Team Project Proposal

Teammates: Zheng Wang, Shen Guan, Yuheng Wang

Original proposal:

KidKnowGarden: An Online Education Website

This is an online K-12 & pre-K education website. The website aims to provide students and their parents with a comprehensive platform to study, test and communicate.

The website consists of several parts: test and learn, fun with learning and forums. Students can learn new knowledge by taking online tests or exams. Students can play interesting games to strengthen their previously learned knowledge with fun. Parents can discuss with each other and get student's feedback from the system or teachers.

Our modification:

We delete the parent discussion part, because it is like the Grumblr. And we mainly focus on the interactive online game part, and focus our target to kindergarten students only, they are usually from 3 years old to 5 years old.

Function:

User Authentication and User Registration:

For the registered users, they are allowed to login to the application using their username and password. They could also retrieve their password by validating the link from the E-mail generated by the application. Also, new users can register a new account by verifying their E-mail.

Reading:

The reading module of this web application could provide multiple reading material for kids with several grades. Users could select the reading materials that fits their interests, learning abilities, and age most. Tags are provided for this kind of selection. Also, only part of the reading resources are free to all users, they need to pay the premium for further chapters of reading. What's more, a link is provided if the users want to buy the printing book.

Video:

Video is a much more interactive and attractive way of learning for kids. So just as reading, we provide many series of video for kids with several grades. Users could select the materials that fits their interests, learning abilities, and age most.

Game:

All games provided has three difficulty levels, users can only unlock higher level by achieving specified score. For each game, we will store the latest 3 scores of each user. When user play the game, they can also see their previous scores. We have scoreboard for all users in our learning platform. Users can check their rank and other users' score on this scoreboard.

We plan to have three games in total, which is related to basic calculation, English and general knowledge in life. When the user finishes the game, she/he can choose to play it again and move to another page. Each time after they finish the game, we will show them the score and at the same time, update the personal scoreboard for this game and corresponding monthly scoreboard. Since our main target is the kindergarten children, the style of our game will be more child-oriented. I will introduce the functionality of three games in more details below.

Happy + and -:

This is a game to practice mathematics, users will do simple add and subtraction game, we will show the question and provide 4 options to users, only one option is correct. They can select the option and automatically enter to next question, there is no time limit on this game.

Word snake:

This is a game to practice English, users will be given more five English characters each round and they need to select the characters from the provided English alphabet list to compose the word, the first alphabet need to put at the specific place, which is marked as a star in the board. The word can only be put in horizontal line and vertical line. Different block on the block has different score, and different alphabet will be given random score, the goal of players is to achieve the highest score within given time. When the timer counts down to 0, the game ends. We will ensure user can compose at least one word from the five English alphabets. And the word the user composed, we will check it by using the Oxford dictionary API.

Image matching:

This is a game is let children know and understand some general stuff in the life, and it compose three classifications: animals, plants and living goods. Users can choose one classification, and we will provide corresponding images. The text description will be on the right to the pictures, what the users need to do is to connect the image and corresponding text description. Each section has three images, user can choose to go to next section when she/he thinks he got the correct answer. But once she/he move to next section, the previous answers cannot be changed.

Payment:

Users need to pay the premium to get access to the advance functions and materials. They could choose their own preferable way of payment, subscription tier and input the payment info. Beyond, all these transactions are done in the secure environment and secure way.

Summary:

Our web application could provide the general performance evaluation for the users and send text message to the users' supervisor. Also, the supervisor could view the recent performance and learning outcome via the scoreboard. We offer simple data visualizations of score to facilitate supervision.

Model:

Store user Info for each user

```
CREATE TABLE user(username VARCHAR(20), password VARCHAR(20), email VARCHAR(50),
firstname VARCHAR(20), lastname VARCHAR(20), userAge INT, phone_number
VARCHAR(20)),password VARCHAR(20))
```

```
class Users(models.Model):
    user_name = models.CharField(max_length=20)
    email = models.EmailField(max_length=50)
    first_name = models.CharField(max_length=20)
    last_name = models.CharField(max_length=20)
    age = models.IntegerField(null=True, blank=True)
    picture = models.ImageField(upload_to="images", blank=True, null=True)
    phone_number = models.CharField(max_length=20)
    password = models.CharField(max_length=20, blank=True)
```

Store payment Info for each user

```
CREATE TABLE payment(owner VARCHAR(20), username VARCHAR(20), credits_card_number
VARCHAR(20), cardholder_name VARCHAR(20), expire_date DATE, card_security_number
VARCHAR(20), subscription_type VARCHAR(20))
```

```
class UsersPayment(models.Model):
    owner = models.ForeignKey(User, null=True)
    user_name = models.CharField(max_length=20)
    credits_card_number = models.CharField(max_length=20)
    cardholder_name = models.CharField(max_length=20)
    expire_date = models.parse_date();
    card_security_number = models.CharField(max_length=20, blank=True)
    subscription_type = models.CharField(max_length=20)
```

Store user score for each game

```
CREATE TABLE WordSnakeScore (studentID VARCHAR(200), recentScore INT, time DATE)
```

```
class WordSnakeScore(models.Model):
    studentID = models.CharField(max_length=200)
    recentScore = models.IntegerField(default = 0)
    time = models.DateTimeField(auto_now=True)
```

```
CREATE TABLE ImageMatchingScore (studentID VARCHAR(200), recentScore INT, time DATE)
```

```
class ImageMatchingScore(models.Model):
    studentID = models.CharField(max_length=200)
    recentScore = models.IntegerField(default = 0)
    time = models.DateTimeField(auto_now=True)
```

```
CREATE TABLE AddSubstractionScore (studentID VARCHAR(200), recentScore INT, time DATE)
```

```
class AddSubstractionScore(models.Model):
    studentID = models.CharField(max_length=200)
    recentScore = models.IntegerField(default = 0)
    time = models.DateTimeField(auto_now=True)
```

Store user score for all games in 12 months

```
CREATE TABLE MonthlyScore (studentID VARCHAR(200), Jan INT, Feb INT, Mar INT, Apr INT,
May INT, Jun INT, Jul INT, Aug INT, Sep INT, Oct INT, Nov INT, Dec INT,)
```

```
class MonthlyScore(models.Model):
    studentID = models.CharField(max_length=200)
    Jan = models.IntegerField(default = 0)
    Feb = models.IntegerField(default = 0)
    Mar = models.IntegerField(default = 0)
    Apr = models.IntegerField(default = 0)
    May = models.IntegerField(default = 0)
    Jun = models.IntegerField(default = 0)
    Jul = models.IntegerField(default = 0)
    Aug = models.IntegerField(default = 0)
    Sep = models.IntegerField(default = 0)
    Oct = models.IntegerField(default = 0)
    Nov = models.IntegerField(default = 0)
    Dec = models.IntegerField(default = 0)
```

Store books info for all reading materials

```
CREATE TABLE book(book_id  VARCHAR(20), book_name VARCHAR(20), book_description
VARCHAR(400), book_grade VARCHAR(20), book_price INT, book_isPremium Boolean)
```

```
class Books(models.Model):
    book_id = models.CharField(max_length=20)
    book_name = models.CharField(max_length=20)
    book_description = models.EmailField(max_length=400)
    book_grade = models.CharField(max_length=20)
    book_price = models.IntegerField(null=True, blank=True)
    book_cover = models.ImageField(upload_to="book_images", blank=True, null=True)
    book_isPremium = models.BooleanField()
```

Store videos info for all reading materials

```
CREATE TABLE video(video_id  VARCHAR(20), video_name VARCHAR(20), video_description
VARCHAR(400), video_grade VARCHAR(20), video_price INT, video_isPremium Boolean)
```

```
class Books(models.Model):
    video_id = models.CharField(max_length=20)
    video_name = models.CharField(max_length=20)
    video_description = models.EmailField(max_length=400)
    video_grade = models.CharField(max_length=20)
    video_price = models.IntegerField(null=True, blank=True))
    video_isPremium = models.BooleanField()
```

Technology:

Google wallet: Payment to play games

Twilio: Parents can receive their children score by text.

Oxford Dictionaries API:

Applied in the word snake game