Application Design Document



**Academic Advisor:** Professor Arnon Sturm

**Technical Advisor:** Talya Marmour Ben Rey

**Team Members:** Shai Dayan, Roy Pillar, Magal Weiss

Table of Contents

1) Use Cases

2) System Architecture

2.1) Architecture diagram

2.2) Components description

3) Data Model

3.1) Data diagrams

3.2) Description of data objects

3.3) Data object relationships

3.4) Entity relationships diagram

4) User Interface Draft

5) Testing

5.1) Functional Testing

5.2) Non-Functional Testing

6) Task List

**1. Use Cases**

Use case ID: #1

Use case name: **Viewing a reading text**

Associated Requirement IDs: #1

Actors: Pupil, Teacher

Description:

Available reading texts could be viewed by the users.

Pre-conditions:

User is Logged in to the system.

Main Success scenario:

1. The User chooses a text to be displayed.
2. The system loads the text and it is displayed to the user.

Alternative scenarios:

2.1) Some error regarding the loading of the text occurs, an appropriate message is displayed to the User.

Use case ID: #2

Use case name: **Hearing a text being read out loud**

Associated Requirement IDs: #2

Actors: Pupil, Teacher

Description:

Any reading text could be Listened to by the reader of the text.

Pre-conditions:

User is viewing a reading text.

Post-conditions:

Appropriate audio file is being played.

Main Success scenario:

1. The user is reading a text, clicks a button to play the text out loud.
2. The system loads the audio and plays it to the user.

Alternative scenarios:

2.1) Some error regarding the loading of the audio occurs, an appropriate message is displayed to the User.

Use case ID: #3

Use case name: **Hearing a selected subtext being read out loud**

Associated Requirement IDs: #3

Actors: Pupil, Teacher

Description:

Any selected sub-text in a reading text could be Listened to by the reader of the text.

Pre-conditions:

User is viewing a reading text.

Post-conditions:

Appropriate audio is being played.

Main Success scenario:

1. The user is reading a text, selects some subtext of the whole reading text.
2. The user clicks a button to play the selected text out loud.
3. The system loads the audio and plays it to the user.

Alternative scenarios:

3.1) Some error regarding the loading of the audio occurs, an appropriate message is displayed to the User.

Use case ID: #4

Use case name: **Seeing the definition of a selected word or phrase**

Associated Requirement IDs: #4

Actors: Pupil, Teacher

Description:

Upon selecting a difficult word or phrase in a text, the definition of that word or phrase could be displayed to the user.

Pre-conditions:

User is viewing a reading text.

Post-conditions:

Appropriate definition is displayed.

Main Success scenario:

1. The user is reading a text, selects some word or phrase.
2. The user clicks a button to show the definition of that word or phrase.
3. The system loads the definition and displays it to the user.

Alternative scenarios:

3.1) A definition for the selected word/phrase was not found.

3.2) The selected text is not a word/phrase.

3.X.1) An appropriate error message is displayed to the User.

Use case ID: #5

Use case name: **Answering questions regarding a reading text**

Associated Requirement IDs: #5

Actors: Pupil

Description:

After a Pupil finishes reading a text, he proceeds to answer questions regarding that text.

Pre-conditions:

Pupil is viewing a reading text.

Post-conditions:

The answers are submitted to the system, to be checked automatically/by the teacher.

Main Success scenario:

1. The pupil is finished reading a text, chooses to go to Questions Phase.
2. All questions regarding the text are displayed to the Pupil for answering.

Alternative scenarios:

2.1) Some error regarding the loading of the questions occurs, appropriate error message is displayed.

Use case ID: #6

Use case name: **Receiving feedback for the number of words in an answer**

Associated Requirement IDs: #6

Actors: Pupil

Description:

After a Pupil finishes answering a question and chooses to "Check" the answer, he receives feedback for the amount of words in the answer (too many words/too little words).

Pre-conditions:

Pupil is answering a question.

Main Success scenario:

1. The pupil is writing an answer to a question after reading a text.
2. The pupil clicks a button to check his answer after he is done.
3. Feedback for the amount of words in his answer is displayed to him.

Alternative scenarios:

3.1) Some error regarding the processing of the answer occurs, appropriate error message is displayed.

Use case ID: #7

Use case name: **Amount of words in an answer is limitable by the uploader of the text**

Associated Requirement IDs: #7

Actors: Pupil

Description:

While answering a question, the Pupil can see the lower and upper bounds of the amount of words needed to submit that answer.

Pre-conditions:

Pupil is answering a question.

Main Success scenario:

1. The pupil is writing an answer to a question after reading a text.
2. The limit for the amount of words in his answer is displayed to him, while he is writing the answer.

Alternative scenarios:

2.1) Some error regarding the processing of the answer occurs, appropriate error message is displayed.

Use case ID: #8

Use case name: **Receiving feedback for the number of connectors in an answer**

Associated Requirement IDs: #8

Actors: Pupil

Description:

After a Pupil finishes answering a question and chooses to "Check" the answer, he receives feedback for the number of connectors in the answer (too many/too little).

Pre-conditions:

Pupil is answering a question.

Main Success scenario:

1. The pupil is writing an answer to a question after reading a text.
2. The pupil clicks a button to check his answer after he is done.
3. Feedback for the number of connectors in his answer is displayed to him.

Alternative scenarios:

3.1) Some error regarding the processing of the answer occurs, appropriate error message is displayed.

Use case ID: #9

Use case name: **Receiving suggestions for key sentences to start an answer**

Associated Requirement IDs: #9

Actors: Pupil

Description:

While answering a question, the Pupil can get a suggestion for an opening sentence for his answer.

Pre-conditions:

Pupil is answering a question.

Main Success scenario:

1. The pupil is writing an answer to a question after reading a text.
2. The pupil clicks a button to receive suggestions for an opening sentence for his answer.
3. Some suggestions for an opening sentence are displayed to him.

Alternative scenarios:

2.1) Some error occurs, appropriate error message is displayed.

Use case ID: #10

Use case name: **Receiving feedback for repeated words in an answer**

Associated Requirement IDs: #10

Actors: Pupil

Description:

After a Pupil finishes answering a question and chooses to "Check" the answer, he receives feedback for repeated words in his answer, if there are any such.

Pre-conditions:

Pupil is answering a question.

Main Success scenario:

1. The pupil is writing an answer to a question after reading a text.
2. The pupil clicks a button to check his answer after he is done.
3. Feedback for repeated words in his answer, if there are any, is displayed.

Alternative scenarios:

3.1) Some error regarding the processing of the answer occurs, appropriate error message is displayed.

Use case ID: #10.1

Use case name: **Receiving suggestions for alternatives for repeated words in an answer**

Associated Requirement IDs: #11

Actors: Pupil

Description:

After a Pupil receives feedback for repeated words in his answer, he also receives suggestions for some alternatives for repeated words in his answer

Pre-conditions:

Pupil is checking his answer.

Main Success scenario:

1. Feedback for repeated words in an answer, if there are any, is displayed.
2. Suggestions for alternate words to be used instead of the repeated words are displayed.

Alternative scenarios:

2.1) Some error regarding the processing of the answer occurs, appropriate error message is displayed.

Use case ID: #11

Use case name: **Exploring related online games via Hyperlinks**

Associated Requirement IDs: #12

Actors: Pupil, Teacher

Description:

Users can be referred to related online games through hyperlinks available in the system.

Pre-conditions:

User is logged in.

Post-conditions:

The selected online game is loaded for the user.

Main Success scenario:

1. The user selects an external online game from a list of available games.
2. The user clicks the related Hyperlink, is directed to the game.

Alternative scenarios:

2.1) link is broken or URL is not found, appropriate error message is displayed.

Use case ID: #12

Use case name: **Viewing accumulated scores in the assorted online games**

Associated Requirement IDs: #13

Actors: Pupil

Description:

Pupils can view their scores in the online games available.

Pre-conditions:

Pupil is logged in.

Main Success scenario:

1. The pupil selects an external online game from a list of available games.
2. The accumulated score of the selected game is displayed.

Use case ID: #13

Use case name: **Tracking progress in the assorted online games**

Associated Requirement IDs: #14

Actors: Pupil

Description:

Pupils can track their progress over time in the online games.

Pre-conditions:

Pupil is logged in.

Main Success scenario:

1. The user selects an external online game from a list of available games.
2. The pupil clicks on a button to track his progress
3. A graph for the pupil's progress in the game is displayed

Alternative scenarios:

2.1) Some error occurs, appropriate error message is displayed.

Use case ID: #14

Use case name: **Tracking progress of tasks**

Associated Requirement IDs: #15

Actors: Pupil, Teacher

Description:

Users can track their progress in their current active **texts**.

Pre-conditions:

User is logged in.

Main Success scenario:

1. The user selects a text to track.
2. Progress in that text is displayed through a graph and the number of remaining questions to be answered.

Use case ID: #16

Use case name: **Ability to view scores of other pupils.**

Associated Requirement IDs: #16

Actors: Pupil, Teacher

Description:

Users can compare their own scores in the various online games with all other pupil's scores in the same class as them (?)

Pre-conditions:

User is logged in.

Main Success scenario:

1. The user selects a game score to compare.
2. Other pupils score in this game is represented to the screen.

Alternative scenarios:

2.1) Some error occurs, appropriate error message is displayed.

Use case ID: #17

Use case name: **Check button for feed backing all the answer's parameters.**

Associated Requirement IDs: #17

Actors: Pupil.

Description:

Pupils can get a feedback on their answer before they send it by the system. All by the parameters of the answer they give. The system should correct them and give them tools for better answer.

Pre-conditions:

User is logged in.

Main Success scenario:

1. The pupil selects to answer questions on some text.
2. Before the pupil sends it he clicks the check button and gets a feedback on his answer.

Alternative scenarios:

2.1) Some error occurs, appropriate error message is displayed.

Use case ID: #18

Use case name: **Marking and correcting sequence of repeated words.**

Associated Requirement IDs: #18

Actors: Pupil

Description:

Pupils get alerted by the system (by marking it) when it recognize repeated words and suggesting the pupil to fix it.

Pre-conditions:

Pupil is writing an answer.

Main Success scenario:

1. The pupil typing his answer in the right place and repeating the same word in sequence.
2. The system marks it and alerts it to the pupil.

Alternative scenarios:

2.1) the pupil typing same word in sequence and the system doesn’t catch it.

Use case ID: #19

Use case name: **New texts could be added to the system.**

Associated Requirement IDs: #19

Actors: Teacher

Description:

Teachers can add texts to the system at any time.

Pre-conditions:

User is logged in.

Main Success scenario:

1. The user selects to upload new text.
2. After entering all the right fields, upload and the correct pupils can view it.

Alternative scenarios:

2.1) Some error occurs, text didn’t upload.

Use case ID: #20

Use case name: **Existing texts could be edited.**

Associated Requirement IDs: #19

Actors: Teacher

Description:

A teacher edits text that is already in the system.

Pre-conditions:

User is logged in.

Main Success scenario:

1. The user selects a to edit existing text.
2. After edit all the right fields, upload the correct text and the correct pupils can view it.

Alternative scenarios:

2.1) Some error occurs, text didn’t uploaded.

Use case ID: #21

Use case name: **Existing texts could be removed from the system.**

Associated Requirement IDs: #19

Actors: Teacher

Description:

A teacher removes text from the system

Pre-conditions:

User is logged in.

Main Success scenario:

1. The user selects to remove old text.
2. After removing, the text will be deleted from the system.

Alternative scenarios:

2.1) Some error occurs, text didn’t removed.

Use case ID: #22

Use case name: **New audio could be added to the system.**

Associated Requirement IDs: #22

Actors: Teacher

Description:

A teacher uploads new audio file to the system

Pre-conditions:

User is logged in.

Main Success scenario:

1. The user selects to upload new audio file for some text.
2. After entering all the right fields, upload and the correct pupils can play it.

Alternative scenarios:

2.1) Some error occurs, audio didn’t uploaded.

Use case ID: #23

Use case name: **Existing audio could be replaced.**

Associated Requirement IDs: #22

Actors: Teacher

Description:

A teacher edits audio that is already in the system.

