

BSD

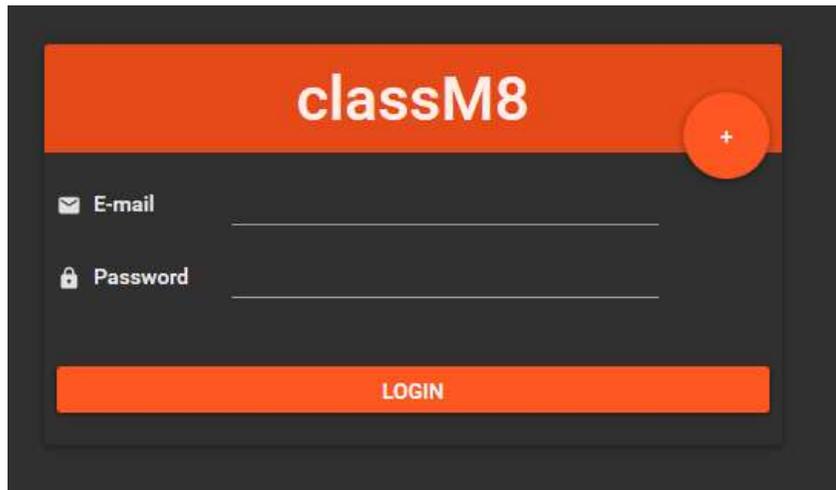
# ClassM8

Sprint 2 Release



# C# - Frontend

# Login Window



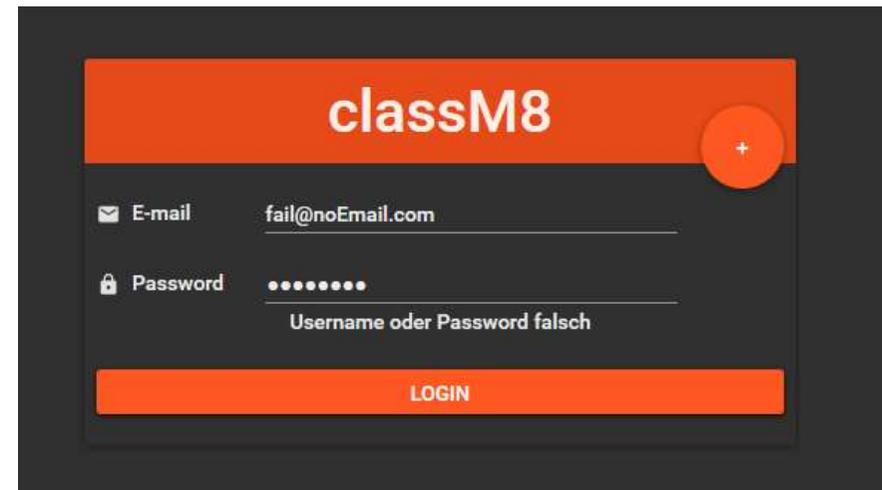
classM8

E-mail \_\_\_\_\_

Password \_\_\_\_\_

LOGIN

The image shows a login window for 'classM8'. It has a dark grey background with an orange header bar containing the text 'classM8' and a circular orange button with a white plus sign. Below the header are two input fields: 'E-mail' with an envelope icon and 'Password' with a lock icon. At the bottom is an orange 'LOGIN' button.



classM8

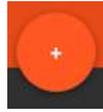
E-mail fail@noEmail.com \_\_\_\_\_

Password ●●●●●● \_\_\_\_\_

Username oder Passwort falsch

LOGIN

The image shows the same login window for 'classM8' but with a failed login attempt. The 'E-mail' field contains 'fail@noEmail.com' and the 'Password' field is filled with seven dots. Below the password field, the text 'Username oder Passwort falsch' is displayed. The 'LOGIN' button remains at the bottom.

Wenn der  - Button gedrückt wird öffnet sich folgender Dialog - **Benutzer Anlegen**

## Benutzer anlegen

Vorname

Nachname

E-mail

Passwort

Passwort bestätigen

## Benutzer anlegen

Vorname

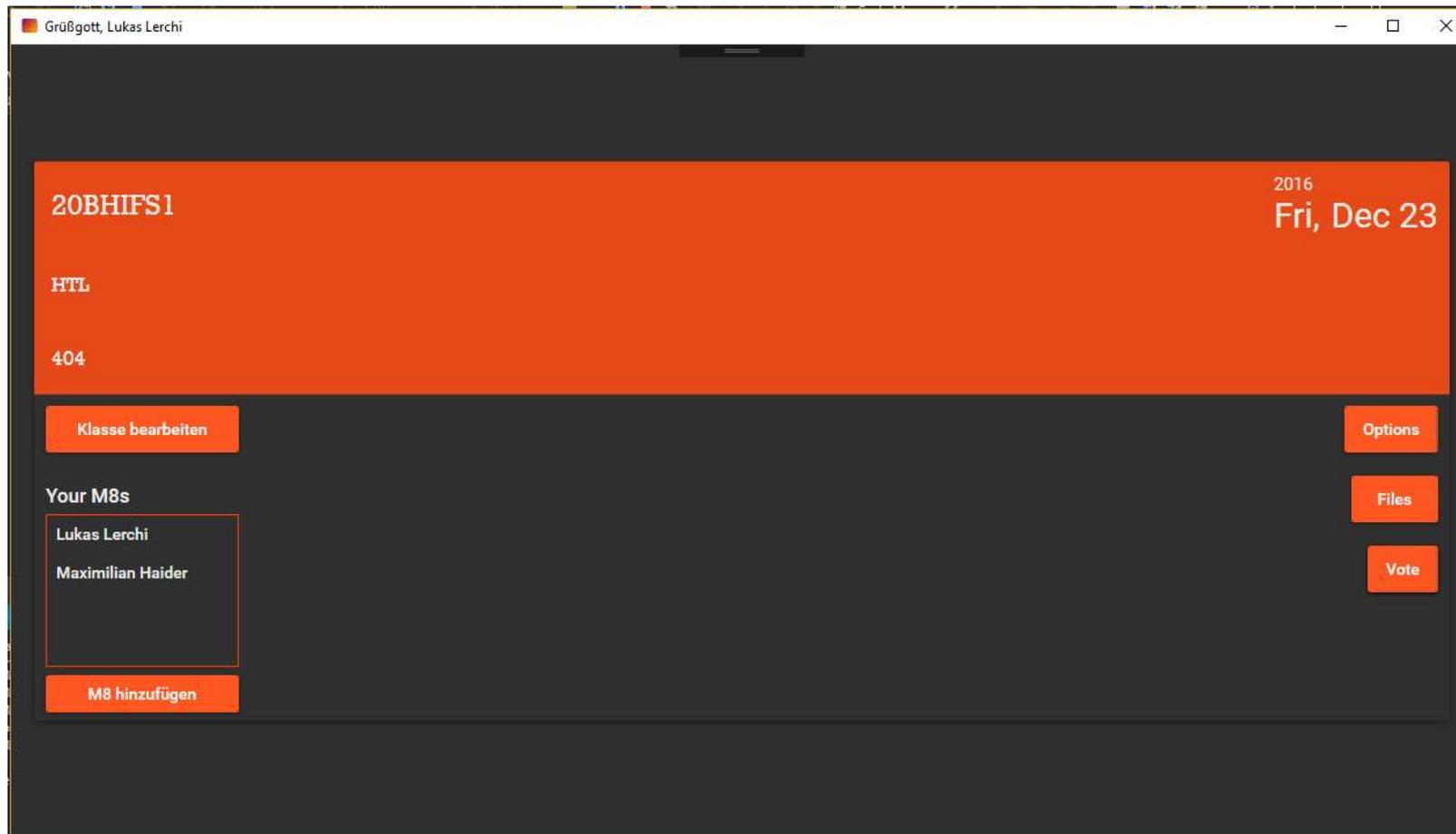
Nachname

E-mail

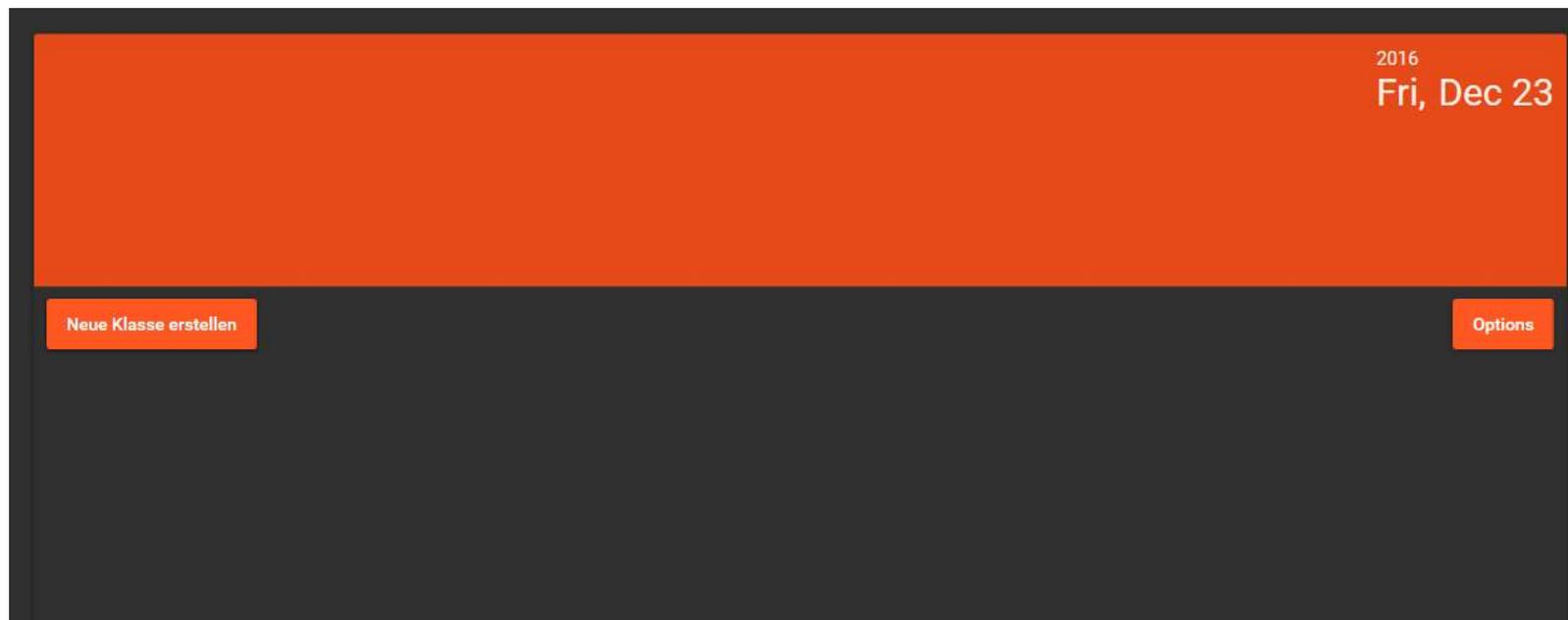
Passwort

Passwort bestätigen

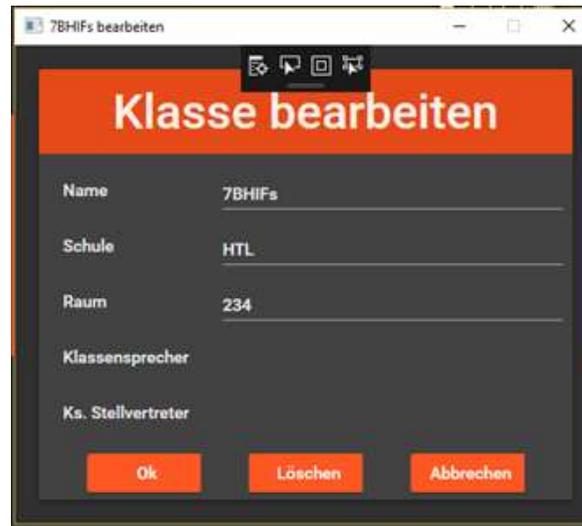
Nachdem man sich erfolgreich eingeloggt hat kommt man zur **Home View**



Wenn man sich mit einem User der in keiner Klasse ist anmeldet....



