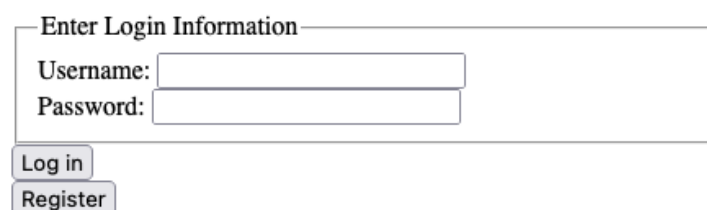


Walkthrough of web application - A diary application with user accounts

The application was implemented using Jakarta servlets, JPA, Hibernate, Tomcat, MySQL and MongoDB.

[Log In](#)

My Diary



Enter Login Information

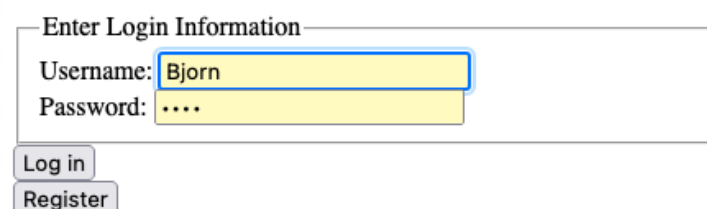
Username:

Password:

Login page. The first page the user can visit upon application start is located at http://localhost:8081/Journeyman_war_exploded/login. The LoginServlet renders a static file, login.html, the result seen in the image above. An HttpSession object is created for the user if one does not exist.

[Log In](#)

My Diary



Enter Login Information

Username:

Password:

The user can enter Username and Password to log in or press “Register” to get to the registration page.

[Log In](#)

My Diary

Enter Login Information

Username:

Password:

Incorrect username or password. New user?

Log in

Register

A non-registered user trying to log in or a user that fills in incorrect credentials, which makes the backend authentication process fail, receives a message about this at “/login”. The application has then gone through the authentication process and returns a user message explaining what went wrong. The user may try again or press the “Register” button to go to the registration page.

[Log In](#)

Register New User

Enter User Information

Username:

Password:

Email:

Register

A non-registered user can access the registration page (shown in the image above) at “/registration” via the login page “Register” button. Registering a new user account requires all form fields to be filled in. Input validation of the form input is the first server-side process to be executed. If the validation process fails, the registration page is displayed with a user message explaining what went wrong. *(See below, wrong email is left out for breviation)*

[Log In](#)

Register New User

Enter User Information

Username:

Password:

Email:

Username must be between 4 and 8 characters long and can only include uppercase and lowercase letters, and digits.

Register

The user inputs an incorrect format on the username.

[Log In](#)

Register New User

Enter User Information

Username:

Password:

Email:

Password must be between 4 and 8 characters long, and must include at least one uppercase letter, one lowercase letter, one digit, and one special character from this set: !@#\$%^&*.

Register

The user inputs an incorrect format on the password.

[Log In](#)

Account Verification

A verification code was sent to your registered email. Please enter code below to verify your account.

Enter Verification Code:

Verify Account

Send New Code

A successful registration authentication results in the account verification page being displayed to the user by the VerificationServlet. On the server-side the EmailSenderServlet has been invoked and has sent an email to the registered email address with a randomly generated 6-digit verification code. The verification code is hashed and persisted in a collection in MongoDB. The user always has the option to send a new verification code by pressing the “Send New Code” button. This generates a new code server-side, sends a new email, and then renders the verification page again.

[Log In](#)

Account Verification

Invalid Verification Code - Try again or send new code.

Enter Verification Code:

If the user enters an invalid verification code, one that cannot be found together with the users registered email address in the MongoDB database, a user message is displayed.

[Log In](#)

Account Verification

Verification code sent to email.

Enter Verification Code:

Above is the view rendered when the user has requested a new verification code is through the “Send New Code” button.

[Log In](#)

Account Verification

Verification code sent to email.

Enter Verification Code:

Enter the correct verification code received in the email. Press “Verify Account”. The server-side checks for correctness between email, the entered code and the hashed code in the database.

[Log In](#)

My Diary

Enter Login Information

Username:

Password:

Account activated! Please log in.

Log in

Register

The account is activated. The user is redirected back to the login page.

If the user in any way aborts the verification processes during registration, the application will not allow the user to log in until the verification process is completed. On the server-side the new user will be registered and persisted in the database with a boolean flag “isActive” set to false until this process is completed.