Pre-conditions:

User is logged in.

Main Success scenario:

1. The user selects to replace existing audio file.
2. After replace, upload the correct audio file and the correct pupils can play it.

Alternative scenarios:

2.1) Some error occurs, new audio file didn’t uploaded.

Use case ID: #24

Use case name: **Existing audio could be removed from the system.**

Associated Requirement IDs: #22

Actors: Teacher

Description:

A teacher removes audio from the system

Pre-conditions:

User is logged in.

Main Success scenario:

1. The user selects to remove old audio file.
2. After removing, the audio will be deleted from the system.

Alternative scenarios:

2.1) Some error occurs, audio file didn’t removed.

Use case ID: #25

Use case name: **New words definitions could be added to the system.**

Associated Requirement IDs: #25

Actors: Teacher

Description:

A teacher uploads new word definition to the system

Pre-conditions:

User is logged in.

Main Success scenario:

1. The user selects to upload new word definition to a text.
2. After entering the word definition, upload and the pupils can view it.

Alternative scenarios:

2.1) Some error occurs, word definition didn’t uploaded.

Use case ID: #26

Use case name: **Existing word definitions could be edited.**

Associated Requirement IDs: #25

Actors: Teacher

Description:

A teacher wants to edit word definition that is already in the system.

Pre-conditions:

User is logged in.

Main Success scenario:

1. The user selects to edit existing word definition.
2. After edit the right fields, upload the correct word definition and the correct pupils can view it.

Alternative scenarios:

2.1) Some error occurs, new word definition didn’t uploaded.

Use case ID: #27

Use case name: **Existing word definitions could be removed from the system.**

Associated Requirement IDs: #25

Actors: Teacher

Description:

A teacher removes word definition from the system

Pre-conditions:

User is logged in.

Main Success scenario:

1. The user selects to remove old word definition.
2. After removing, the word definition will be deleted from the system.

Alternative scenarios:

2.1) Some error occurs, word definition didn’t removed.

Use case ID: #28

Use case name: **New questions could be added to the system.**

Associated Requirement IDs: #28

Actors: Teacher

Description:

A teacher uploads new question to the system

Pre-conditions:

User is logged in.

Main Success scenario:

1. The user selects to upload new question to a text.
2. After entering the question, upload and the pupils can view it.

Alternative scenarios:

2.1) Some error occurs, new question didn’t uploaded.

Use case ID: #29

Use case name: **Existing questions could be edited.**

Associated Requirement IDs: #28

Actors: Teacher

Description:

A teacher wants to edit question that is already in the system.

Pre-conditions:

User is logged in.

Main Success scenario:

1. The user selects to edit existing question.
2. After edit the right fields, upload the correct question and the correct pupils can view it.

Alternative scenarios:

2.1) Some error occurs, new question didn’t uploaded.

Use case ID: #30

Use case name: **Existing questions could be removed from the system.**

Associated Requirement IDs: #28

Actors: Teacher

Description:

A teacher removes question from the system

Pre-conditions:

User is logged in.

Main Success scenario:

1. The user selects to remove old question.
2. After removing, the question will be deleted from the system.

Alternative scenarios:

2.1) Some error occurs, question didn’t removed.

Use case ID: #31

Use case name: **New key sentences could be added**

Associated Requirement IDs: #31

Actors: Teacher

Description:

Teachers can add a new key sentence related to a question, to be offered for the pupils.

Pre-conditions:

Teacher is logged in. A question exists in the system.

Post-conditions:

A key sentence is added to the system, related to a question.

Main Success scenario:

1. The teacher selects a question and chooses the option to add key sentence.
2. The teacher writes a key sentence and chooses to add it.
3. A success message displayed to the teacher.

Alternative scenarios:

2.1) some error occurs, appropriate error message is displayed.

Use case ID: #32

Use case name: **Existing key sentences could be edited**

Associated Requirement IDs: #32

Actors: Teacher

Description:

Teachers can edit an existing key sentence.

Pre-conditions:

Teacher is logged in. A key sentence exists in the system.

Post-conditions:

The chosen key sentence has changed and saved in the system.

Main Success scenario:

1. The teacher selects a key sentence and chooses the option to edit it.
2. The key sentence is represented with the option to edit.
3. The teacher edits it and chooses to save.
4. A success message displayed to the teacher.

Alternative scenarios:

3.1) some error occurs, appropriate error message is displayed.

Use case ID: #33

Use case name: **Existing key sentences could be removed**

Associated Requirement IDs: #33

Actors: Teacher

Description:

Teachers can delete an existing key sentence from the system.

Pre-conditions:

Teacher is logged in. A key sentence exists in the system.

Post-conditions:

The chosen key sentence has been removed from the system.

Main Success scenario:

1. The teacher selects a key sentence and chooses the option to delete it.
2. The teacher is asked whether he is sure and clicks 'yes'.
3. A success message displayed to the teacher.

Alternative scenarios:

2.1) the teacher clicks 'no' and he is back to the former window without changes.

3.1) some error occurs, appropriate error message is displayed.

Use case ID: #34

Use case name: **Feedback for punctuation marks usage**

Associated Requirement IDs: #34

Actors: Pupil

Description:

Pupils can have feedback for some missing punctuation marks or wrong once.

Pre-conditions:

Pupil is logged in.

Main Success scenario:

1. The pupil has written an answer and clicks the 'check' button.
2. Missing punctuation marks or wrong once are being represented to the pupil.

Alternative scenarios:

* 1. Some error occurs, appropriate error message is displayed.

Use case ID: #35

Use case name: **Answers can be checked and feed backed by the teacher**

Associated Requirement IDs: #35

Actors: Teacher

Description:

Teacher can view answers of pupils who are connected to a same class as him, check them and deliver a feedback, which the student will see upon logging in to the system.

Pre-conditions:

Teacher is logged in. The teacher and the pupil are connected to the same class.

Post-conditions:

The feedback is saved in the system.

Main Success scenario:

1. The teacher selects a pupil's answer and chooses to give feedback.
2. The answer is represented to the teacher and he can add feedback.
3. A success message displayed to the teacher.

Alternative scenarios:

2.1) some error occurs, appropriate error message is displayed.