# Klasse Bearbeiten

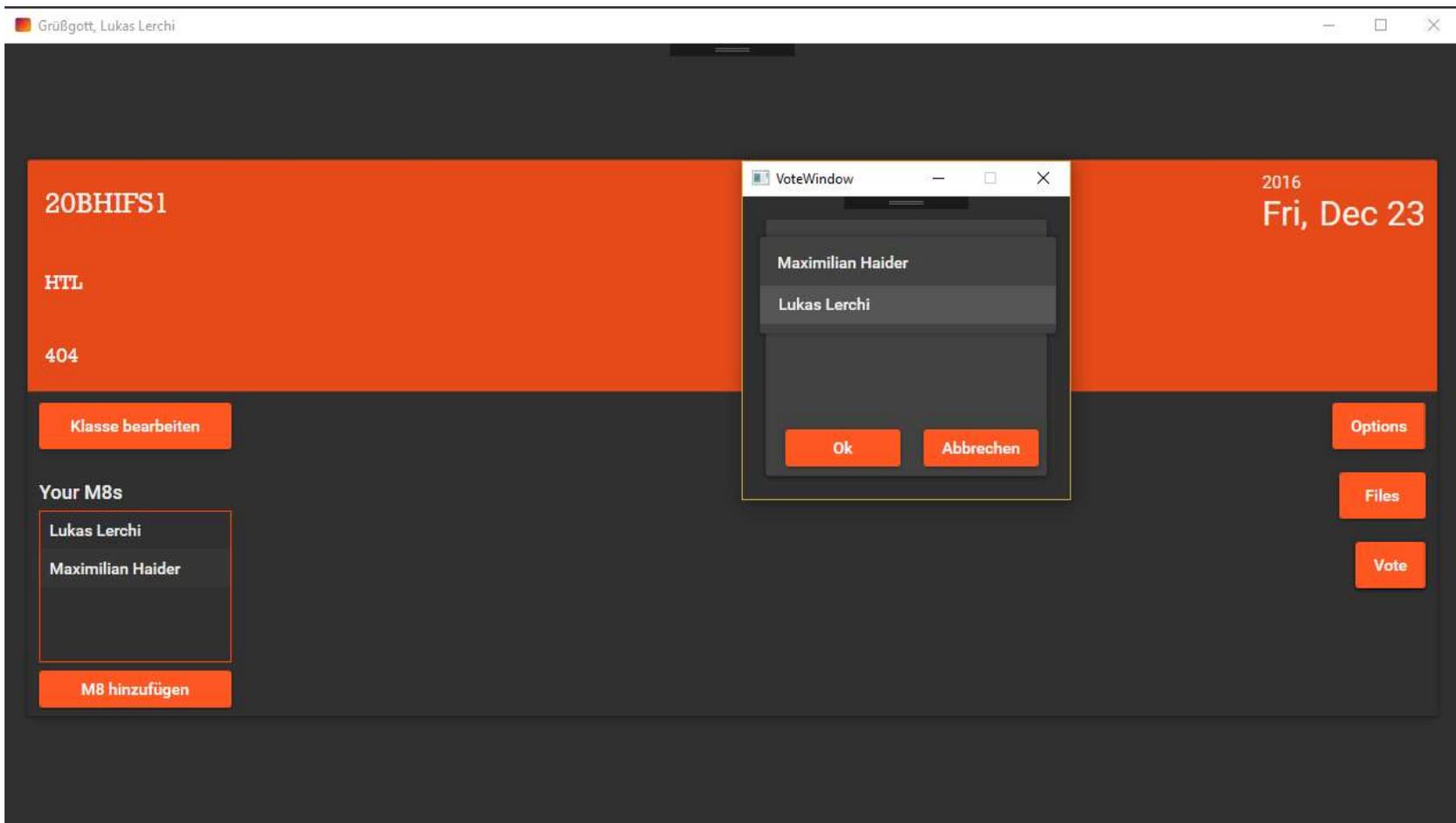


The image shows a dialog box titled "7BHIFs bearbeiten" with a dark gray background and an orange header bar. The header bar contains the text "Klasse bearbeiten" in white. Below the header, there are several input fields with labels on the left and values on the right:

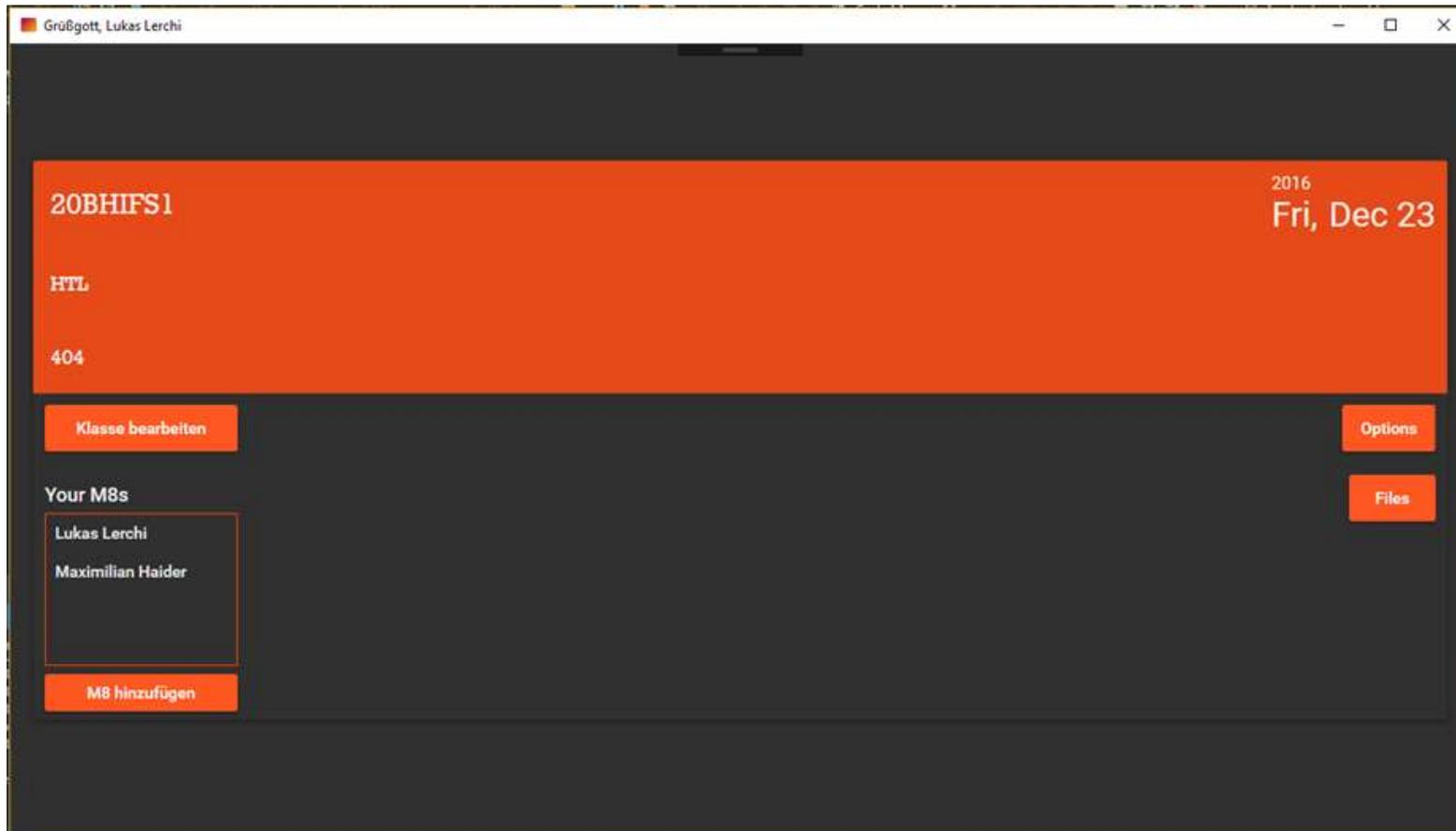
Name	7BHIFs
Schule	HTL
Raum	234
Klassensprecher	
Ks. Stellvertreter	

At the bottom of the dialog, there are three orange buttons: "Ok", "Löschen", and "Abbrechen".

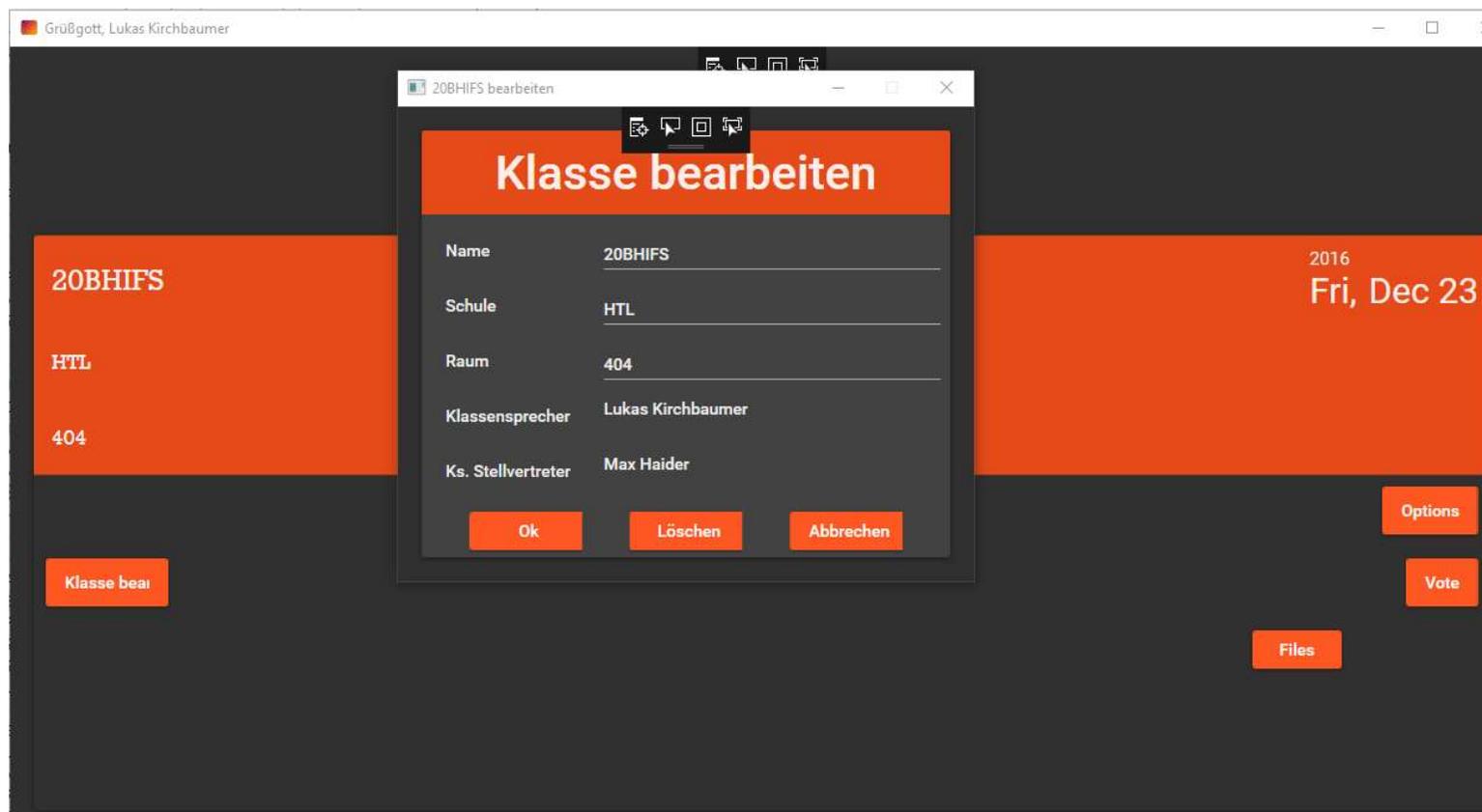
# Klassensprecherwahl (Vote)



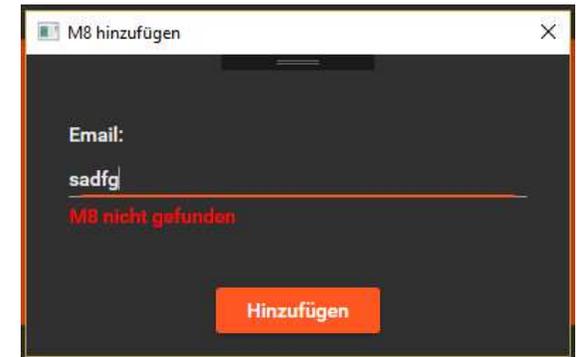
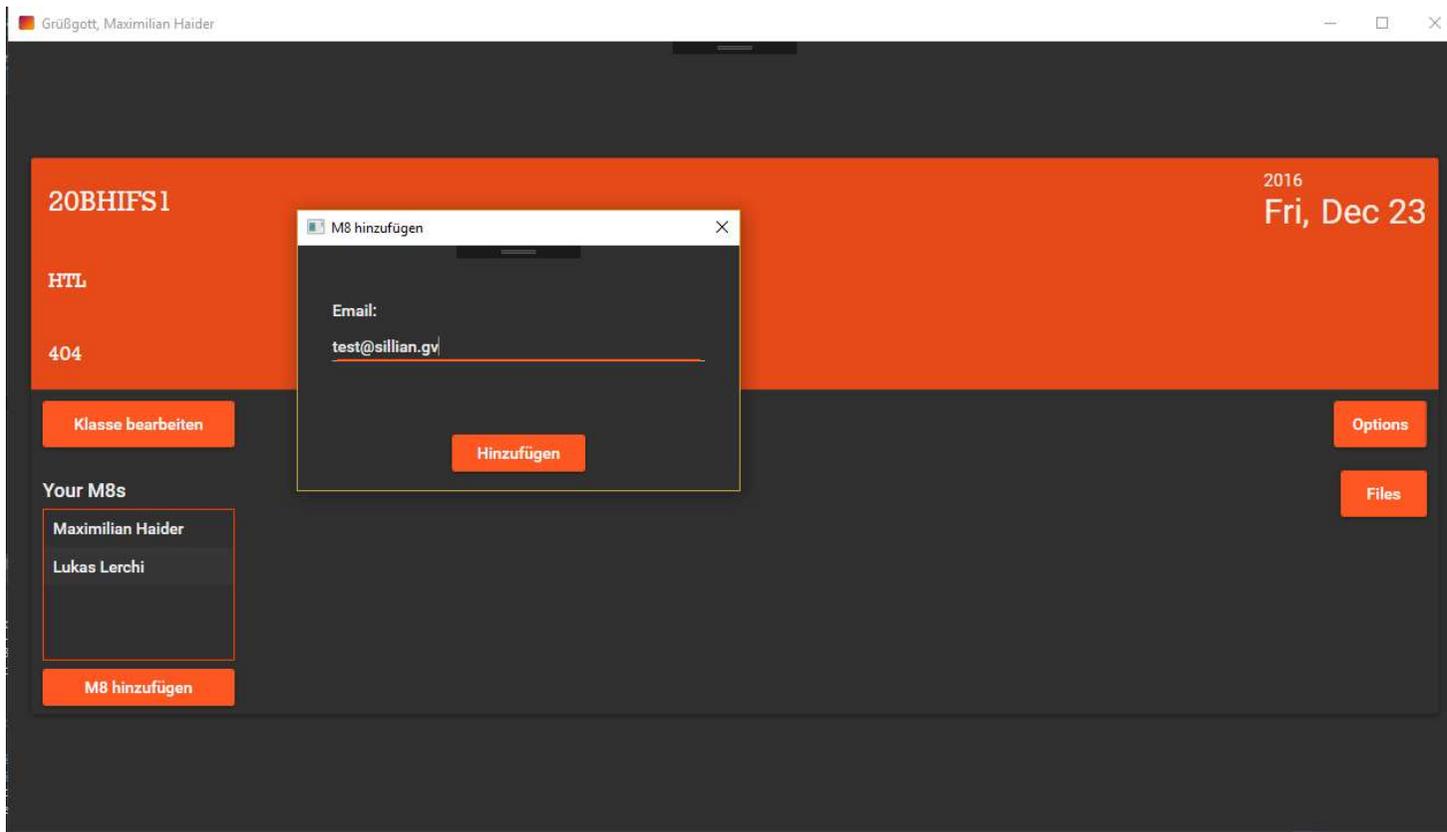
# Vote Button wird nach dem Wählen ausgeblendet