User page walkthrough

[Settings](#)

[Log Out](#)

The Diary Stories of Bjorn

New Diary Post

Add A Title (max 200 characters):

Add Diary Post (max 20.000 characters):

Diary post

Add An Image (optional, supported: jpeg, png, gif; max file size 5MB):

Browse...

No file selected.

Add Image Caption (optional, max 200 characters):

Post!

Upon completing registration and verification, the user can log in. The image above displays the rendered user page, which dynamically inserts the name of the logged-in user. Upon successful login, a session attribute “loggedIn” has been set to the user session. This session attribute must be present to visit any of the pages/servlets with paths starting with “/user/*”. Otherwise, the LoginFilter will redirect the user to the login page.

At the user page, the user can view their diary posts and submit new ones. The requirements stated by the input field labels are validated on the server-side, and if incorrect, a new post will not be persisted. Instead, a user message with what failed will be presented. However, the HTML attributes “maxlength” and “accept” restrict the user on the client-side and therefore the user message will not be displayed if the user doesn’t bypass the HTML.

[Settings](#)

Log Out

The Diary Stories of Bjorn

New Diary Post

Add A Title (max 200 characters):

My first post!

Add Diary Post (max 20.000 characters):

Today was a lovely day...

Add An Image (optional, supported: jpeg, png, gif; max file size 5MB):

Browse... No file selected.

Add Image Caption (optional, max 200 characters):

Image Caption

Post!

The user enters a title and a post.

The screenshot shows a file explorer window with the 'Downloads' folder selected. The file list is as follows:

May	Size	Kind	Date Added
IMG_1574.JPG	907 KB	JPEG image	5 May 2023 at 17:50
IMG_1575.JPG	1 MB	JPEG image	5 May 2023 at 17:50
IMG_1576.JPG	537 KB	JPEG image	5 May 2023 at 17:50
IMG_1577.JPG	764 KB	JPEG image	5 May 2023 at 17:50
IMG_1578.JPG	604 KB	JPEG image	5 May 2023 at 17:50
IMG_1579.JPG	733 KB	JPEG image	5 May 2023 at 17:50
IMG_1580.JPG	928 KB	JPEG image	5 May 2023 at 17:50
IMG_1581.JPG	1,1 MB	JPEG image	5 May 2023 at 17:50
IMG_1582.JPG	790 KB	JPEG image	5 May 2023 at 17:50
IMG_1583.JPG	1 MB	JPEG image	5 May 2023 at 17:50
IMG_1584.JPG	1 MB	JPEG image	5 May 2023 at 17:50
IMG_1585.JPG	1,3 MB	JPEG image	5 May 2023 at 17:50
IMG_1586.JPG	1,5 MB	JPEG image	5 May 2023 at 17:50
IMG_1587.JPG	1,6 MB	JPEG image	5 May 2023 at 17:50
IMG_1588.JPG	1,5 MB	JPEG image	5 May 2023 at 17:50
IMG_1589.JPG	928 KB	JPEG image	5 May 2023 at 17:50

At the bottom of the window, there are buttons for 'Show Options', 'Cancel', and 'Open'.

The user selects the “Browse” button to enable an image file upload to be added to the post.

The Diary Stories of Bjorn

New Diary Post

Add A Title (max 200 characters):

Add Diary Post (max 20.000 characters):

Today was a lovely day...

Add An Image (optional, supported: jpeg, png, gif; max file size 5MB):

Browse...

IMG_1580.JPG

Add Image Caption (optional, max 200 characters):

Hanging out with some friends in the sun

Post!

Image is selected. The user adds an image caption for describing the image. Presses "Post!" to persist the post.

My first post!

Post: #1

Posted: 2023-08-10 13:25:50.665

Today was a lovely day...



Hanging out with some friends in the sun

The post is created and presented below the form.

Post!

My first post!

Post: #1

Posted: 2023-08-10 13:25:50.665

Today was a lovely day...



Hanging out with some friends in the sun

Day two of writing a diary

Post: #2

Posted: 2023-08-10 13:27:38.578

This seems like a nice habit

Another post is made.

[Back To Diary](#)

Settings

Delete Account

The user presses the “settings” link on the user page. The settings page offer the choice of deleting the account or returning back to user page.

[Back To Settings](#)

Delete Account

Are you sure? This operation cannot be undone.

Confirmation

Yes! Delete My Account

The user presses the “Delete Account” and goes to the delete account page.

[Settings](#)

Log Out

The Diary Stories of Bjorn

New Diary Post

Add A Title (max 200 characters):

Title

The user decides not to delete account and goes back to the user page by following the links, "Back To Settings" then "Back To Diary".