Use case ID: #36

Use case name: **Control over font and display - size, color, underline, bold**

Associated Requirement IDs: #36

Actors: Pupil

Description:

While writing an answer, pupil could choose some text to have a specific font, size, color, underline or bold.

Pre-conditions:

Pupil is logged in. question exists in the system, such that the pupil is allowed to answer.

Main Success scenario:

1. The pupil chooses to answer a question.
2. The pupil may click a button from a menu (font, size, color, underline, bold).
3. The button is market as clicked and the pupil's text is according to it.

Alternative scenarios:

2.1) the button is already clicked, so the click cancels its effect.

Use case ID: #37

Use case name: **New pupils users could be added to system**

Associated Requirement IDs: #37

Actors: Teacher

Description:

Teacher can add new pupils to the system.

Pre-conditions:

Teacher is logged in.

Post-conditions:

A new pupil is created in the system. A password is generated for the pupil.

Main Success scenario:

1. The teacher chooses to add a new pupil.
2. The teacher fills up the pupil's details (name, id, email, class) and clicks 'Add'.
3. A success message is represented to the teacher.

Alternative scenarios:

2.1) pupil's details are illegal or some other error occurred and appropriate error message is represented to the user.

Use case ID: #38

Use case name: **Existing pupils' user's details could be edited**

Associated Requirement IDs: #38

Actors: Teacher

Description:

Teacher can edit an existing pupil's details.

Pre-conditions:

Teacher is logged in. A pupil exists in the system.

Post-conditions:

The pupil's details has changed and saved in the system.

Main Success scenario:

1. The teacher chooses a pupil and chooses to edit his details.
2. The current details are represented to the teacher with the option to edit.
3. The teacher edits the details and clicks 'save'.
4. A success message is represented to the teacher.

Alternative scenarios:

3.1) pupil's details are illegal or some other error occurred and appropriate error message is represented to the user.

Use case ID: #39

Use case name: **Existing pupils' users could be removed from the system**

Associated Requirement IDs: #39

Actors: Teacher

Description:

Teacher can remove an existing pupil from the system.

Pre-conditions:

Teacher is logged in. A pupil exists in the system.

Post-conditions:

The pupil has removed from the system.

Main Success scenario:

1. The teacher chooses a pupil and chooses to remove him.
2. The teacher is asked whether he is sure and clicks 'yes'.
3. A success message is represented to the teacher.

Alternative scenarios:

2.1) the teacher clicks 'no' and back to the last page without any changes in the system.

2.2) some error occurred and an appropriate error message is represented to the user.

Use case ID: #40

Use case name: **New teachers' users could be added to the system**

Associated Requirement IDs: #40

Actors: Teacher

Description:

Teacher with administrator permission can add new teacher to the system.

Pre-conditions:

Teacher is logged in. The teacher holds administrator permission.

Post-conditions:

New teacher has added and saved in the system.

Main Success scenario:

1. The teacher chooses to add a new teacher.
2. The teacher fills up the new teacher's details (name, id, classes and email) and clicks 'add'.
3. A success message is represented to the teacher.

Alternative scenarios:

2.1) some error occurred and an appropriate error message is represented to the user.

2.2) the teacher clicks 'cancel' and the last page is represented without any changes in the system.

Use case ID: #41

Use case name: **Existing teachers' user's details could be edited**

Associated Requirement IDs: #41

Actors: Teacher

Description:

Teacher with administrator permission can edit an existing teacher's details.

Pre-conditions:

Teacher is logged in. The teacher holds administrator permission.

Post-conditions:

The teacher's details has changed and saved in the system.

Main Success scenario:

1. The teacher chooses to an existing teacher and chooses to edit his details.
2. The current details are represented to the user with the option to edit.
3. The teacher edits the teacher's details and clicks 'save'.
4. A success message is represented to the teacher.

Alternative scenarios:

3.1) some error occurred and an appropriate error message is represented to the user.

3.2) the teacher clicks 'cancel' and the last page is represented without any changes in the system.

Use case ID: #42

Use case name: **Existing teachers' users could be deleted from the system**

Associated Requirement IDs: #42

Actors: Teacher

Description:

Teacher with administrator permission can delete an existing teacher from the system.

Pre-conditions:

Teacher is logged in. The teacher holds administrator permission.

Post-conditions:

The teacher has removed from the system.

Main Success scenario:

1. The teacher chooses to an existing teacher and chooses to delete him.
2. The teacher is asked whether he is sure and clicks 'yes'.
3. A success message is represented to the teacher.

Alternative scenarios:

2.1) some error occurred and an appropriate error message is represented to the user.

2.2) the teacher clicks 'no' and the last page is represented without any changes in the system.

Use case ID: #43

Use case name: **Existing teachers could be appointed as administrators**

Associated Requirement IDs: #43

Actors: Teacher

Description:

Teacher with administrator permission can appoint a teacher to be administrator.

Pre-conditions:

Teacher is logged in. The teacher holds administrator permission.

Post-conditions:

The teacher now holds administrator permissions.

Main Success scenario:

1. The teacher chooses to an existing teacher and chooses to appoint him administrator.
2. The teacher is asked whether he is sure and clicks 'yes'.
3. A success message is represented to the teacher.

Alternative scenarios:

2.1) some error occurred and an appropriate error message is represented to the user.

2.2) the teacher clicks 'no' and the last page is represented without any changes in the system.

Use case ID: #44

Use case name: **New admin teachers could be added to the system**

Associated Requirement IDs: #44

Actors: Teacher

Description:

Teacher with administrator permission can add a new teacher with administrator permissions to the system.

Pre-conditions:

Teacher is logged in. The teacher holds administrator permission.

Post-conditions:

A new teacher with administrator permissions has added to the system.

Main Success scenario:

1. The teacher chooses to add a new teacher.
2. The teacher fills up the new teacher's details (name, id, classes and email), marks the teacher as 'administrator' and clicks 'add'.
3. A success message is represented to the teacher.

Alternative scenarios:

2.1) some error occurred and an appropriate error message is represented to the user.

2.2) the teacher clicks 'cancel' and the last page is represented without any changes in the system.

