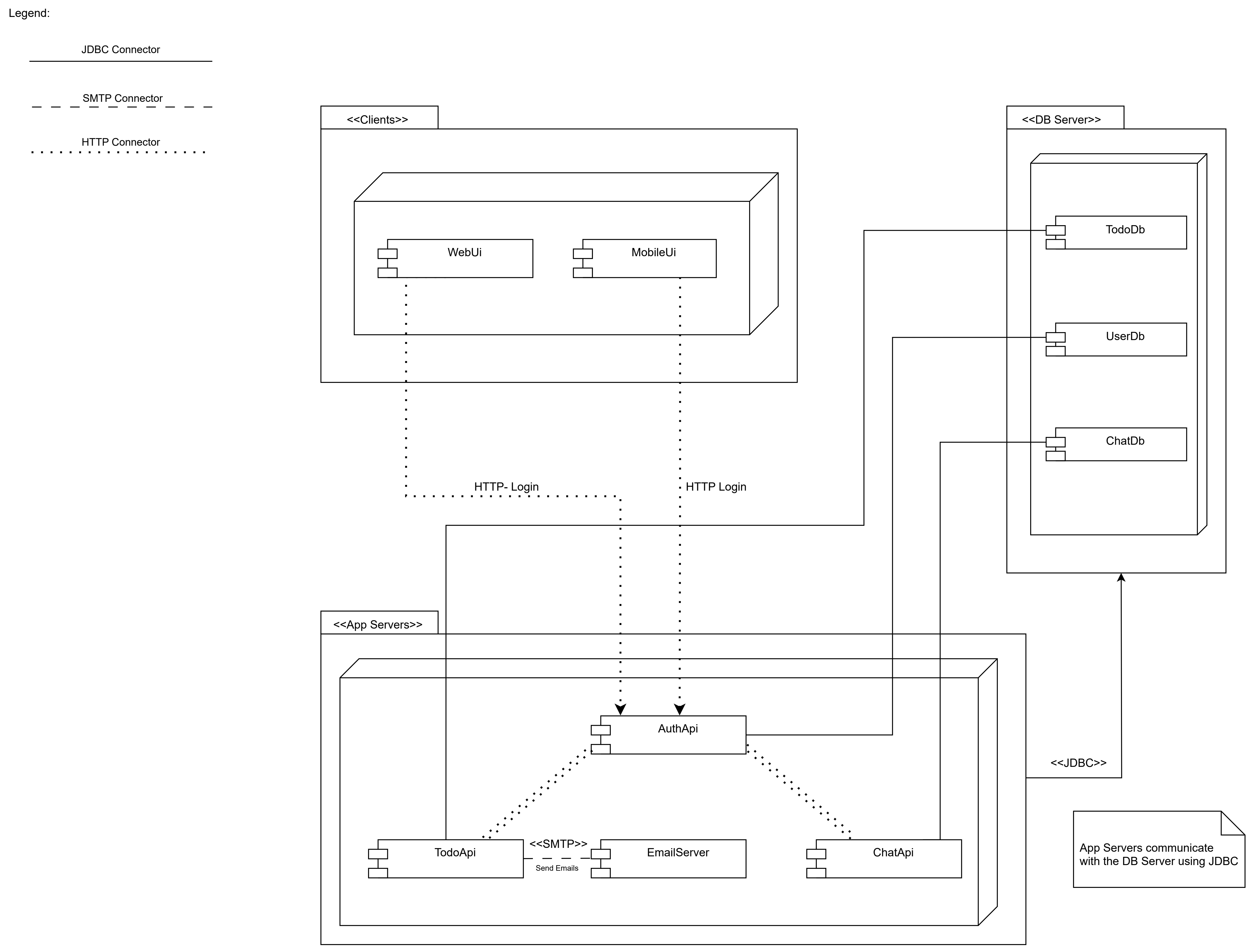
Legend:
JDBC Connector
SMTP Connector
HTTP Connector
<<Clients>>
WebUi
MobileUi
<<DB Server>>
TodoDb
UserDb
ChatDb
HTTP- Login
HTTP Login
<<App Servers>>
AuthApi
TodoApi
<<SMTP>>
Send Emails
EmailServer
ChatApi
<<JDBC>>
App Servers communicate with the DB Server using JDBC

Aufgabe 2:

2.1

a)

```
CREATE TABLE IF NOT EXISTS todos (
    id INTEGER PRIMARY KEY,
    title varchar(100) NOT NULL DEFAULT 'New todo',
    description varchar(500)
);
```

b)

```
INSERT INTO todos (id, title, description)
VALUES (1, 'dekorieren', 'Es ist nun endlich so weit! Mit dem 01. November wird es Zeit, zügig die Weihnachtsdekorationen auszupacken.');
```

c)

```
SELECT description FROM todos WHERE description LIKE '%Weihnacht%';
```

Ergebnisse:

- Es ist nun endlich so weit! Mit dem 01. November wird es Zeit, zügig die Weihnachtsdekorationen auszupacken.
- Bald sollte ich Weihnachtsplaetzchen backen.

2.2

Ergebnisse:

a)

```
EntwickLUnGPrOgrAMMII.
```

b)

```
 This is the Id for the Letter 'V': 52
This is the Id for the Letter 'V': 78
This is the Id for the Letter 'b': 9
This is the Id for the Letter 'b': 32
This is the Id for the Letter 'b': 58
This is the Id for the Letter 't': 50
This is the Id for the Letter 't': 76
```

c)

```
This is the sum of the ID's: 4167
This is the avg of the ID's: 50.81707317073171
```

Code:

```java
package org.example.a;

import com.j256.ormlite.field.DatabaseField;
import com.j256.ormlite.table.DatabaseTable;

@DatabaseTable(tableName = "letters")
public class Letter {

    @DatabaseField(id = true)
    private Integer id;

    @DatabaseField(columnName = "letter")
    private String letter;

    public Letter() {
    }

    public Integer getId() {
        return id;
    }

    public String getLetter() {
        return letter;
    }

    public void setId(Integer id) {
        this.id = id;
    }

    public void setLetter(String letter) {
        this.letter = letter;
    }


}

```

```java
package org.example.a;

import com.j256.ormlite.dao.Dao;
import com.j256.ormlite.dao.DaoManager;
import com.j256.ormlite.jdbc.JdbcConnectionSource;
import com.j256.ormlite.support.ConnectionSource;

import java.io.IOException;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;

public class Main {


    public static void main(String[] args) {

        try {

            ConnectionSource connectionSource = new JdbcConnectionSource("jdbc:mariadb://bilbao.informatik.uni-stuttgart.de/pe2-db-a1", "pe2-nutzer", "esJLtFm6ksCT4mCyOS");
            Dao<Letter, Integer> letterDao = DaoManager.createDao(connectionSource, Letter.class);

            /* Aufgabe 2.2 a) */

            int[] arrayIndexes = {
                    20, 44, 50, 13, 17, 33, 41,
                    68, 77, 44, 29, 72, 48, 71,
                    37, 48, 11, 69, 5, 65, 65
            };

            List<Letter> letterList = new ArrayList<>();

            for (int i : arrayIndexes) {
                Letter letter = letterDao.queryForId(i);
                letterList.add(letter);
            }

            StringBuilder stringBuilder = new StringBuilder();

            for (Letter letter : letterList) {
                stringBuilder.append(letter.getLetter());
            }

            System.out.println(stringBuilder.toString());


            /* Aufgabe 2.2 b) */

            List<Letter> vId = letterDao.queryForEq("letter", "V");
            List<Letter> bId = letterDao.queryForEq("letter", "b");
            List<Letter> tId = letterDao.queryForEq("letter", "t");

            for (Letter v : vId) {
                System.out.println("This is the Id for the Letter 'V': " + v.getId());
            }

            for (Letter b : bId) {
                System.out.println("This is the Id for the Letter 'b': " + b.getId());
            }

            for (Letter t : tId) {
                System.out.println("This is the Id for the Letter 't': " + t.getId());
            }


            /* Aufgabe 2.2 c) */
            List<Letter> letters = letterDao.queryForAll();
            List<Integer> letterIDs = new ArrayList<>();
            Integer sum = 0;
            double avg = 0;

            for (Letter letter : letters) {
                Integer id = letter.getId();
                letterIDs.add(id);
            }

            //calculate the sum of the IDs
            for (Integer id : letterIDs) {
                sum += id;
            }

            //calculate the avg of the IDs
            avg = (double) sum / letterIDs.size();

            System.out.println("This is the sum of the ID's: " + sum);
            System.out.println("This is the avg of the ID's:" + avg);

            connectionSource.close();
        } catch (SQLException | IOException e) {
            throw new RuntimeException(e);
        }


    }
}
```

Aufgabe **3:**

a)

```
{
  "categories": [
    "history"
  ],
  "created_at": "2020-01-05 13:42:19.576875",
  "icon_url": "https://api.chucknorris.io/img/avatar/chuck-norris.png",
  "id": "rqcvwdgqq6amwony3nngba",
  "updated_at": "2020-01-05 13:42:19.576875",
  "url": "https://api.chucknorris.io/jokes/rqcvwdgqq6amwony3nngba",
  "value": "In the Words of Julius Caesar, \"Veni, Vidi, Vici, Chuck Norris\". Translation: I came, 
I saw, and I was roundhouse-kicked inthe face by Chuck Norris."
}
```

b)

```
{
 "args": {},
 "data": {
  "key": "pe2ws23",
  "purpose": "This is a test."
 },
 "files": {},
 "form": {},
 "headers": {
  "host": "postman-echo.com",
  "x-request-start": "t=1730484190.270",
  "connection": "close",
  "content-length": "59",
  "x-forwarded-proto": "https",
  "x-forwarded-port": "443",
  "x-amzn-trace-id": "Root=1-672517de-0baf8e62597b27d25d0ebc8a",
  "content-type": "application/json",
  "user-agent": "PostmanRuntime/7.42.0",
  "accept": "*/*",
  "postman-token": "5c0a671e-e26c-4422-ac8b-18903a8f1813",
  "accept-encoding": "gzip, deflate, br"
 },
 "json": {
  "key": "pe2ws23",
  "purpose": "This is a test."
 },
 "url": "https://postman-echo.com/post"
```

}

c)

1.Erstellen einer neuen DVD:

<u>POST /dvds</u>

2.Abholen einer DVD Über eine ID:

<u>GET /dvds/$id</u>

3.Aktualisiere eine DVD über eine ID:

<u>PUT /dvds/$id</u>
(Aktualisiert die Daten der DVD mit der übergebenen ID. Die neuen Daten, werden im Request-Body übergeben)

4.Löschen einer DVD über eine ID:

<u>DELETE /dvds/$id</u>

5.Abholen aller DVDs mit Filteroptionen

<u>GET /dvds?category=$category&ageRestricted=$ageRestricted&title=$title</u>

<u>Beispiel:</u> GET /dvds?category=SciFi&ageRestricted=false&title=TheMovie