[Log In](#)

My Diary

Enter Login Information

Username:

Password:

Successful logout.

Log in

Register

The user presses the "Log Out" button, which invalidates the session and passes the user message "Successful logout." to the LoginServlet in the request parameters.

[Back To Settings](#)

Delete Account

Are you sure? This operation cannot be undone.

Confirmation

Yes! Delete My Account

The user decides to delete the account, logs in again, navigates to the delete account page and presses the "Yes! Delete My Account" button.

[Log In](#)

My Diary

Enter Login Information

Username:

Password:

Account deletion successful.

Log in

Register

The user is redirected to the login page and is notified that the account deletion has been successful.

[Log In](#)

My Diary

Enter Login Information

Username:

Password:

This account has been deactivated. If you think this is a mistake, please contact support.

Log in

Register

If the user tries to log in with the user credentials, a user message is displayed. A soft deletion has been made in the MySQL database. The User instance property "isDeleted" has been set to true in the database, but the entity is still persisted.

Example of database records

```
mysql> SELECT * FROM Users;
ERROR 4031 (HY000): The client was disconnected by the server because of inactivity. See wait_timeout and interactive_timeout for configuring this behavior.
No connection. Trying to reconnect...
Connection id: 200
Current database: user
```

is_active	is_deleted	id	email	password	username
0x01	0x00	1	forsberg.bjorn.a@gmail.com	\$argon2i\$v=19\$m=65536,t=10,p=1\$ep1QAxxUNj0vX+qRssPFYA\$CKA7xwFJJQDVEptgx0xMbjV9/X+J2FxtSVzv0RxY2Q	Bjorn

```
1 row in set (0.04 sec)
```

View of the Users table in the MySQL database.

```
mysql> SELECT * FROM Diary_Posts;
```

id	image_id	timestamp	user_id	imageCaption	post	title
1	1	2023-08-10 18:09:38.218000	1	fgsdfg	sdfhsdfgh	fdg
2	2	2023-08-10 18:09:53.691000	1	sdfgsdfg	sfdgsdfg	dsfgsdfg

```
2 rows in set (0.00 sec)
```

View of the DiaryPosts table in the MySQL database.

id	mimeType	imageData
2	image/jpeg	0xFFD8FFE000106950686F6E652031332050726F000000004E90120002000000007000002C6910100070003000400000001000000834A217000300000032313A33353A3030002B30313A30300000008000A00000003000004D4000C000A000A0000000001000000003000700000000000

Columns in the Images table.

2	image/jpeg	0xFFD8FFE000106950686F6E652031332050726F000000004E90120002000000007000002C6910100070003000400000001000000834A217000300000032313A33353A3030002B30313A30300000008000A00000003000004D4000C000A000A0000000001000000003000700000000000
---	------------	---

Example row in the Images table.

```
test> use codeverification
switched to db codeverification
codeverification> db.verifications.find()
[
  {
    _id: ObjectId("64cba6b83980fd049dc603c7"),
    email: 'foo@foooooo.se',
    hashedCode: '$argon2i$v=19$m=65536,t=10,p=1$yEKkollu9e2ej0G1wYFgIA$mIzTSPJm7vohoAK3mMxy1Rw8qwPMCfrKfcRbBLBz8gA'
  },
  {
    _id: ObjectId("64ccdd8455ceb4067d0eb1f9"),
    email: 'max@max.se',
    hashedCode: '$argon2i$v=19$m=65536,t=10,p=1$HUK3IdLsvCYMChQWyuyl1A$qnnCma9tU830W/y1at52NusBOAT5wc8zTFMMQa9t/oY'
  },
  {
    _id: ObjectId("64cfd56b8ed4ca12040bdf7f"),
    email: 'bakka@bakka.com',
    hashedCode: '$argon2i$v=19$m=65536,t=10,p=1$IxyTacL9zd+GsOaW2FMnFQ$8T5FzNVgMsMh2w/ko4RRZ9hNGZQSLX47z5GB9CEXAsw'
  },
  {
    _id: ObjectId("64d49b5e2b3c9e5e6935acf8"),
    email: 'fo@foo.se',
    hashedCode: '$argon2i$v=19$m=65536,t=10,p=1$c/NPZo3LsWkbNxH6+fYs2A$d20uuAWWbpNdDrhaAN4rB0EVowBrzP/L3fhRr6p95iA'
  }
]
codeverification> █
```

Verifications collection in the MongoDB database codeverification.