1222·2022 800 ANNI
UNIVERSITÀ DEGLI STUDI DI PADOVA
DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

Web Applications A.Y. 2023-2024
Homework 1 – Server-side Design and Development

Master Degree in Computer Engineering
Master Degree in Cybersecurity
Master Degree in ICT for Internet and Multimedia

Deadline: 29 April, 2024

Group Acronym	ACME	
Last Name	**First Name**	**Badge Number**
Bortolatto	Stefano	2103656
Kumar	Vinayak	2106528
Sadat	Farhad	2080650
Omarbekova	Ramilya	2080400

1 Objectives

The project BookRec is a website aimed for book enthusiasts and people who like reading books of all kinds.
In this website, accessible both by logging in with an account or anonymously, people will be able to search for specific books and their main information (like author, genres and publication date) or simply obtain a list of books reccomended by other users of the website based upon ratings, genre and number of readers.

2 Main Functionalities

The main functionalities rotate around users and their ratings of books they read and searching for new book to read. Keeping this in mind we want a database to store information both of books (with a main focus on "book", "author" and "publisher") and users (with an eye on the rating mechanic).
For the presentation and the UI the idea is to keep things simple: in the main page there will be a search bar on top, with different rows of books below it; each rows is a different category of suggested book (top 5 rated, top 5 read, ...) and alternatively you can search in the search bar either user, author or title of book.

3 Data Logic Layer

3.1 Entity-Relationship Schema

The ER schema for the database is quite simple:

- **Users**: this entity represent the users registered in the website, they are described by a *name* (the application is not formal, so just the username should be enough) and by the login credentials *email* and *password*.
- **Books**: this entity contains all the main information about a book, like its *title*, *plot*, *release* date and *ISBN code*, in addition to its *cover* image for aesthetic purpose in the website.
- **Authors**: here we have the various authors of the books present in the website, with information about them such as *pen name* and *biography*.
- **Publishers**: simple entity to store efficiently the publisher of books, with only the attribute *name*.
- **Genres**: simple entity to store the genres of the books, described only by its *name*.

For what concerns the relationships, we have:

- **wrote**: this relationship connect one or more authors to one or more books; a book can have more than one author and an author, likely, wrote more than just one book.
- **hasRead**: this relationship connects users to books which they read and which they can rate on a scale from 1 to 5; it's a N:N relationship, since a user can read more than one book and books can be read by many users.
- **bookGenre**: this relationship connects books to their genres and it's N:N (a book can have more than one genre and vice versa)
- **publish**: this relationship connects books to their publishers; a book with a ISBN code can only be published by one publisher, so this is a 1:N relationship with books on the 1 side, while a publisher can have multiple books in its catalog.

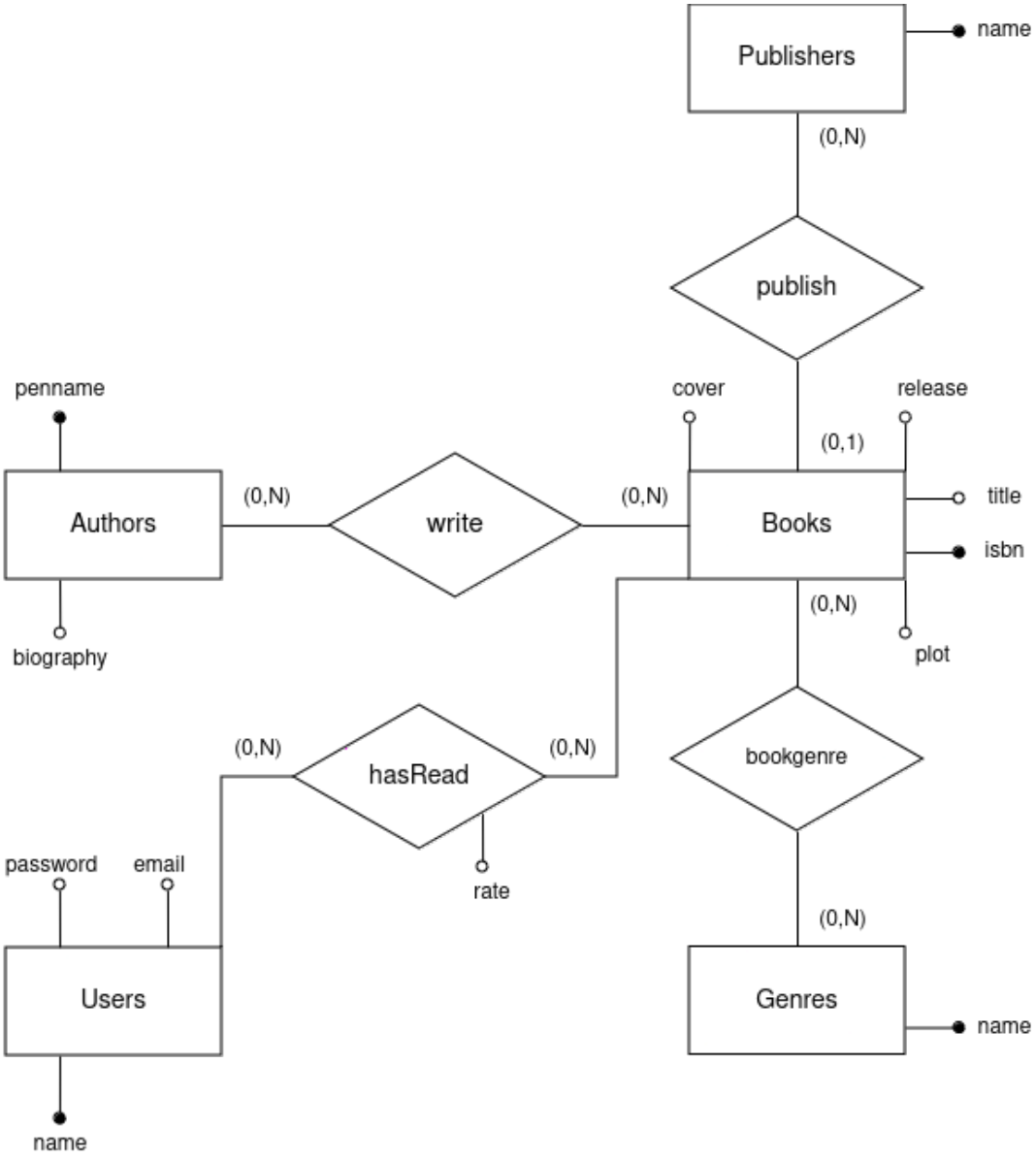


Figure 1: ER schema of the database

3.2 Other Information

Some of the relationships mentioned should have mandatory participation (a book must have genres it belongs to, and author should wrote a book present in the website, ...), but since most of them requires an extra table to be implemented (since they are N:N it cannot be done with attributes used as reference key) it would be hard to implement this constraint without breaking the possibility of adding entries, hence the decision to keep them with optional participation.
The decision of not using numerical ID as primary key for the entities comes from the idea of having a simpler code in the implementation part.

4 Presentation Logic Layer

Below we define the pages we are going to develop in the BookRec application. All the pages have been developed using html or jsp.

- **Starting page**: the very first page that the user sees. On this page the user can search for books, genres, or other users;
- **Sign-up page**: this page allows the user to register in BookRec.
- **Login page**: this page allows the user to log in if the user is already registered in BookRec.
- **Homepage**: the page seen by the user once they are logged in.
- **Search results**: shows a list of findings accorfing to the search.

Each page, except for the login and the starting page include and header with the logged in user information which allows them to logout.

4.1 Starting page

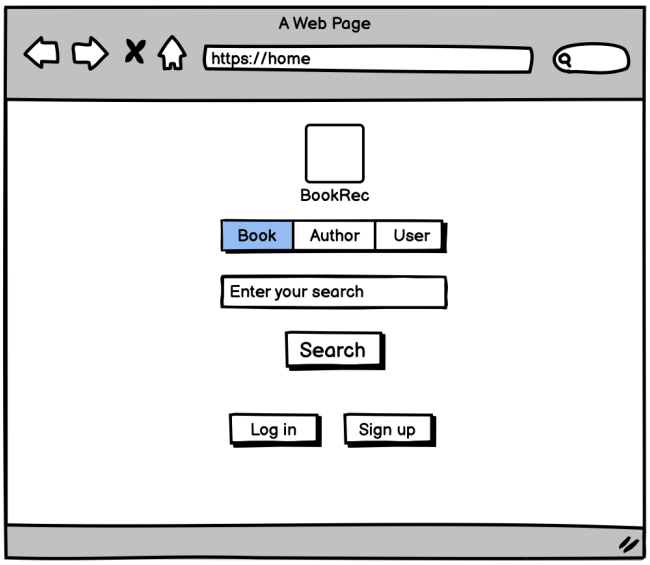


Figure 2: Starting page

Starting page is the very first page that the user sees. On this page the user can search for books, genres, or other users

4.2 Sign up page (Interface mockup)

In Figure 3 we described a simple sign up page. On the Sign up page, the user can provide their credentials to use to access their account. The information a user must provide is the email, the password, and the username. The password to be provided must contain at least 8 characters, at least one number, and at least one upper case letter. After clicking on Sign up the user's credentials are added their credentials to the database, and they are redirected to the homepage. If the credentials do not comply with those required, an error message is returned. If a user already has an account, they can proceed to the login page.

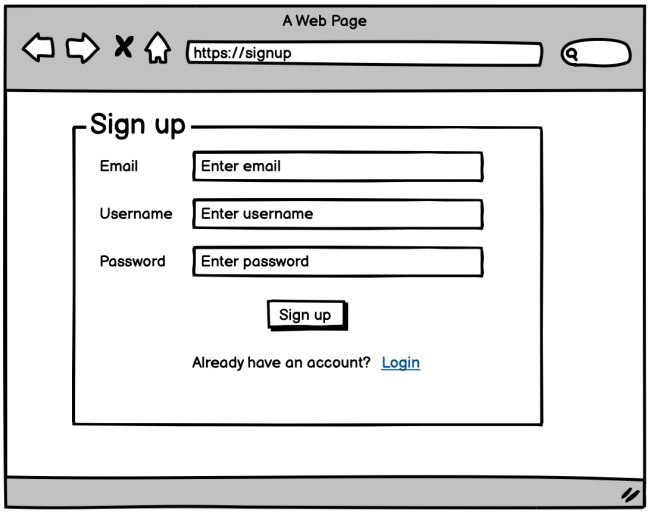


Figure 3: Sign up page

4.3 Login page (Interface mockup)

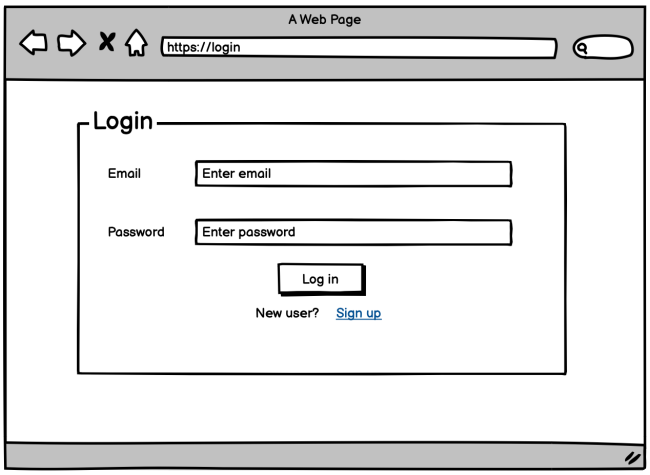


Figure 4: Login page

The login form in Figure 4 is similar to the sign up page. If a user is not registered, they can proceed to the sign up page. To enter the system a user must provide their email and password that they entered when signing up.

4.4 Search results (Interface mockup)

The search page in Figure 5 displays a table of results for a book search. The table contains the following information: ISBN number, name, release date, and the publisher of the book. If there are no matches in the database, "No matches found" message will be displayed. User can navigate back to search for another book, author or user.

