**Shared Models**

**Model Name**: UserLab

**Purpose:** Keep track of user progress across the various labs and record the quiz result/response. Each logged in user will have one entry in the database for a given lab. Guests will have a new entry every time since they receive a new usersession id.

**Attributes:**

|  |
| --- |
| **{  userlab: {  type: DataTypes.INTEGER,  primaryKey: true,  unique: true,  autoIncrement: true  },  usersessionid: {  type: DataTypes.BIGINT  },  labid: {  type: DataTypes.INTEGER  },  quizscore: { type: DataTypes.INTEGER, defaultValue: 0},  aboutcompletedtime: { type: DataTypes.DATE },  readingcompletedtime: { type: DataTypes.DATE },  gamecompletedtime: { type: DataTypes.DATE },  videocompletedtime: { type: DataTypes.DATE },  quizcompletedtime: { type: DataTypes.DATE },  quizresult: {type: DataTypes.TEXT} }** |

**Interaction with other Models:** Usersession. It is used along with ‘labid’ to identify uniqueness of the entry

**Interaction with Frontend:** As the user progresses through the sections, Userlab records the timestamp on each page. This allows us to track completion of the lab.

**Model Name**: User

**Purpose**: Keep track of user’s information. New google users get a new entry with name and image, while guests get a new entry without any firstname or image.

**Attributes**:

|  |
| --- |
| **{  userid: {  type: DataTypes.INTEGER,  unique: true,  primaryKey: true,  autoIncrement: true  },  firstname: { type: DataTypes.TEXT },  image: { type: DataTypes.TEXT },  },  { tableName: 'users' }  );** |

**Interaction with other models**: Users who log in using Google Auth have their designated session ID and userid. Guest users always get a new usersessionid and userid. When a guest logs into a Google account, their userid gets updated to match the Google user userid. This way we can map the guests progression into a signed-in user.

**Interaction with frontend**: Guest users see a “Sign In with Google” button in the navigation bar because they are not logged in. Logged-in users see their Google profile picture in the navigation bar and are able to click it to view a dropdown menu, in which they are able to click a “LogOut” button.

**Model Name**: Page

**Purpose**: To track time spent on each game page ( ie. including repair page).

**Attributes**:

|  |
| --- |
| **{  pageid: {  type: DataTypes.INTEGER,  unique: true,  primaryKey: true,  autoIncrement: true  },  usersessionid: {  type: DataTypes.BIGINT  },  pagename: { type: DataTypes.TEXT },  completiontime: { type: DataTypes.INTEGER },  labname: {type: DataTypes.TEXT}  },  { tableName: 'page' } );** |

**Interaction with other models**: Usersession. Used along with labid and pagename to identify uniqueness. If the entry exists, simply update the completiontime.

**Interaction with frontend**: As the user progresses through the game, a new page entry is created for the given lab and page (if one doesn’t exist under the user) to store completion time of the page. Lab1 does not use this model since it keeps track using the game model.

**Model Name**: Session

**Purpose**: Keeps track of a single user, user id, and the interactions the user has with the pages.

**Attributes**:

|  |
| --- |
| **{  usersessionid : {  type: DataTypes.BIGINT,  unique: true,  primaryKey: true,  autoIncrement: true  },  userid: {  type: DataTypes.INTEGER  }  },  { tableName: 'session' } );** |

**Interaction with other models**: Every user has a session, whether they are a guest or a signed-in user. A guest’s user id gets updated when they sign in as a Google user. Signed-in users don’t get a new session because we are using the session id given by Google Auth to identify them and their interactions.

**Interaction with frontend**: If a user has a session id that comes from Google Auth, then it is assumed to be a signed-in user that can see a Google profile picture in the navigation bar. Otherwise, it is a guest user that sees a “Sign In with Google” button in the navigation bar.

**Lab Specific Models**

**Model Name**: Game

**Purpose**: Keep track of score and timestamp of each playthrough

**Attributes**:

|  |
| --- |
| **{  gameid: {  type: DataTypes.INTEGER,  unique: true,  primaryKey: true,  autoIncrement: true  },  usersessionid: {  type: DataTypes.BIGINT  },  score: { type: DataTypes.INTEGER, defaultValue: 0 },  playthrough: { type: DataTypes.INTEGER },  timeplayed: { type: DataTypes.DATE, defaultValue: DataTypes.NOW }  },  { tableName: 'lab1\_game' } );** |

**Interaction with other models**: Interacts with Round. There are multiple rounds per game

**Interaction with frontend**: When a user starts a new game, a new game entry is created. Once the playthrough ends (timer runs out), and a new game gets started, then a new entry with the latest playthrough gets created. Ie Game id 1 Playthrough 0 -> Game id 2 Playthrough 1

**Model Name**: Choice

**Purpose**: **Attributes**:

|  |
| --- |
| **{  choiceid: {  type: DataTypes.INTEGER,  unique: true,  primaryKey: true,  autoIncrement: true  },  roundid: {  type: DataTypes.INTEGER  },  boxnumber: { type: DataTypes.INTEGER },  correct: { type: DataTypes.BOOLEAN }  },  { tableName: 'lab1\_choice' } );** |

**Interaction with other models**: In every game, users choose between four boxes. Each Choice is recorded as a new entry and is linked to the Round number. Before the user runs out of time, which is indicated by the timer, the user can make Choices. If their Choice is correct, it creates a new Round.

**Interaction with frontend**: If the user makes an incorrect Choice, the box will turn red. The score will be decreased and incorrect answer count will increment by 1. If the user makes a correct Choice, the timer will stop, the score will be increased, correct answer count will increment by 1, the next Round starts, and the Round count will increment by 1. Clicking on the “Hint Box” button will cause the correct Choice to appear with a glowing red background. If there is no correct Choice, all boxes will be marked with a red “X”.

**Model Name**: Round

**Purpose**: Keeps track of the rounds in between every *correct* Choice the user makes for as long as there is still time left.

**Attributes**:

|  |
| --- |
| **{  roundid: {  type: DataTypes.INTEGER,  unique: true,  primaryKey: true,  autoIncrement: true  },  gameid: {  type: DataTypes.INTEGER    },  hintused: { type: DataTypes.BOOLEAN },  soundoption: { type: DataTypes.BOOLEAN }  },  { tableName: 'lab1\_round' }  );** |

**Interaction with other models**: A Game id is used to link each Round to a Game, and every Game can keep track of who is playing using the user session id, the game score, the playthrough, and the timestamp.

**Interaction with frontend**: After every Round, or when a user makes a correct Choice, the page shows a five-second countdown and proceeds to the next Round. The page also increments the Round count by 1.

**Model Name**: Repair

**Purpose**: Tracks the repairs for the game

**Attributes**:

|  |
| --- |
| **{  repairid: {  type: DataTypes.INTEGER,  unique: true,  primaryKey: true,  autoIncrement: true  },  usersessionid: {  type: DataTypes.BIGINT  },  availablemessage: {  type: DataTypes.STRING  },  unavailablemessage: {  type: DataTypes.STRING  },  availablebackgroundcolor: {  type: DataTypes.STRING  },  unavailablebackgroundcolor: {  type: DataTypes.STRING  }  },  { tableName: 'lab1\_repair' } );** |

**Interaction with other models**: Interacts with the Usersession. This allows us to track repairs for each user.

**Interaction with frontend**: After a user hits the repair button, and submits the repair, a new repair entry is created with the fixes stored in the attributes above.

*\*Template - copy and paste\**

**Model Name**:

**Purpose**:

**Attributes**:

**Interaction with other models**:

**Interaction with frontend**: