Internship for the lecture databases
BAI, Komedia, ISE CSCE/CE, Mathematics/Technomathematics and Teaching Department Computer Science

Alfred Sliwa, M.Sc.
Ioannis Karatassis, M.Sc.
Iryna Rushchak
Dakai Men
Huthayfa Ghunaim

Winter semester 2019/2020

Date	
Team (Account)	
Password	

Up-to-date information, contact persons, materials and uploads at: http://www.is.inf.uni-due.de/courses/db_ws19/

v1.0

Table

0 Block 0: Introduction to Linux

No levy

Task 1: Set up the work environment (0 points) Prepare your work environment first so that you can work with it comfortably in the next few weeks. Most settings can be made directly from the menu in the upper right corner (a balloon with your login name) and then system settings.

Task 2: Linux environment (0 points) To get used to the work environment, some simple tasks specified by the internship manager are processed.

At the end of the task, you should be able to navigate directories, view directories, and view files using the Linux shell. You should know where to access a text editor, file browser or web browser and how to create a PDF from a text file.

1 Block 1: Modeling the database for the crowdfunding platform "ProjectFunder" (12 points)

Delivery until 01.12.2019, 23:59

Task 3: Database modelling (6 points) In this semester, "ProjectFunder" - a web application for the financing of projects - is to be developed. The platform is very versatile and is intended to support various functions, including:

- Users can create, update, and delete projects.
- Users can support projects with a donation.
- Users can comment on projects .

In this task, the first step is to model the underlying database with an entity relationship diagram (ER diagram).

Appendix A.1 on page 11 shows how the web application to be created works. The ER diagram to be modeled should only be based on the description of the mini-world. It is important to note that not every requirement can be implemented in the ER diagram.

It is strongly recommended to read and understand the full specification (see Appendix A.2 on page 12) before designing, in addition to the description of the mini-world.

- Uses the Dia program to create the ER diagram (http://dia-installer.de/).
- If you want to use more attributes than specified during modeling, this is of course allowed in meaningful places.
- Specifies cardinalities in the Chen notation (1:1, 1:N, N:M).

Levy:

- ER-Diagramm as PDF-File

Task 4: Database creation (6 points) Creates CREATE TABLE and INSERT INTOStatements required for the database designed in task 3, which create and populate the database tables in DB2. Only the data that has been explicitly specified in the description of the mini-world should be inserted. Apply the techniques from the Database Lecture to first transfer your ER diagram from Task 3 to a database schema.

The levy should include:

- CREATE TABLE statements incl. meaningful integrity conditions for all attributes (4.5 points)
- INSERT INTO Statements (0.5 points)

- A trigger that checks when inserting a donation to see if the funding limit for the project has been reached. If so, change the status of the project to Closed. (1 point)

Levy:

- CREATE TABLE-Statements, INSERT INTO-Statements, und CREATE Triggerstatement as a text file (.txt)

2 Block 2: SQL with DB2 (15 points)

Delivery until 22.12.2019, 23:59

For the following tasks, a description of the database used in Appendix B is available on page 23.

Note: Block 2 is independent of blocks 1 and 3. The latter build on each other.

Task 5: Catalog (1 point) In this internship we will address the DB2 from the command line. There are also graphical, Java-based tools, but we do not use them here.

The database we will be working with is located on a remote computer and must therefore be imported into your computer. Briefly familiarize yourself with the different modes described in the documents. Then catalogs the instance dbmaster on the machine bijou.is.inf.uni-due.de with server port 50005.

Catalogs the IMDB database locally under an alias. Then, let's view the Node Directory and Database Directory.

Submission:

- Entries for instance (node) and database (database) as text file (.txt)

Task 6: Views (2 pointse)

(a) Creates a view[1]dbpXXX.avgRatedMovies, which should include movies that have been rated better than average.

? The view should have 4 columns: MOVIEID, TITLE, YEAR, and RANK.

(b) Creates a view dbpXXX.waltDisneyMovies that contains all the movies produced by the producer "Disney, Walt". In addition, the genres of the movies should be displayed. If a movie has more than one genre, you want to display one line per genre.

? The view should contain 3 columns: MOVIEID, TITLE, and GENRE.

Submission:

- SQL commands as a text file (.txt)

Task 7: Use of views (2 points)

(a) Uses the dbpXXX.avgRatedMovies view from Task 6 to output the highest rating films for each year. (There may well be several films per year with the highest rating.) *?* The output should contain 3 columns: YEAR, TITLE, and RANK.