The search form in Figure 6 displays a table of results for author search. The table contains the name of the author and their biography. If there no authors are matched in the database, "No matches found" message will

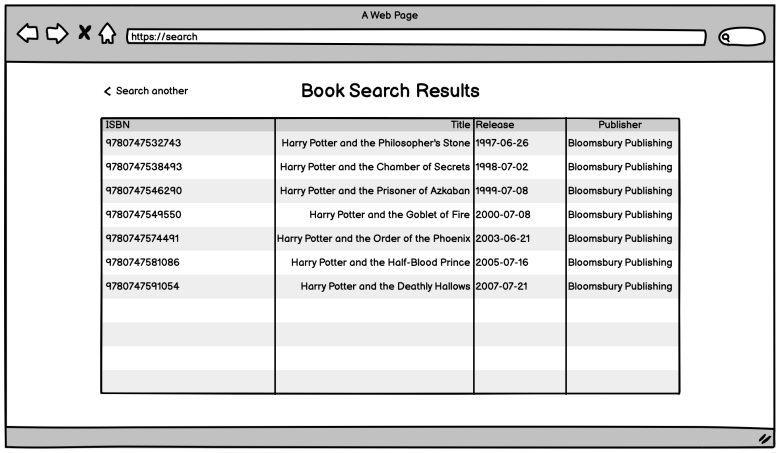


Figure 5: Search book results page

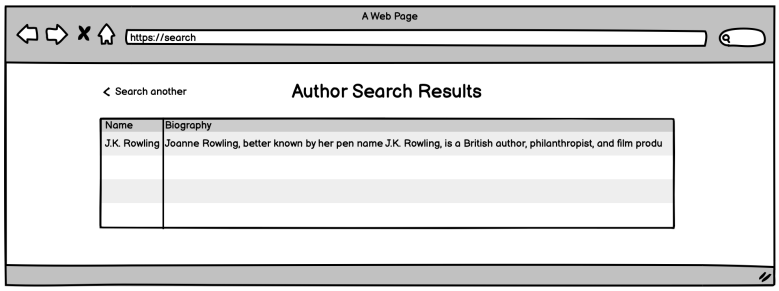


Figure 6: Search author results page

be displayed. The user search is similar, the table of user search will display a table of matched users, so that the user can explore other users' recommendations by navigating to their profiles.

