# Data Transfer Processes

| Page | Page process | Database Transfers |
|---|---|---|
| log-in | 1. Request to server: verification of username and password. <br> 2. Response to client: <br> 3. If verified, <br>　　gain access to home page. <br> If not, <br>　　Send input value = "wrong username or password" | **Request "GET"** <br>　Username, password. <br><br> **Process** <br>　Verifies with database <br><br> **Response** <br>　If ok: sends home page. <br>　If not: sends error |
| home | 1. No transfer of data except request for user-interface files. <br> 2. This page is for navigation only (you can navigate to classes, setup, and back to home) <br> 3. There will be a button that can be pressed to start set-up on the page. <br> 4. Once set-up data transfer is completed. **Button** will change to "Resume Grading" which will send you to student-grader page. | **Request "GET"** <br>　Home files <br><br> **Process** <br>　Verifies request <br><br> **Response** <br>　Sends Home files to user |
| set-up | 1. User presses button "import student information". Will require log-in | **Requests "POST", "CREATE"** <br> 3 buttons: <br>　1. import student-info, |

| | | |
|---|---|---|
| | information (don't know how to handle this yet<br>2. On click, request starts python "beautiful soup in server to grab ('webscrape') student names, class, and subject into database.<br>3. Customize questions button: allows the user to change the default number and content of questions in the database.<br>4. Customize question range: Allows student to change default rating range anywhere between 1-10 | 2. save questions<br>3. save rating range;<br><br>**Process**<br>1. Grabs students through scraping and places them into a table.<br>2. Clears table and saves new questions to database with id.<br>3. Sets rating range in database to the user-input value.<br><br>**Response**<br>1. Sends list of students and subjects to display<br>2. Sends question list<br>3. Sends rating range. |
| classes | 1. Accessed from any document from nav bar.<br>2. Request for classes and corresponding Subjects into dropdown menu.<br>3. User can select individual class/subject or press resume rating or start rating, and it will start slideshow of classes or resume slideshow. | **Request "Get"**<br>1. Classes files<br>2. classes/subject<br><br>**Process**<br>2. Accesses database To get classes/subject<br><br>**Response**<br>1. Sends classes files<br>2. Sends classes/subject |
| classes-students | 1. Requests students and any previous student for that class. | **Request "Get"**<br>1. Student_names<br>2. Grade and subject<br>3. Last five scores and |

| | | |
|---|---|---|
| | 2. Each student should have the last five participation scores displayed with corresponding dates in the same row.<br>3. Can access older scores by clicking "older" which will grab the five previous scores.<br>4. Each score should be clickable and access a printable/pdf document of that students' scores for each question.<br>5. Clicking on students will send user to rating page to create new score and portfolio for that date.<br>6. You can also press a button to start a grading slideshow.<br>7. Refer to edge-case | dates (can access more scores and dates from page).<br>**4. Refer to edge case**<br><br>**Process**<br>1. Grab all of this data from table<br><br>**Respond**<br>1. Send student_names<br>2. Send grade and subject<br>3. Send last 5 scores and date |
| classes-student-portfolio | 1. User accesses a portfolio of each individual score which contains student_name, grade/subject, questions and corresponding scores, and the average score. | **Request "GET"**<br>1. Student_names<br>2. grade/subject<br>3. questions (default or questions from set-up) and recorded scores.<br>4. Average Score<br>**5. Refer to edge case** |
| classes-student-rating | 1. User accesses individual student and the list of questions. | **Request "GET" "POST"**<br>1. Get student_name, questions, and rating_range. |

| | | |
|---|---|---|
| | 2. User input rating between rating range, and each score gets saved with an id with a foreign key corresponding to the corresponding question.<br>3. Average score is calculated and listed with a date stamp | 2. rating.on("On click") will post a score for that question at the end of rating that student.<br>3. Post average score with date stamp for that student<br>4. **Refer to edge case**.<br><br>**Process**<br>1. Grab student_name, questions, and rating_range from the database.<br>2. Input scores for each question for that particular student into the database.<br>3. Post average score with date-stamp. #2 and #3 can be posted together.<br><br>**Response**<br>1. send student_name, questions, and rating_range on document load.<br>2. After input into database, send new student with same questions and rating_range. |
| **Edge Case** | 1. I do not which page this will be accessed from right now (probably on the classes page), but when a button is pushed the server will check to see if all score values for a particular are not | **Request**<br>1. Post grades and date stamp into fast-direct if scores for a class are not null.<br><br>**Server**<br>1. Uses python to send |

| | | |
|---|---|---|
| | null. Then if ok: it will access fast-direct and input a grade for current date stamp in for habits of learning performance grade. | data to fast-direct.<br><br>**Response**<br>1. Completion message. |
| | | |