(b) Uses the dbpXXX.waltDisneyMovies view from Task 6 to output movies that have no more than 2 genres.

[1] "XXX" stands for your group number, e.g. dbp001.

? The output should contain 2 columns: TITLE, AMOUNT.

Submission:

- SQL commands (SELECT Statements) as a text file (.txt)

Task 8: SQL Requests (7 points) Formulates the following requests in SQL:

(a) How many actors does a horror movie average?

? Output should include 1 column: AMOUNT

(b) Determines the average profit (large) per film for each director. Sorts the output descendd by profit.

? The output should contain 2 columns: NAME and MEAN_GROSS

(c) Which movie has the most actors? (There may be multiple movies.) In addition to the title of the film, also give out the number of actors.

? The output should include 2 columns: TITLE, AMOUNT

(d) Which directors have played in one of their own films at least once? We assume that a director and an actor (actor) are the same person if they have the same name.

? The output should include 1 column: NAME (director)

(e) Which writers have only participated in films published between 1920 and 1929.

? Output should include 1 column: NAME (Writer)

(f) For each film, determine the actors and directors involved (role). One line should be output per film and contributor. His role (actor/director) will also be issued. The output should be sorted in descending order by the movie title .

? The output should include 4 columns: TITLE (film), YEAR (film), NAME (contributor) and ROLE (actor/director)

(g) For each genre, calculate the relative market share as a percentage relative to the films that own the genre. For example, if 10 of the 80 films are horror films, the market share for horror films is 12.5%. Films that do not have a genre should not be taken into account.

? The output should have 3 columns: GENRE, MOVIES_PER_GENRE, MARKET_SHARE.

Submission:

- SQL commands as a text file (.txt)

Task 9: Recursive request (1 point) The game "Six Degrees of Kevin Bacon" goes like this (description of http://www.cs.virginia.edu/oracle):

The object of the game is to start with any actor or actress who has been in a movie and connect them to Kevin Bacon in the smallest number of links possible. Two people are linked if they've been in a movie together. We do not consider links through television shows, made-for-tv movies, writers, producers, directors, etc. For example, you might wonder how Alfred Hitchcock can be connected to Kevin Bacon. One answer is that:

Alfred Hitchcock was in Show Business at War (1943) with Orson Welles, and Orson Welles was in A Safe Place (1971) with Jack Nicholson, and Jack Nicholson was in A Few Good Men (1992) with Kevin Bacon!

Then we can count how many links were necessary and assign the actor or actress a Bacon number.

Finds all actors and actresses with a maximum bacon number of two. As a thought exercise, a recursive SQL statement writes that the actors and actresses find any bacon numbers.

Tip: In SELECT statements, you can return the result of calculations. Example: SELECT 1+3 FROM title;.

Submission:

- SQL commands as a text file (.txt)

10:Trigger(2Points) Createsthetable dbpXXX.frequentRatedMovies (replacesXXXreeuregroup number):

```
CREATE TABLE dbpXXX.frequentRatedMovies (
  movieID INTEGER NOT NULL, title VARCHAR(400),
  year VARCHAR(100),
  PRIMARY KEY (movieID)
);
```

Writetwotriggermitdemnames dbpXXX.trigger1 and dbpXXX.trigger2 that do the following:

- trigger1: As soon as a movie receives its 100th rating, it should be included in the table dbpXXX.frequentRatedMovies with the specified attributes.
- trigger2: As soon as a rating for a movie is deleted and that movie has less than 100 ratings, the movie should be removed from the dbpXXX.frequentRatedMovies table.

Levy:

- SQL commands as a text file (.txt)

3Block 3: Implementation of the crowdfunding platform "ProjectFunder" (18 points)

Completion by acceptance date (27.01.2020 - 31.01.2020)

11: Correction of the draft (0 points) Once you have received the fixes to Block 1, revise your draft and may fix any errors and problems in your database. Then set the latter up and fill it with sample data. However, we recommend that you use the sample solution to set up the database. This will be displayed at the end of the 1st block on the
event website.

Levy:

- None.

12: Implementation (18 points) The goal of the 3rd block is to implement a web application that implements the described requirements (see description A.2 in Appendix A). The structure of the data is already known from the description of the mini-world (see Annex A.1). The database scheme created in block 1 is the database.

The following requirements must be observed:

- Using DB2 as DBMS. Other DMS (such as MySQL or MariaDB) are not allowed and will not exist when used. We recommend using DB2 on our computers. The registration details will be received at the beginning of the internship.
- The choice of programming language is limited to Java and Python. You are required to use a web application that reads / writes and displays the data from db2. Sample projects in Java and Python are provided, which can be used as the basis for the upcoming implementation.