Use case ID: #45

Use case name: **Pupils could log in to the system.**

Associated Requirement IDs: #45

Actors: pupil

Description:

Pupil want to log in to the system.

Pre-conditions:

The pupil exists in the system.

Post-conditions:

Pupil is logged in to the system.

Main Success scenario:

1. The pupil entered his details and click on the log in button.
2. The system verifies the details and then the pupil gets logged in.

Alternative scenarios:

2.1) some error occurs, appropriate error message is displayed.

2.2) wrong details entered, appropriate error message is displayed.

Use case ID: #46

Use case name: **Pupils could log out of the system.**

Associated Requirement IDs: #45

Actors: pupil

Description:

Pupil wants to log out from the system.

Pre-conditions:

The pupil is logged in to the system.

Post-conditions:

Pupil is logged out of the system.

Main Success scenario:

1. The pupil click on the log out button.
2. The system logs him out.

Alternative scenarios:

2.1) some error occurs, appropriate error message is displayed, pupil still logged in.

Use case ID: #47

Use case name: **Teachers could log in to the system.**

Associated Requirement IDs: #47

Actors: Teacher

Description:

Teacher wants to log in to the system.

Pre-conditions:

The Teacher exists in the system.

Post-conditions:

Teacher is logged in to the system.

Main Success scenario:

1. The Teacher entered his details and click on the log in button.
2. The system verifies the details and then the Teacher gets logged in.

Alternative scenarios:

2.1) some error occurs, appropriate error message is displayed.

2.2) wrong details entered, appropriate error message is displayed.

Use case ID: #48

Use case name: **Teachers could log out of the system.**

Associated Requirement IDs: #47

Actors: teacher

Description:

Teacher wants to log out from the system.

Pre-conditions:

The teacher is logged in to the system.

Post-conditions:

Teacher is logged out of the system.

Main Success scenario:

1. The teacher click on the log out button.
2. The system logs him out.

Alternative scenarios:

2.1) some error occurs, appropriate error message is displayed, teacher still logged in.

Use case ID: #49

Use case name: **Answer could be submitted.**

Associated Requirement IDs: #49

Actors: pupil.

Description:

Pupil wants to submit answer on a question.

Pre-conditions:

The pupil is logged in to the system.

Post-conditions:

The Pupils Answer submitted to the system, correct teachers can view it.

Main Success scenario:

1. The pupil fill the correct answer field and click on the submit button.
2. The system submits his answer.

Alternative scenarios:

2.1) some error occurs, appropriate error message is displayed, the answer didn’t submit.

Use case ID: #50

Use case name: **Pupil will get a notification when he got a feedback.**

Associated Requirement IDs: #50

Actors: pupil, teacher.

Description:

After the teacher sends a pupil a feedback over an answer, the Pupil will get notified upon his next log-in to the system.

Pre-conditions:

The Teacher feed backing the pupil answer.

Post-conditions:

The Pupils can view this feedback.

Main Success scenario:

1. After the teacher gives his feedback, the pupil logged in and get notify by the system over it.

Alternative scenarios:

2.1) some error occurs, the student didn’t get notify on the feedback.

Use case ID: #51

Use case name: **Pupils can be connected to a class**

Associated Requirement IDs: #51

Actors: Teacher.

Description:

A teacher can make the connection between a pupil and a class.

Pre-conditions:

The teacher is logged in. A pupil exists in the system. A class exists in the system.

Post-conditions:

The pupil is connected to the class.

Main Success scenario:

1. The teacher chooses a pupil and chooses to connect him to a class.
2. The teacher chooses a class to connect the pupil to.
3. A success message is represented to the teacher.

Alternative scenarios:

2.1) some error occurs and appropriate error message is represented to the user.

Use case ID: #52

Use case name: **Teachers can be connected to a class**

Associated Requirement IDs: #52

Actors: Teacher.

Description:

A teacher with administrator permissions can make the connection between a teacher and a class.

Pre-conditions:

The teacher is logged in. The teacher holds administrator permissions. A pupil exists in the system. A class exists in the system.

Post-conditions:

The teacher is connected to the class.

Main Success scenario:

1. The teacher chooses a teacher and chooses to connect him to a class.
2. The teacher chooses a class to connect the teacher to.
3. A success message is represented to the teacher.

Alternative scenarios:

2.1) some error occurs and appropriate error message is represented to the user.

Use case ID: #53

Use case name: **Pupils can be disconnected from a class**

Associated Requirement IDs: #53

Actors: Teacher.

Description:

A teacher can disconnect a pupil from a class.

Pre-conditions:

The teacher is logged in. The pupil is connected to a class.

Post-conditions:

The pupil is disconnected from the class.

Main Success scenario:

1. The teacher chooses a pupil and chooses to disconnect him to a class.
2. The teacher chooses a class to disconnect the pupil from.
3. A success message is represented to the teacher.

Alternative scenarios:

2.1) some error occurs and appropriate error message is represented to the user.

Use case ID: #54

Use case name: **Teachers can be disconnected from a class**

Associated Requirement IDs: #54

Actors: Teacher.

Description:

A teacher with administrator permissions can disconnect a teacher from a class.

Pre-conditions:

The teacher is logged in. The teacher holds administrator permissions. A teacher is connected to a class.

Post-conditions:

The teacher is disconnected from the class.

Main Success scenario:

1. The teacher chooses a teacher and chooses to disconnect him to a class.
2. The teacher chooses a class to disconnect the teacher from.
3. A success message is represented to the teacher.

Alternative scenarios:

2.1) some error occurs and appropriate error message is represented to the user.

Use case ID: #55

Use case name: **A class can be created**

Associated Requirement IDs: #55

Actors: Teacher.

Description:

A teacher can create a new class in the system.

Pre-conditions:

The teacher is logged in.

Post-conditions:

A new class has added to the system.

Main Success scenario:

1. The teacher chooses to add a new class.
2. The teacher fills up the class details (grade, name, subject of the class) and clicks 'create'.
3. A success message is represented to the teacher.

Alternative scenarios:

2.1) some error occurs and appropriate error message is represented to the user.

2.2) the teacher clicks 'cancel' and the last page is represented without any changes in the system.

