**Domains Model**

**Login Model**

Login feature is required by client to provide security to users of the application. The users that are required to login are scorers, markers, judges and administrator ((if there is one). Each user will need a user name or user id and a password. Both must be validated and if it is valid checked against the database for a match. In case of a match users is successfully logged in. Otherwise an error message will be shown on the page accordingly.

The sequence diagram illustrates the overview of the process. At the moment, users can only login. There are no requirements to register users and or if users have forgotten their password.

**Purpose**

The purpose of the login domain is to restrict users to access any feature or data they are not eligible to access, change or see. The application could also hold personal data of users which must be preserved and secured.

**Marker WebApp Overall Model**

The overall model has been developed to reflect the follow requirements gathered from our client and throughout our assessment. It is expected:

* Login page for the marker
* Marker have limited access to the feature of the web application.

**The interface is expected to have:**

* A ‘correct’ button.
* An ‘incorrect’ button. (There **is no need to a wrong button as the amount of attempts for each team is irrelevant for the competition**). Replaced by undo button
* A ‘pass’ button.

**Unclear or must be revised:**

* Buttons must be time-delayed so the judge must hold the button down for at least 2 seconds before input is sent to the database. (**This requirement must be further discussed**).
* Buttons must display a confirmation when data is sent to the database. (Confirmation from marker is required. What kind of confirmation is needed from the database).
* Buttons must deactivate when data is sent.
* Buttons must reactivate when the user lifts their finger/cursor off the button.

**Who**

Marker is a member of the organizer team. A desired user for the Web Application. Usually a teacher or a person involved with AMA or MATHEX.

**What**

Markers are responsible for marking the answer of each team that they are assigned to. Each marker, usually, marks two teams.

Markers also give out the questions whenever it is appropriate.

**How**

Marker hands the question to a student. Then, student returns with the answer. If answer is correct, marker will note it down and compute the new total score. Marker hands to the student a new question to answer. This process continues until there is not more questions or the time is out. In case answer is wrong, student has the choice to try again or pass, if student pass a new question is given and marker notes it down as a pass.

**Diagrams**

**Sequence diagram –** shows the required actions from the marker all the way to the database. It illustrates the login process, the selection of teams to mark, the process of chosen a selected team to mark and the possible interactions from the marker to with the webpage.

**Activity diagram** – Shows only the possible actions the marker has.

**State diagram** – shows al states the webpage will hold as a result of each action performed by the marker.

**Initial prototype design** – retrieved from the initial paper prototype example to illustrate the suggested design for the marker’s interface. **This diagram is for support only.**

**Marker interaction with webApp walk through**

Markers are standard users of the application until they have logged in. In order to login, marker must access the login page. The login page can be reached indirectly by accessing the corresponding link. Or, directly through hyperlinks in the main page.

When logged-in. marker should be presented with a marker’s home page where they can select teams they want to mark. The number of teams to mark are usually 1 to 2, but it should be allowed more if required. Marker are allowed to deselect teams from the list.

Each team will have its page for marking. A team’s page shows the question number and the question in which a team is currently answering. The answer will be revealed when required. There will be thee action available in each page:

* A way to confirm a team has got a correct answer
* A way to pass a team to the next stage (question)
* A way that markers can return (undo) previous questions

Markers will transit to or from teams’ pages as long as the teams have been selected to be marked.

**Scorers’ WebApp Overall Model**

A “scorer” is to update the score of each team in the database. There are no requirements given by the client so far and the use of this solution eliminates the need for the team page as mentioned previously. The applicable requirements are as follow:

* Login page for the scorers.
* Scorers have limited access to the feature of the web application.
* Scorers must be able to update score of any team.

**Who**

Scorer is a member of the organizer team. A possible user for the Web Application. Usually a teacher or a person involved with AMA or MATHEX.

**What**

Scorers are responsible for updating the scores in current existing system. It is unclear how many teams a scorer look after. It seems to be flexible, however, it is expected that each scorer to look after at least 2 markers.

**How**

There are about 30 scorers in every competition. Each will look at the current score of a team in the paper sheet being filled by the markers and correspondingly updated the score of that team at the wall

**Why**

Markers do not have the time to get up and update the score due to the intensity of the competition. Therefore, markers count on the assistance of the scorers to constantly check and update the scores.

**Scorer interaction with WebApp walk through**

Markers are standard users of the application until they have logged in. In order to login, marker must access the login page. The login page can be reached indirectly by accessing the corresponding link. Or, directly through hyperlinks in the main page.

When logged-in, scorers will be presented with a scorer’s home page. A scorer will select one of the 120 teams competing and update its current score.

The score will be entered manually, by the typing of numbers. Scores entered can be lower or higher than a team’s current score. This allows corrections. Scorers can return to their home page to select a new team at any time by the use of a button.

**Backend Overall Model**

The system backend must run on a Linux operating system. The application server and the database system will work concurrently in the same machine to satisfy the proposed prototype product. The different implementation solutions are still in analysis and requires more research.

The backend will be developed using PHP. For this purpose, Apache application server will be used to serve the Webpages accordingly. The backend will be responsible to serve data from the database to the client as it is demanded.

The backend system should avoid doing any heavy computation or rendering to the client side such that most of the processing and rendering is done on client’s side. It should minimize the use of the server.

More information can be find in the backend domain folder.

**Database Overall Model**

The information that must be stored\update in the data base is listed below. This reflects the requirements that have gathered so far and it is believed to be the least expected from the database.

Profile:

* Login credentials and password.
* Name
* Privilege level : scorer\marker\judge\admin

List of teams in the competition.

* Name of the school
* Abbreviation
* Student’s name
* Any supervisor

Answer for each question for all teams.

* Correct
* Pass
* Total pass on each question
* Record timestamps

Total score for each team.

* Sum of the points gained by each correct answered question

Number of passes for each team.

* Sum of all passes for a team

**Database**

The database to be used is a relational database. Only data will be stored in the database, any other file such as pictures, sounds, videos will be stored in the application server.

A relational database design has been constructed to illustrate the minimum needs of the application. This design contains 8 tables each will store information about the competition, competition organizers, school, school participants, all question and answers, and all teams’ records.

**Which database?**

The current proposed databases are MySQL or PostgreeSQL. The system of choice must run in a linux operation system as per client’s requirement.

Although we have highlighted in the project proposal that we would be using MySQL database it may not be the case. The use of postgreeSQL is more likely as it is SQL compliant and ACID compliant which preserves the integrity of the data, in the case of our project concurrency must be handled carefully in the database.

**Database overall Model Walk Through**

The database design must satisfy a series of requirements as listed above. Those are extended in details as follow.

**Login\profile –**  Requires a user name and a password (password may want to be hashed for safety). Also, the name of the user and their privilege level ( such as judge, marker, scorer). No further details is required at the moment.

**List of teams in the competition** – each team represent a school. Each school may have one or more teams and must also be referred by their initials. Each team may have a supervisor or a supervisor may represent a school. A team contains 4 to 5 students. A team may or not have a name or id.

Note: the need of a supervisor is not clear but because it is a group of children they must have somebody to look after them.

**Question and answers –** All questions and the corresponding answers must be stored for each section.

**Team performance track** - all teams of one session will answer the same questions of that session. The result of their performance must be recorded. A team must either answered a question correctly or pass, the time must be recorded too. The total number of passes and correctly answered questions must be stored such that they can be used to compute the total score if necessary.