5 Business Logic Layer

5.1 Class Diagram

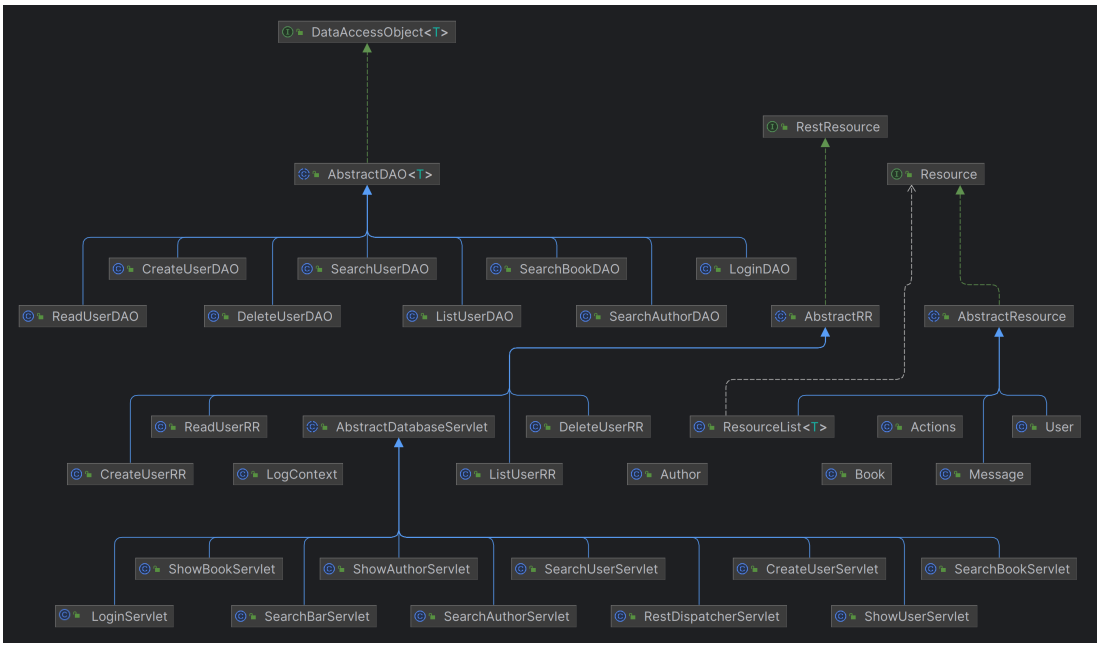
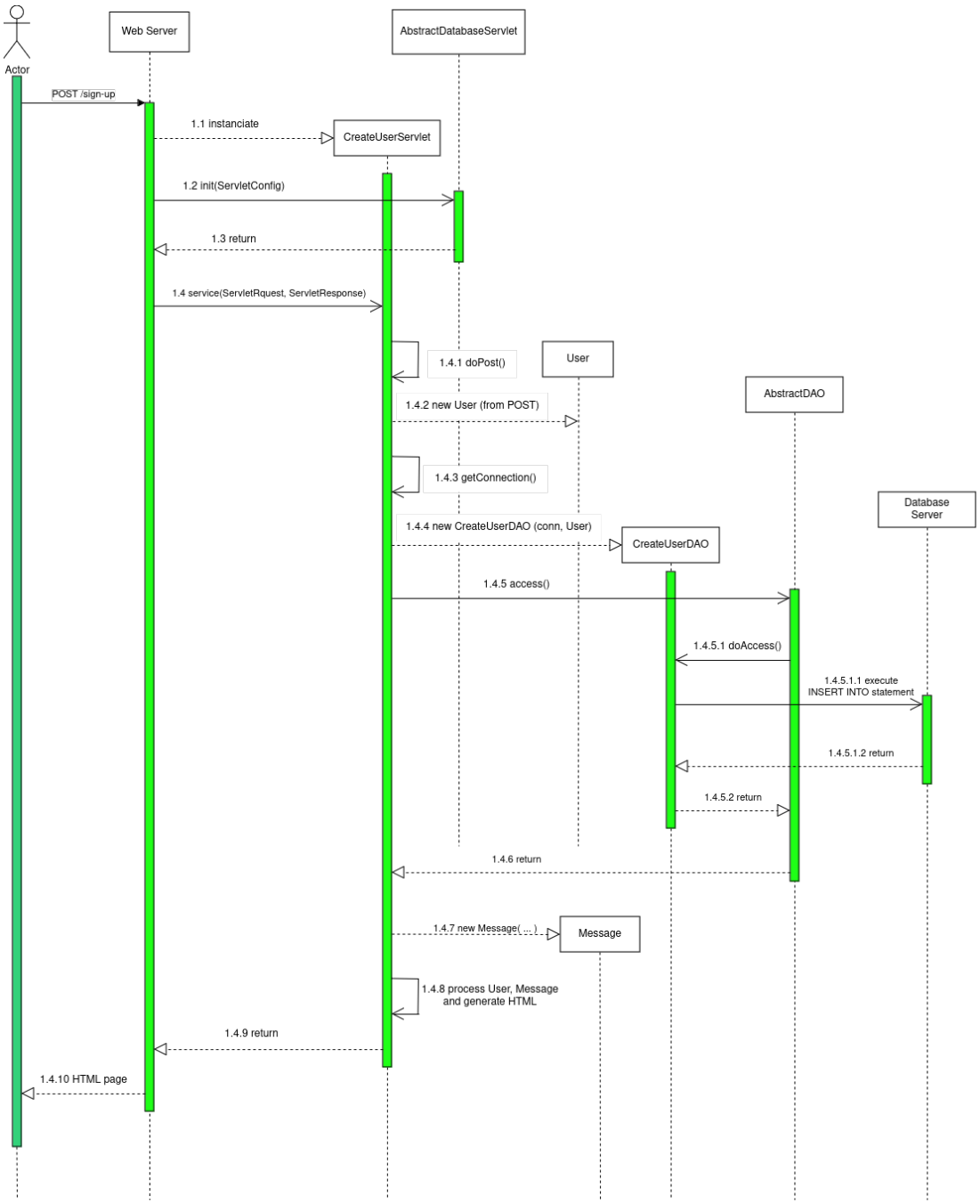


Figure 7: Class diagram of the book recommendation system

The class diagram represents the structure of a web-based application for managing users, books, and authors in a book recommendation system. The classes can be divided into the following categories:

- **Data Access Objects (DAO)**: This category includes abstract classes like AbstractDAO and concrete classes like CreateUserDAO, SearchUserDAO, SearchBookDAO, LoginDAO, ReadUserDAO, DeleteUserDAO, ListUserDAO, and SearchAuthorDAO. These classes handle data access operations related to users, books, and authors, such as creating, reading, updating, and deleting data.
- **Resource Classes**: The ‘Resource‘ class and its subclasses ‘RestResource‘, ‘AbstractRR‘, and ‘AbstractResource‘ represent resources in the application, which could be related to users, books, authors, or other entities.
- **Entity Classes**: Classes like ‘User‘, ‘Book‘, ‘Author‘, ‘Actions‘, ‘Message‘, and ‘ResourceList‘ represent the entities or data models in the application.
- **Servlet Classes**: Classes like ‘ShowBookServlet‘, ‘ShowAuthorServlet‘, ‘SearchUserServlet‘, ‘CreateUserServlet‘, ‘SearchBookServlet‘, ‘LoginServlet‘, ‘SearchBarServlet‘, ‘SearchAuthorServlet‘, ‘RestDispatcherServlet‘, and ‘ShowUserServlet‘ are servlets that handle HTTP requests and responses, and interact with the DAO and resource classes to perform the required operations.
- **Utility Classes**: Classes like ‘DataAccessObject‘, ‘AbstractDatabaseServlet‘, and ‘LogContext‘ provide utility functions or shared functionality across the application.

5.2 Sequence Diagram



The sequence diagram explain what happens when a new user register on the website, operations which can be summarised in:

1. The web server see a POST request on the link /sign-up and forward the request to the servlet mapped in that link, in this case CreateUserServlet.

2. After initial configuration, the servlet creates a new User object with the data from the POST request received.

3. This object is passed to the CreateUserDAO class, which execute the statement to insert that new user in the database.
4. Once the DAO has done its job, the servlet check the reurn to see if everything went well and it creates a Message object to be displayed on the page and to be saved as log of that operation.
5. The message is printed on a HTML page which is sent to the web server and then shown to the user.

5.3 REST API Summary

URI	Method	Description	Filter
/rest/user	GET	Retrieve a list of users	Yes
/rest/user/name	GET	Retrieve information about a specific user	Yes
/rest/user/name/update	PUT	Update information for a specific user	Yes
/rest/user/name/delete	DELETE	Delete a specific user	Yes

Table 2: Describe in this table your REST API

5.4 REST Error Codes

Error Code	HTTP Status Code	Description
400	Bad Request	The server cannot process the request due to a client error. This could be due to malformed syntax or invalid parameters.
401	Unauthorized	The request requires user authentication. The client must authenticate itself to get the requested response.
404	Not Found	The server cannot find the requested resource. This could be due to a mistyped URL or the resource being deleted.
500	Internal Server Error	The server encountered an unexpected condition that prevented it from fulfilling the request.

Table 3: Describe in this table your REST API

5.5 REST API Details

BookRec Resource

The BookRec resource allows retrieval of book recommendations.

- URL: `http://localhost:8080/BookRec/rest/user`
- Method: `GET`
- URL Parameters:None
- Data Parameters: None
- Success Response:
  - Status Code: 200 OK
  - Content:

```
{
    "resource-list": [
        {"User": {"name": "vin", "email": "email@empty.com"}},
        {"User": {"name": "abc", "email": "email@empty.com"}},
        {"User": {"name": "abcd", "email": "email@empty.com"}},
        {"User": {"name": "alpha", "email": "email@empty.com"}}
    ]
}
```

- Error Response:
  - Status Code: 404 Not Found
  - Content: Error message indicating resource not found

6 Group Members Contribution

Stefano Bortolatto I worked mostly on the data logic part, by making the ER schema and helping in the SQL implementation and connection with the website. Other contributions are on documentation, registration and search code and sequence diagram for the signup process.

Vinayak Kumar I worked on getting the Project and Servlet started, Getting war file and deploying on tomcat.Majority of time was spent on debugging the class instantiation error(which was because old tomcat version).Added Rest Template and made it work.Implemented Class diagram as well.

Farhad Sadat I have been working on the connection between database and application and I also try to work on the data logic and main functionality.

Ramilya Omarbekova I worked on the Presentation logic layer, made the mockups of the application.