Use case ID: #56

Use case name: **Class details can be edited**

Associated Requirement IDs: #56

Actors: Teacher.

Description:

A teacher can edit an existing class details.

Pre-conditions:

The teacher is logged in. A class exists in the system.

Post-conditions:

The class details has changed and saved in the system.

Main Success scenario:

1. The teacher chooses a class and chooses to edit it.
2. The current details are represented with the option to edit.
3. The teacher edits the details and clicks 'save'.
4. A success message is represented to the teacher.

Alternative scenarios:

3.1) some error occurs and appropriate error message is represented to the user.

3.2) the teacher clicks 'cancel' and the last page is represented without any changes in the system.

Use case ID: #57

Use case name: **A class can be removed**

Associated Requirement IDs: #57

Actors: Teacher.

Description:

A teacher can delete an existing class from the system.

Pre-conditions:

The teacher is logged in. A class exists in the system.

Post-conditions:

The class has removed from the system.

Main Success scenario:

1. The teacher chooses a class and chooses to delete it.
2. The teacher is asked whether he is sure and clicks 'yes'.
3. A success message is represented to the teacher.

Alternative scenarios:

2.1) some error occurs and appropriate error message is represented to the user.

2.2) the teacher clicks 'no' and the last page is represented without any changes in the system.

Use case ID: #58

Use case name: **Pupil gets notified when a new task (=text) is available for** **solving**

Associated Requirement IDs: #58

Actors: Pupil

Description:

When a Pupil gets associated with a text, he shall receive an appropriate notification within the system upon his next login.

Pre-conditions:

The Pupil is a member of a class.

Main Success scenario:

1. The teacher uploads a new text, associates it with a class.
2. All pupils in that class receive an appropriate notification upon their next login.

Alternative scenarios:

2.1) some error occurs and appropriate error message is presented to the user.

Use case ID: #59

Use case name:  **A list of all classes in the system can be viewed**

Associated Requirement IDs: #59

Actors: Teacher

Description:

Teachers can be presented with a list of all classes in the system, along with details about the classes – number of students, what grade that class is in, participants in that class (upon clicking an entry the teacher will be redirected to a page dedicated to the class and it's details), and any other details required.

Pre-conditions:

Teacher is logged in

Main Success scenario:

1. The teacher clicks on the "Class list" button
2. The teacher is presented with the list of classes.

Alternative scenarios:

2.1) some error occurs and appropriate error message is presented to the user.

Use case ID: #60

Use case name:  **A list of all pupils in a class can be viewed**

Associated Requirement IDs: #60

Actors: Teacher, Pupil

Description:

Teachers and Pupils can be presented with a list of all pupils attending a class, by getting to the class' page. (for pupils- the only class page available to them is their own).

Pre-conditions:

User is logged in, is in a class' page

Main Success scenario:

1. The user clicks on the "Pupil list" button
2. The user is presented with the list of pupils in that class.

Alternative scenarios:

2.1) some error occurs and appropriate error message is presented to the user.

Use case ID: #61

Use case name:  **A list of all teachers in a class can be viewed**

Associated Requirement IDs: #61

Actors: Teacher, Pupil

Description:

Teachers and Pupils can be presented with a list of all teachers teaching a class, by getting to the class' page. (for pupils- the only class page available to them is their own).

Pre-conditions:

User is logged in, is in a class' page

Main Success scenario:

1. The user clicks on the "Teacher list" button
2. The user is presented with the list of teachers teaching that class.

Alternative scenarios:

2.1) some error occurs and appropriate error message is presented to the user.

Use case ID: #62

Use case name:  **A pupil's details can be viewed by any teacher teaching him**

Associated Requirement IDs: #62

Actors: Teacher

Description:

Teachers can access the details of pupils whom they teach, by getting to the class page of the class in which the pupil is enrolled, clicking on "Pupil list" – and clicking on the desired Pupil.

Pre-conditions:

Teacher is logged in, is in a class page.

Main Success scenario:

1. The teacher clicks on the "Pupil list"
2. The teacher clicks on a certain pupil.
3. The teacher is presented with the Details page for that pupil.

Alternative scenarios:

2.1) some error occurs and appropriate error message is presented to the user.

3.1) some error occurs and appropriate error message is presented to the user.

Use case ID: #63

Use case name:  **A pupil's details can be viewed by himself**

Associated Requirement IDs: #63

Actors: Pupil

Description:

Pupils can access their own details page.

Pre-conditions:

Pupil is logged in.

Main Success scenario:

1. The Pupil clicks on the "My Profile" button.
2. The Pupil is presented with his details

Alternative scenarios:

2.1) some error occurs and appropriate error message is presented to the user.

Use case ID: #64

Use case name:  **A teacher's details can be viewed by all teachers in the system**

Associated Requirement IDs: #64

Actors: Teacher

Description:

Teachers can access the details of all other teachers in the system incl. their own.

Pre-conditions:

Teacher is logged in.

Main Success scenario:

1. The teacher clicks on "Teacher list"
2. The teacher clicks on a certain teacher.
3. The teacher is presented with the Details page for that teacher.

Alternative scenarios:

* 1. The teacher gets to a class page
     1. The teacher clicks on "teacher list" to get a list for the teachers of that class

1.1.1.1) The teacher clicks on a certain teacher to get their details.

2.1) some error occurs and appropriate error message is presented to the user.

3.1) some error occurs and appropriate error message is presented to the user.

Use case ID: #65

Use case name:  **A teacher's details can be viewed by all their pupils**

Associated Requirement IDs: #65

Actors: Pupil

Description:

Pupils can access their teachers' details

Pre-conditions:

Pupil is logged in.

Main Success scenario:

1. The Pupil gets to his class' page.
2. The Pupil clicks "Teacher List"
3. The Pupil clicks on a certain teacher to get to their details page

Alternative scenarios:

2.1) some error occurs and appropriate error message is presented to the user.

Use case ID: #66

Use case name:  **A list of all teachers in the system can be viewed**

Associated Requirement IDs: #66

Actors: Teacher

Description:

Teachers can be presented with a list of all teachers in the system.

