters**

| | |
|---|---|
| *name* | the name of the topic |

The documentation for this class was generated from the following file:

- models.py

## 5.15 api.serializers.TopicSerializer Class Reference

Inheritance diagram for api.serializers.TopicSerializer:

Collaboration diagram for api.serializers.TopicSerializer:

### Classes

- class Meta

The documentation for this class was generated from the following file:

- serializers.py

## 5.16 api.models.User Class Reference

Model for User.

Inheritance diagram for api.models.User:

Collaboration diagram for api.models.User:

### Public Member Functions

- def **__str__** (self)

### Static Public Attributes

- **name** = models.CharField(max_length=100)
- **ldapid** = models.CharField(max_length=100, unique=True)
- **gender** = models.CharField(max_length=1, choices=gender_choices, default='M')
- **department** = models.CharField(max_length=4, choices=department_choices, default='CSE')
- **bio** = models.CharField(max_length=1000)
- **student_professor** = models.BooleanField(default=True)
- **degree** = models.CharField(max_length=9, choices=degree_choices, default='NA')
- **year** = models.CharField(max_length=2, choices=year_choices, default='NA')
- **specialization** = models.CharField(max_length=100, blank=True)
- **password** = models.CharField(max_length=20)
- **subscribed_topics** = models.ManyToManyField(Topic)
- **totalvotes** = models.IntegerField(default=0)

### 5.16.1 Detailed Description

Model for User.

**Parameters**

| | |
|---|---|
| *name* | the name of the user |
| *ldapid* | the ldap id of the user associated with the IITB account |
| *gender* | this is gender of the user with 2 options - M and F |
| *department* | this is the department of the user with possible options - ('AE', 'BSBE', 'CHE', 'CHM', 'CE', 'CSE', 'ES', 'EE', 'ESE', 'CESE', 'HSS', 'IDC', 'MTH', 'ME', 'MEMS', 'PHY') |
| *bio* | this is a brief description of the user (optional) |
| *student_professor* | this is a boolean variable representing whether the user is a student or a professorr |

**Parameters**

| | |
|---|---|
| *degree* | this is the degree of the student user among the following options - ('BTECH', 'DUAL', 'BS', 'BDES', 'MTECH', 'MTECH-PHD', 'MDES', 'MPHIL', 'MMGT', 'EMBA', 'MSC', 'MSC-PHD', 'PHD') |
| *year* | this is the year of the student user among - ('F', 'S', 'T', 'FO', 'FI') |
| *specialization* | this applies to students pursuing higher studies as well as professors |
| *password* | this is the password associated with the user in our app |
| *subscribed_topics* | This represents the list of topics subscribed with the user. Since each user can be subscribed with multiple topics and each topic can be associated to multiple users, there is a many-to-many relationship between model User and Topic which is identified by the ManyToManyField option |
| *totalvotes* | this is an integer describing the total cumulative votes of a user (can be both positive or negative) |

The documentation for this class was generated from the following file:

- models.py

## 5.17 api.serializers.UserSerializer Class Reference

Inheritance diagram for api.serializers.UserSerializer:

Collaboration diagram for api.serializers.UserSerializer:

### Classes

- class Meta

The documentation for this class was generated from the following file:

- serializers.py

## 5.18 api.models.Vote Class Reference

Model for Vote.

Inheritance diagram for api.models.Vote:

Collaboration diagram for api.models.Vote:

### Public Member Functions

- def **__str__** (self)

### Static Public Attributes

- **upvote_downvote** = models.BooleanField(default=True)
- **user** = models.ForeignKey(User)
- **answer** = models.ForeignKey(Answer)

### 5.18.1 Detailed Description

Model for Vote.

**Parameters**

| | |
|---|---|
| *upvote_downvote* | boolean variable such that True denotes an upvote while False denotes a downvote |
| *user* | the user who gave the vote |
| *answer* | the answer for which the vote was given |

The documentation for this class was generated from the following file:

- models.py

## 5.19 api.serializers.VoteSerializer Class Reference

Inheritance diagram for api.serializers.VoteSerializer:

Collaboration diagram for api.serializers.VoteSerializer:

### Classes

- class Meta

The documentation for this class was generated from the following file:

- serializers.py