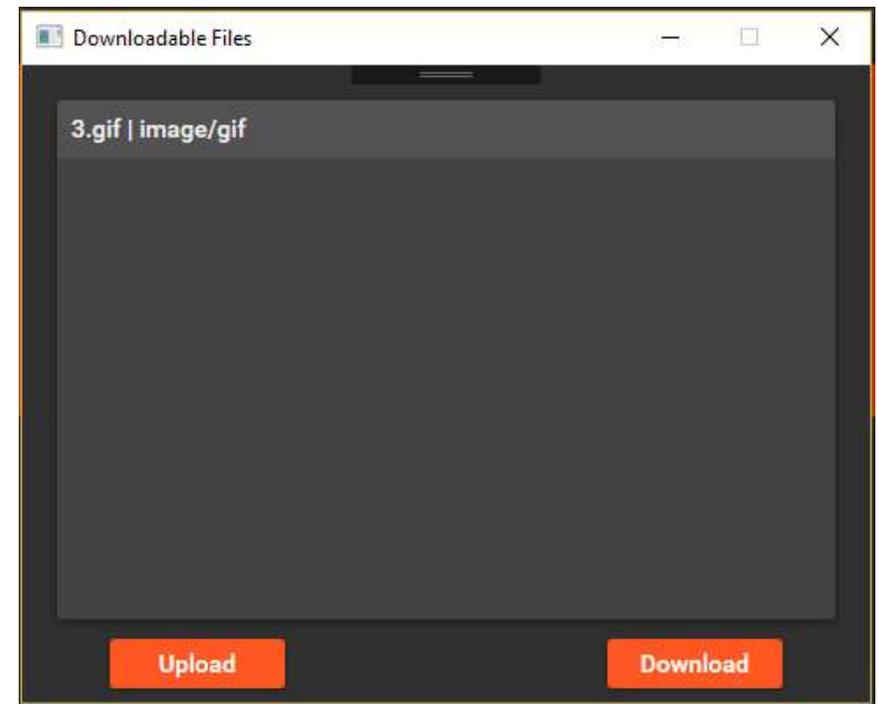
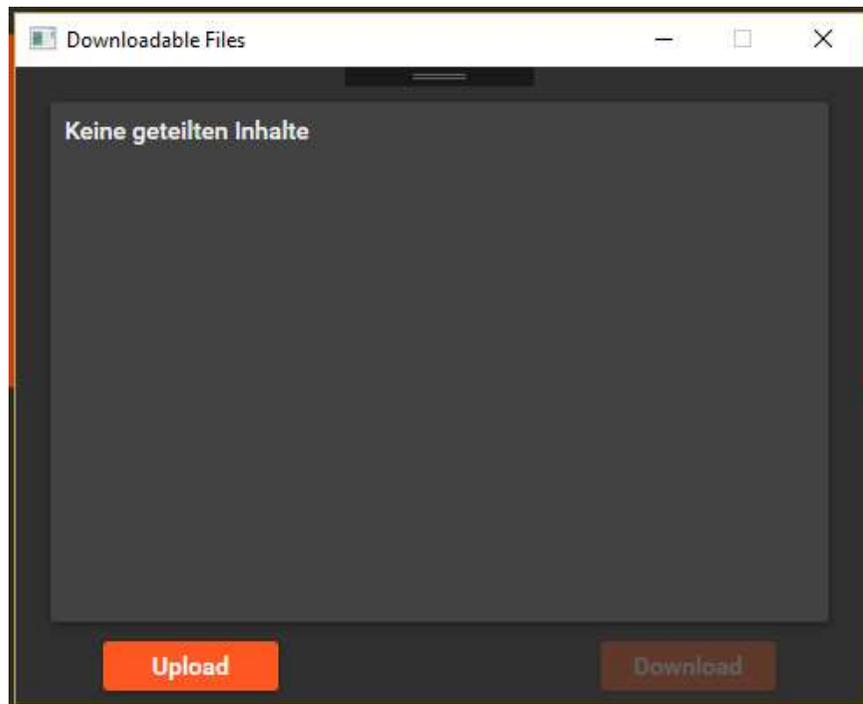
Nach dem Wählen werden der Klassensprecher und der Stellvertreter gesetzt.