Pre-conditions:

Teacher is logged in

Main Success scenario:

1. The teacher clicks on the "Teacher list" button
2. The teacher is presented with the list of teachers in the system.

Alternative scenarios:

2.1) some error occurs and appropriate error message is presented to the user.

Use case ID: #66

Use case name:  **A list of all pupils in the system can be viewed**

Associated Requirement IDs: #66

Actors: Teacher

Description:

Teachers can be presented with a list of all pupils in the system.

Pre-conditions:

Teacher is logged in

Main Success scenario:

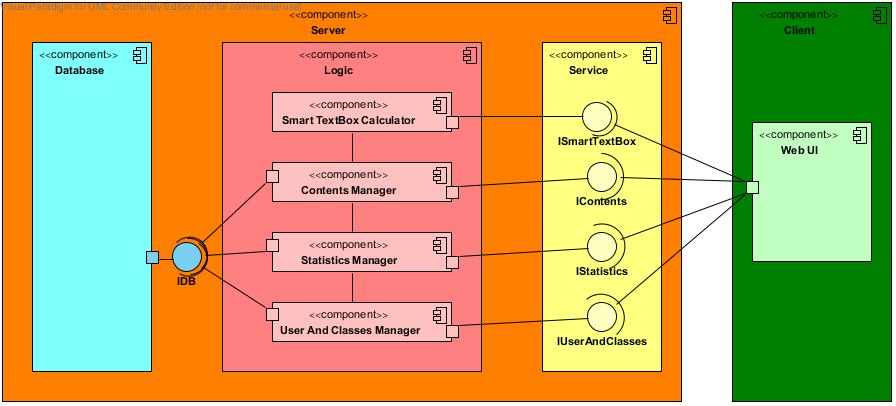
1. The teacher clicks on the "Pupil list" button
2. The teacher is presented with the list of all pupils in the system.

Alternative scenarios:

2.1) some error occurs and appropriate error message is presented to the user.

**2. System Architecture**

2.1 Architecture diagram



2.2 Components description

**Client:**

**Web UI:**

The client application. Contains all of the graphics: menus, contents presentation (texts, online games, smart text box, etc). This is the implementation of the mockups from the ARD.

**Server**

**Service:**

* **ISmartTextBox:** the interface of the smart textbox functionality (getting key sentences, answer check, etc).
* **IContents:** an interface for contents edits (upload, remove, etc). This interface is for teachers use.
* **IStatistics:** the interface for getting statistics over pupils progress (getting tasks progress, online games scores, etc).
* **IUserAndClasses:** this interface is used for managing users and classes in the system (add user, add class, connect between a pupil and a class, etc).

**Logic:**

* **Smart TextBox Calculator:** responsible for the logic and calculations behind the functionality of the smart text box. Implements ISmartTextBox.
* **Contents Manager:** responsible for the contents pulling from the Database (texts, audios, etc) and for the contents managing and delivering. Implements IContents.
* **Statistics Manager:** responsible for all of the statistics calculations, including data collecting for the computations. Implements IStatistics.
* **User and Classes Manager:** responsible for pulling the users and classes information from the Database, as well as to manage it while doing all of the needed computations. Implements IUserAndClasses.

**Database:**

* **IDB:** the interface providing all of the needed functions from Database, including data pulling and pushing and all of the needed queries.

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |

**3. Data Model**

3.1 Data diagrams

Users and Classes:

Users & Class CD.png

Contents:

Text Heirarchy CD.png

3.2 Description of data objects

Users and Classes:

* **User (abstract):** an abstract entity, represent a user, which can be one of the two types of the system's users. Gathering shared information for both users, such as name, id, etc.
* **Teacher:** represents a teacher user and has information related to a teacher only, such as classes list and some permissions given to a teacher only.
* **Pupil:** represents a pupil user and has information related to a pupil only, such as tasks and some permissions given to a pupil only.
* **Class:** this is a connector entity, which connects between a teacher and a pupil. Teacher and pupil who share a same class are able to connect with each other.
* **Subject:** this is an entity which represents the different class' subjects in the system, the subjects taught at school. There might be sub-subjects.

Contents:

* **Text:** represents a reading material uploaded by the teacher, which is a story or some other text used for teaching.
* **Task:** a batch of questions given by the teacher as a task.
* **Question:** a textual question given by the teacher, wrapped by a task.
* **Policy:** this entity defines rules related to a specific question. The rules may include limits for number of words use, number of connector words, etc.
* **Answer:** a textual answer given by a pupil related to a specific question.

3.3 Data objects relationships

Users and Classes:

There are two kinds of users – teacher and pupil. They both inherit from an abstract class – User, holding the shared information of both of them.

A pupil can be connected to one class only, but a teacher can be connected to more than one class, according to the classes he teaches, or has permissions to.

A class can be connected to subjects, which define the subjects the class study (science, history, etc). a subject might have a sub-subject, for example: history may have a sub-subject history of Israel.

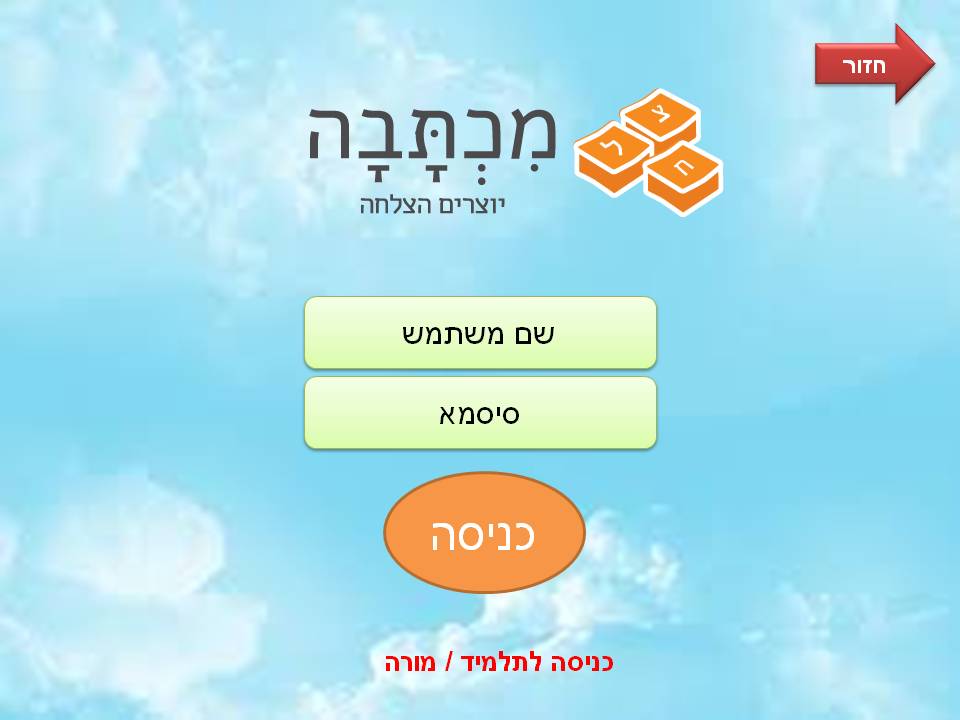
Contents:

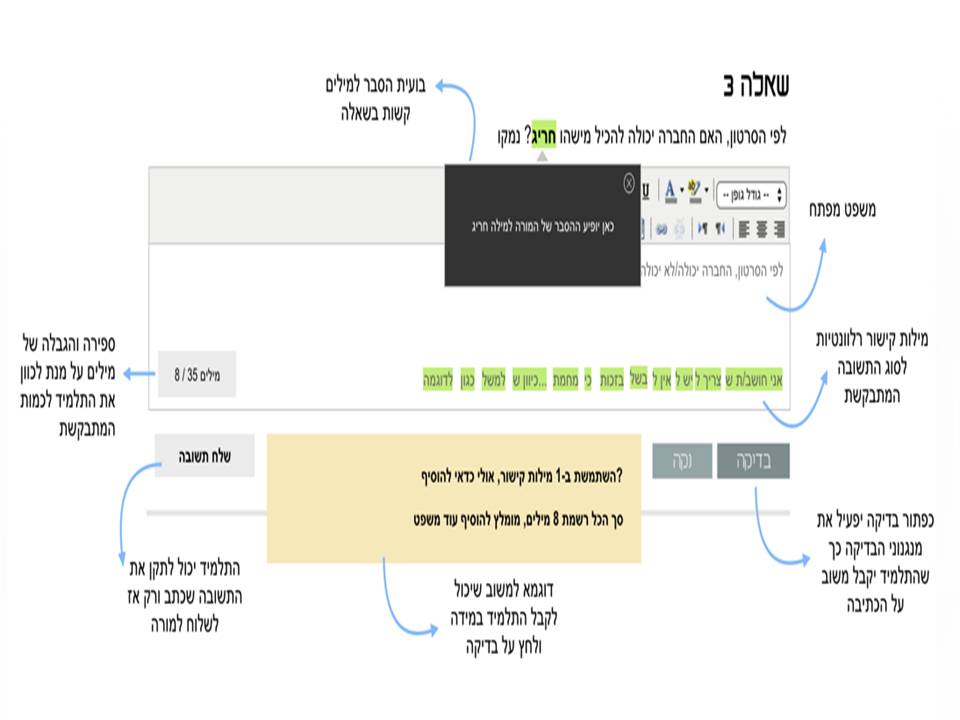
Texts might have tasks related to them and therefore connected to tasks. Task is a collection of questions related to a text, and therefore has a connection with the questions.

Each question has a policy of rules for answering it and therefore connected to a policy defining these rules.

In addition, every question answered by a pupil trigger a new answer object, which represents the pupil's answer and contains details about it.

**4. User Interface Draft**





**5. Testing**

5.1 Functional Testing

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tested Component | Testing Component | Requirement ID | Is Automatic? | |
| Smart Text Box | Smart Text Box Tests | 6 | Yes | |
| 8 | Yes | |
| 9 | Yes | |
| 10 | Yes | |
| 11 | Yes | |
| 17 | Yes | |
| 18 | Yes | |
| 34 | Yes | |
| Contents Manager | Contents Manager Tests | 2 | No | |
| 3 | No | |
| 5 | Yes | |
| 7 | Yes | |
| 12 | No | |
| 19-33 | Yes | |
| 35 | Yes | |
| 49 | Yes | |
| 50 | Yes | |
| 58 | Yes | |
|  |  |  |
|  |  | |
| Statistics Manager | Statistics Manager Tests | 13 | Yes | |
| 14 | Yes | |
| 15 | Yes | |
| 16 | Yes | |
| Users Manager | Users Manager Tests | 37-48 | Yes | |
| 51-57 | Yes | |
| Frontend | None | 36 | No | |
| 59-67 | No | |
| 1 | No | |
| 4 | No | |

5.2 Non-Functional Testing

|  |  |
| --- | --- |
| Requirement ID | Is Automatic? |
| 1 | Yes |
| 2 | Yes |
| 3 | No |
| 4 | Yes |
| 5 | No |
| 6 | Yes |
| 7 | Both automatic and manual |
| 8 | No |
| 9 | No |
| 10 | No |

**6. Task List**

**Task ID:** 1

**Task Title:** Complete end-to-end implementation of Smart Text Box

**Estimated start time:** 19.1.2017

**Estimated end time:** 1.3.2017

**Description:** Working functionality of the Smart Text Box end-to-end: UI, Server, DB.

**Sub-Tasks:**

* Complete Smart Text Box view page
* Implementation of the Smart Text Box controller
* Complete Smart Text Box logic component
* Ensure correct communication through all the layers

**Task ID:** 2

**Task Title:** Ensure right connections in DB through entity framework

**Estimated start time:** 19.1.2017

**Estimated end time:** 13.3.2017

**Description:** Make sure the connections created by entity framework match the data model

**Sub-Tasks:**

* Code all of the models according to the data model
* Test insertions of model instances to DB
* Manually check the connections created in DB

**Task ID:** 3

**Task Title:** Get the customer's agreement for the prototype

**Estimated start time:** 1.3.2017

**Estimated end time:** 30.3.2017

**Description:** Involve the customer in the development process and get her agreement for the current product

**Sub-Tasks:**

* Schedule a meeting with the customer
* Prepare the current solution for customer's presentation