  It is important to note that any frameworks/libraries are not allowed with respect to SQL statements. The standard functions of the programming language (e.g. statements in Java) should be used.
- Correct use of transactions.
- Close DB resources in meaningful places.
- Preventing SQL injection.
- There should be enough sample data in each created table to test the site and for acceptance.
- All interactions should be performed by the currently logged on user. A login function is not necessary. The user can be inserted hard-coded into the program code.

13: Bonus task (1 point) Appendix A.2.8 contains a bonus task. You can earn a bonus point by fully implementing the requirements. Subpoints are not awarded.

Levy:

- No. The web application is presented and checked at a separate acceptance date together with the organizers.

Notes on the acceptance of block 3:At the acceptance date there are two

Possibilities:

(a) You can also demonstrate your solution on a computer in the computer pool. To do this, the final version of your project should be in the workspace of your account.

(b) You can have your solution removed by demonstrated on your own laptop, etc. To do this, bring the device to the appointment and please think of power supply and/or enough battery charge. Please test the demonstration beforehand. In case of need, there should be a copy of your final project in the workspace of your account.

We recommend variant (a) as we can still intervene if something doesn't work. In both cases, we will look at both the functionality of the program and the source code! Subpoints are awarded for each feature that is implemented correctly. Therefore, it is worthwhile to register for acceptance in any case.

Important: In order to successfully pass the acceptance, the following conditions must be met:

- Both the web application and the database must be able to be started. Any changes can not be made during acceptance !
- The main page of the web application must be accessible.
- All data to display (except image files) must exist in the database. The web application must be able to read, write and delete data.
- The project must have been created by your group. Attempts at deception (e.g. hard-coded data instead of reading from the database) or plagiarism lead to exclusion from the internship and are
  Audit Committee.

If a condition does not apply, Block 3 is scored 0 points. Therefore, test your project extensively beforehand and refrain from (untested) last-second changes!

Annex

ASpecification of the website

A.1Description of the mini-world

- Users have a name, a unique email address, and an optional profile description.
- All users have an account that manages their balance on the platform. The account is uniquely identifiable by the user and has a secret number in addition to a credit.
- A user can create multiple projects. Each project is created by exactly one user.
- A project has a unique identifier (such as a sequential number), a title, a description (optional), and a funding limit.

- Users can support projects through donations. Of course, we also save the donated amount (donation amount). In order to protect the privacy of donors, it is also stipulated whether the donation should be made anonymously or publicly. A donation can only be made if there is enough money in the user's account.
- The credits, the funding limit and the amount of donations are floating-point numbers with two decimal places each. Basically, the currency Euro is used on the platform.
- A project has a Status attribute that can only take the values "open" and "closed". When a project is created, the status assumes the value "open" by default. Closed, on the other hand, means that the funding limit has been reached and no further donations can be made.
- Projects can be in a predecessor-successor relationship. A project can have multiple successors but a maximum of one predecessor; we clearly mark both roles!

  For example, the project named "Ubuntu Touch" has "Ubuntu Touch Light" and "Ubuntu Touch Pro" as its successor.
- Categories each have an ID, a name, and an icon. We offer the following categories: Health & Wellness, Art & Creative Works, Education and Tech & Innovation. Each project is uniquely associated with a category.
- Users can create comments for each project. A comment consists of an ID, a text message, and the creation date. In addition, the user can specify the visibility of the comment (private or public). Hide private comments in contrast to

  to public —the actual user name and display the name "Anonymous" instead.

A.2Description of the website to be created

A web application called "ProjectFunder" is to be created. This is based on the database from block 1. Our web application should include several pages to create or support projects, among other things. All pages that need to be implemented for block 3 are described in sections A.2.1 to A.2.7.

Each page to be created has its own ID to use to access it. In addition, there is a graphical representation (mockup) for each page. References to other pages are indicated by blue arrows in the mockups. In any case, a link or button is required, which forwards the user to the corresponding page. It should be noted that the predetermined design in the mockups does not have to be recreated 1 to 1. Only the functionality of your web application is evaluated. Sub-points are awarded for each required functionality that has been successfully implemented.

For a better understanding, the functionality in the name is specified for each page to be created. We usethe prefixes view, new and edit:

- view is mainly used to display data.
- new is used to create data.
- edit is used to update existing data.

For example, the view_project page is used to display the details of a pro-

---

during new_project allows you to create projects. We can update an existing project via edit_project.

A.2.1 Main page (ID: view_main)

- You want to get started from the main page with the ID view_main (see Figure 1). On the main page, all open and closed projects should be displayed separately. For each listed project, you want to display the icon, title, user name of the creator, and the current donation amount (Current). The icon is the icon of the corresponding category. In addition, the main page contains two buttons: "My Profile" and "Create Project".
- If you click on the button "My Profile", you will be redirected to your own profile (page view_profile A.2.6).
- If you click on the "Create Project" button, you will be redirected to the page new_project (A.2.2).
- If you click on the title of a project, the corresponding detail page (view_project A.2.3) should be accessed.

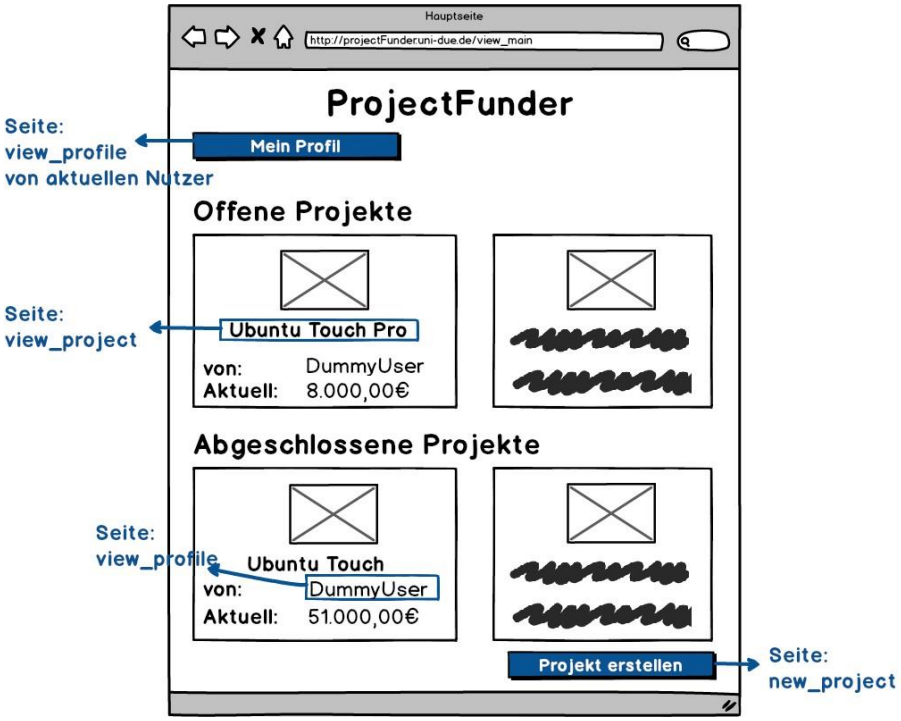

Figure 1: Overview of all projects (ID: view_main)

- If you click on the user name (creator of a project), the corresponding profile page (view_profile A.2.6) should be called up.

A.2.2Create Project (ID: new_project)

- You want to be able to create a new project from the new_project page (see Figure 2).
  - The title and funding limit should be specified by a text box.
  - For our four categories "Health & Wellness","Art & Creative Works", "Education" and "Tech & Innovation" we use RadioButtons.
  - For the predecessor project, a project can be selected from the list of projects that have already been created by (and only by) the current user. The selection is made via radio buttons. There is also the "No Predecessor" option (default selection) if the user does not want to link the project to be created.

    For example, if the user has previously created the Ubuntu Touch and Ubuntu Touch Pro projects, these two projects will be made available to select the predecessor project of the newly created Ubuntu Touch Light project.

Figure 2: Project Creation Page (ID: new_project)

  - The description should be specified via a text area.

- If the user clicks on the "Create" button, the project will be entered into the database. A sequential number (ID) can be used for the ID. If you encounter errors when creating a project, you want to receive an error message and not create the project. The following errors must be caught:
  - If the length of the title you entered exceeds 30 or is empty.
  - If no numeric value or value less than 100 has been entered for the funding limit.
  - If no category is selected.
  - If no option for predecessors has been selected. - General error with the database.

A.2.3Detail page of a project (ID: view_project)

Figure 3 illustrates the detail page of a project. This consists of 4 sections:

Information

- Based on the identification of a project, the following information should be read from the database and then displayed: icon of the category, title, user name of the project creator, description (if any), funding limit, the current donation amount and status.
- If the user clicks on the user name of the creator, the profile page of the creator (A.2.6) should be called.
- It should also include a link to the predecessor project (A.2.3) if a predecessor relationship exists. If this is not the case, then a simple text with the content "No predecessor present" should appear.

Action bar The user can perform various actions on the detail page of a project. The "Donate", "Edit" and "Delete" actions in the form of buttons/links are supported.

- If the user clicks on the button/link "Donate", you will be redirected to the page new_project_fund (A.2.5) with the corresponding parameter (identification).
- If the user clicks on the button/link "Edit Project", you will be redirected to the page edit_project (A.2.4) with the corresponding parameter (identification).
- If the user clicks on the "Delete Project" button, the following actions should be performed, but only if the current user is the creator. Otherwise, an error message isdisplayed on.
  - You want to delete the comments associated with the project.

- The donations associated with the project should be deleted.
- Users who supported this project will receive their donation back. This means that the donations will be credited to the accounts of the respective users.
- The project should be deleted from the database.
- The current user is then redirected to the main page (A.2.1).

If an error occurs during the operation, you should cancel the entire operation and display an error message.

List of donors

- In the middle section of the Web page, the donors associated with the project should be loaded and displayed from the database. In addition to the names of the donors, the respective donation amount should be displayed.

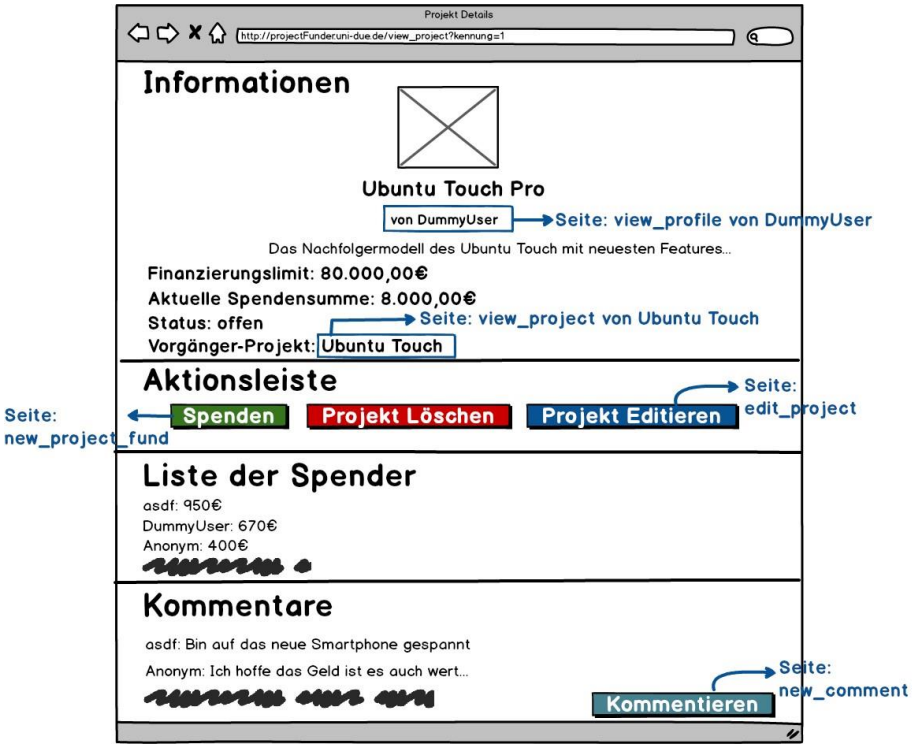

Figure 3: Detail page of a project (ID: view_project)

- The list of donors should be sorted in descending order by the donation amount (largest first).
- If a user has donated anonymously, the name "Anonymous" should appear in this list and not his actual user name.

Comments

- At the bottom of the Web page, you want to load and display the comments associated with the project from the database. Next to the comment, you want to see the author's user name (or the name "Anonymous" if the comment is not public).
- The comments should be sorted descending (newer first) based on the creation date.
- If you click on the button/link "Comment", you will be redirected to the page new_comment (A.2.7).

A.2.4 Edit project (ID: edit_project)

The page for editing a project has the same structure as the page for creating a project (see A.2.2). First, the data on the

Figure 4: Editing a Project (ID: edit_project)

selected project from the database and then displayed (see Figure 4). The following conditions must be observed as soon as the "Update" button is clicked:

- A project can only be edited by the creator and only if the status is "open".
- The title must be between 1 and 30 characters long.

- The funding limit must be a number. In addition, you can only increase the limit: the new limit must be greater (or equal) than the previous funding limit.
- A category must be selected.
- An option must be selected for the predecessor project (or the No Predecessor option).
- Generally error with the database.

If you click on "Update" and the requirements are met, the database should be updated and the user will be redirected to the detail page of the project (A.2.3). Otherwise, an error message is displayed; the details will not be updated.

A.2.5 Project Donation (ID: new_project_fund)

On this page (see Figure 5), the user can support a selected project financially by making a donation. First, the title of the project is displayed. The user can enter a numeric amount from a text box to support the project. In addition, he can use a checkbox to indicate whether the donation should be made anonymously. The user confirms the input via the "Donate" button. The following must be observed:

- First, it checks whether the status of the project is also open. Only open projects are supported.
- The value entered for the donation must be a numeric value and greater than 0.
- Check that the donation amount does not exceed the user's balance.
- You can only make a donation for a project if you have not supported it before. Multiple donations related to a project is not possible.

If one of the conditions mentioned is not met, an error message should appear and the operation should be canceled.

If all requirements are met, the following actions should be taken:

- The donor will be entered into the database with the corresponding donation amount.
- If the user donates anonymously, this should also be noted in the database.
- The credit in the user's account will be reduced by the corresponding donation amount.
- In the end, it is important to ensure that the status is updated from "open" to "closed" if the funding limit has been reached by the current donation.
- The user should then be redirected to the project's details page (A.2.3).

A.2.6User profile (ID: view_profile)

The user profile is divided into three sections (see Figure 6):

Header area In the header area, you want the information of the corresponding user to be read and displayed from the database. The e-mail address should be used to determine the correct data. View the following information: e-mail address, user name, and the number of projects created and supported (whether anonymous or public).

Figure 5: Donation page (ID: new_project_fund)

Created Projects This section displays all projects created by the selected user. For each project, you want to display the icon of the category, title, the current donation amount and the status ("open" or "closed").

If the title of a project is clicked, you will be redirected to the corresponding detail page (A.2.3).

Supported Projects This section displays all projects that have been publicly supported by the selected user. Anonymous donations should therefore not be taken into account. The icon of the category, title, funding limit, status and, last but not least, the amount donated by the user are displayed.

If the title of a project is clicked, you will be redirected to the corresponding detail page (A.2.3).

Add A.2.7Comments (ID: new_comment)

On this page (see Figure 7), the user can create a comment about the selected project. The title of the project is displayed as information. The comment is entered via a text area. A checkbox can also be used to indicate whether the comment is stored anonymously.

As soon as the user clicks on the "Add comment" button, the corresponding comment should be inserted into the database. The following must be observed:

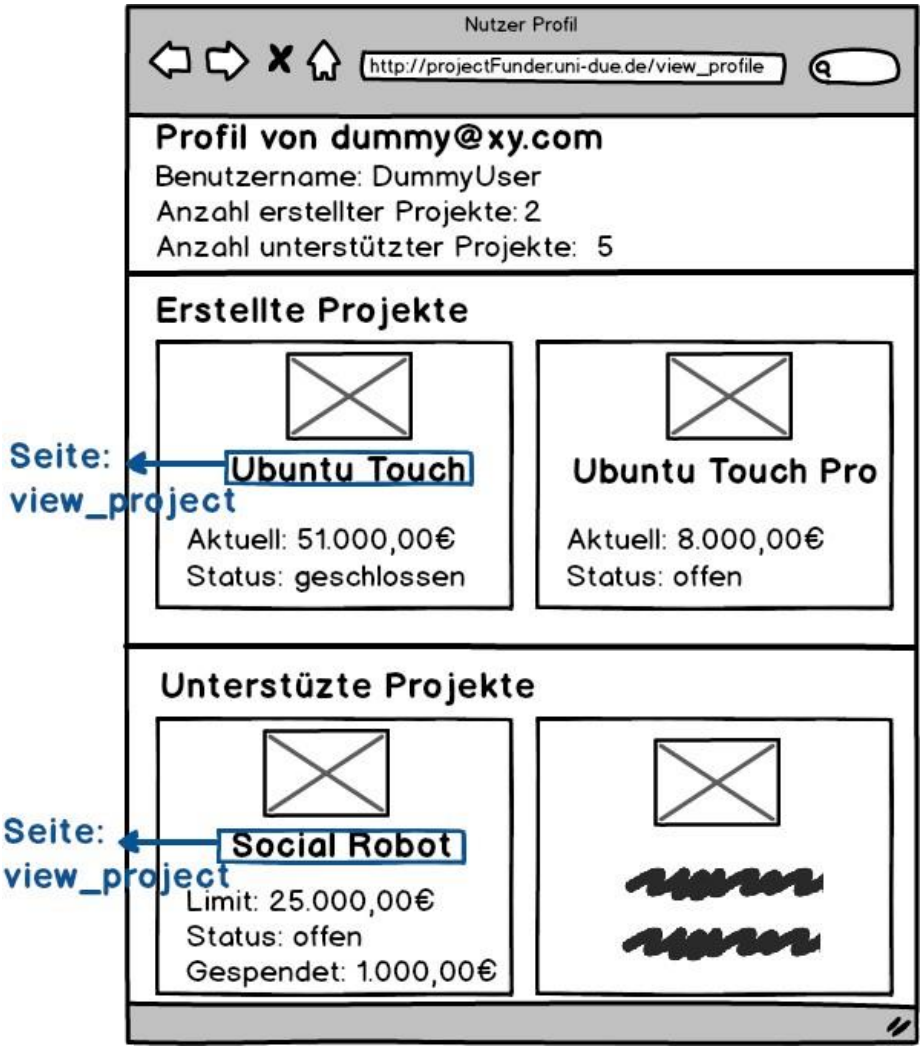

Figure 6: The user's profile (ID: view_profile)

Figure 7: Adding a Comment to a Project (ID: new_comment)

- The comment cannot be empty. In this case, you want to issue an error message and cancel the operation.
- If a comment is created successfully, you should be redirected to the detail page of the project (A.2.3).

A.2.8Bonus task (1 point)

The following task is optional. Another point can be collected by editing.

Search (ID: search)

Of course, a search function must not be missing in our web application. On the new page, the user has the ability to search for projects by title (see Figure 8).

- If the user clicks on the "Search" button, projects whose title starts with the specified search query will be found. Case and lowercase can be ignored. The search operation must be done at the database level!
- For each project found, you want to see the category icon, title, creator's user name, current donation total, and status ("open" or "closed").

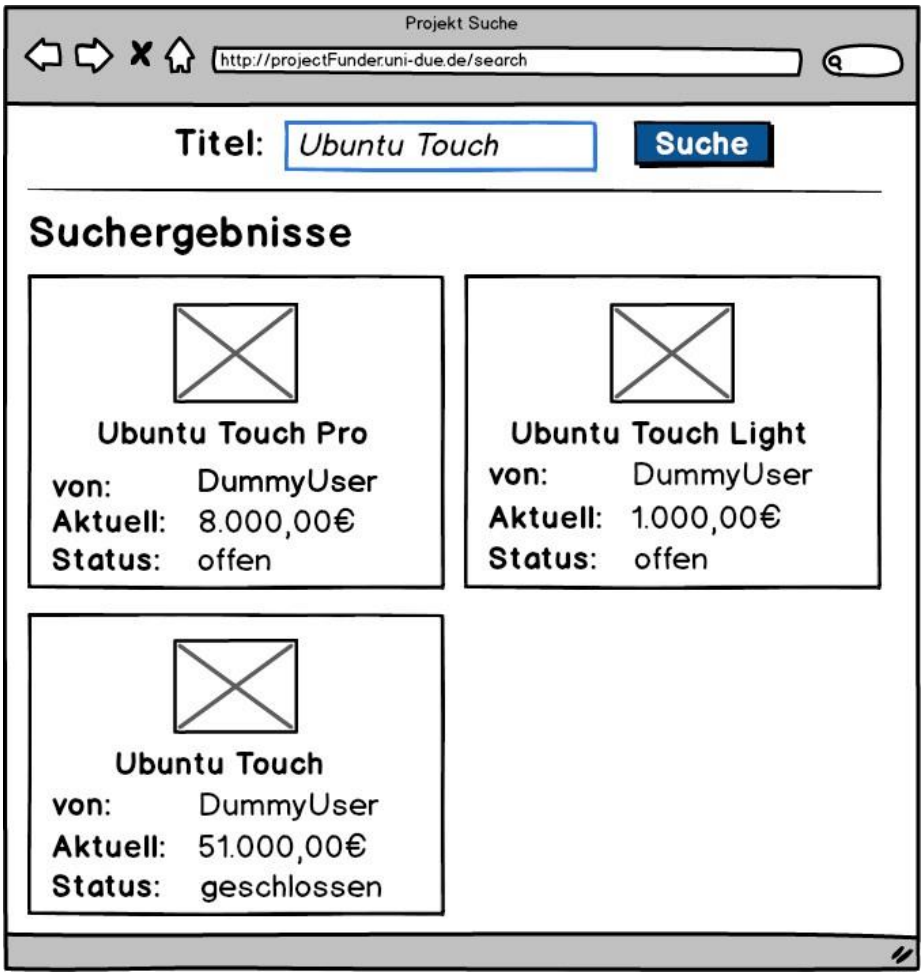

Figure 8: Bonus Task: Search

- If the search query is empty or does not produce any results, you should display a message instead.

For example, if the user searches for "Ubuntu Touch", the following Projects are returned: "Ubuntu Touch", "Ubuntu Touch Light" and "Ubuntu Touch Pro".

You do not want to see projects whose title does not start with "Ubuntu Touch", such as "Ubuntu Tablet" or "Windows Touch".

BDescription of the IMDB database

A sample database of movie data is specified for the SQL tasks. You should have already cataloged the IMDB database under a local alias in the first week.

The relevant section of the schema of the database is described below. You can also ask for the structure of the tables directly from the database. The data in the tables do not partly give a complete picture of the world, but should be sufficient for our purposes.

B.1UML diagram of the IMDB database

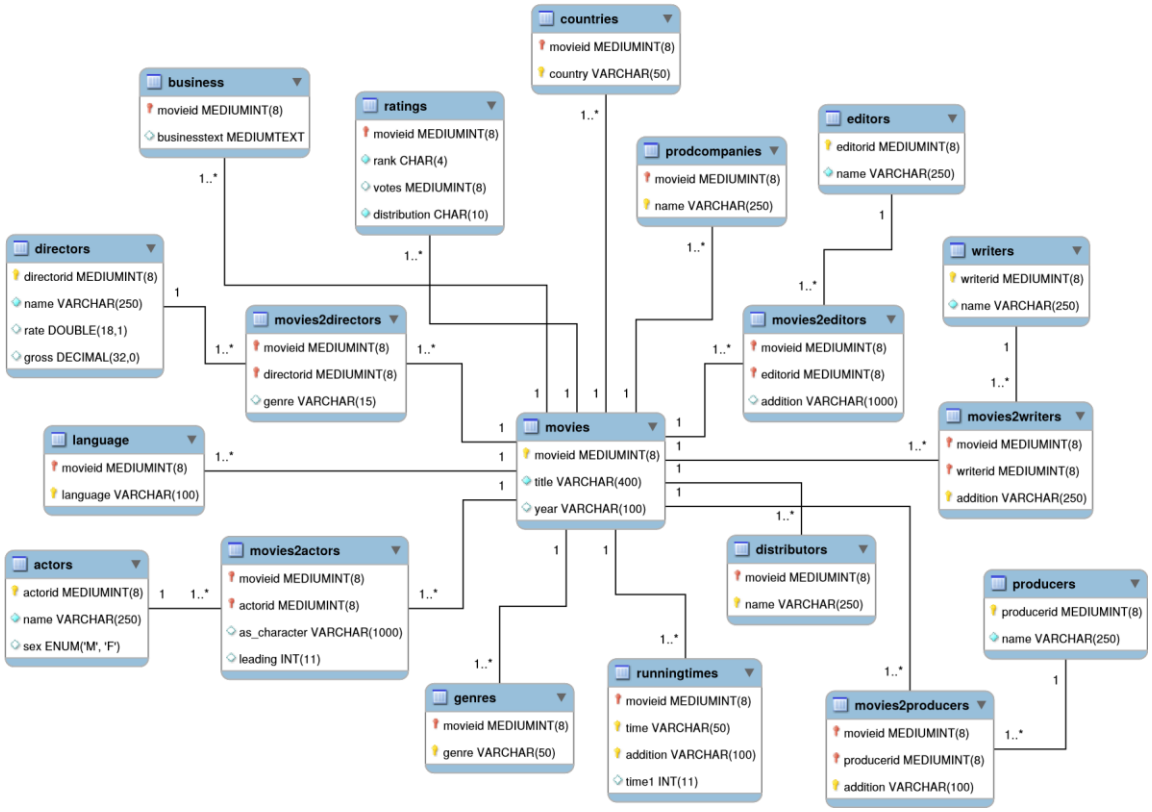

Figure 9: UML Diagram of the IMDB Database