# M8 zur Klasse hinzufügen



# Fileshare



# Fileshare - Upload

Current User: Maximilian Haider

The image shows a web application interface for file management. On the left, a sidebar contains navigation elements: '20BHIFS1', 'HTL', '404', 'Klasse bearbeiten', 'Your M8s', 'Lukas Lerchi', 'Maximilian Haider', and 'M8 hinzufügen'. The main content area is titled 'Downloadable Files' and displays a list of files: '3.gif | image/gif' and 'were-all-mad-here.png | image/png'. At the bottom of this list are 'Upload' and 'Download' buttons. A Windows File Explorer window is overlaid on the application, showing the path 'Dieser PC > DATA (D:) > Eigene Bilder > Wallpapers'. The file 'were-all-mad-here.png' is selected in the Explorer. A red arrow points from this file to the 'Upload' button in the web application. The user's name 'Grüßgott, Maximilian Haider' is visible in the top-left corner of the application window.

# Fileshare - Upload

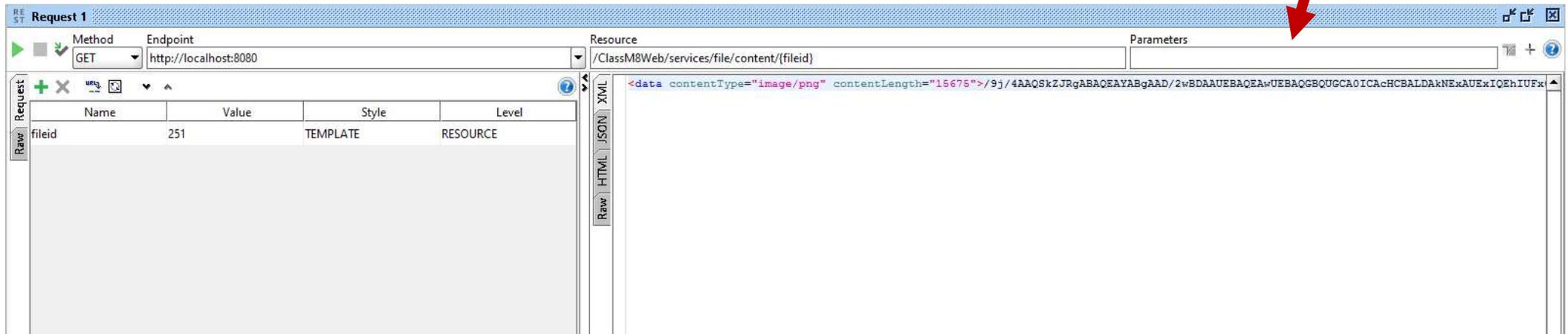
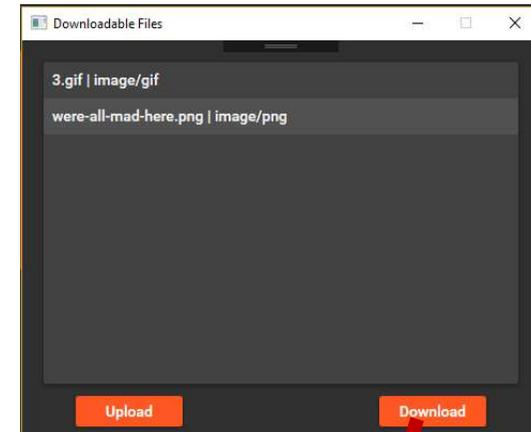
Current User: Lukas Lerchi

The screenshot shows a web application interface for file sharing. The browser title is "Grüßgott, Lukas Lerchi". The main content area has a dark background with orange accents. On the left, there's a sidebar with "20BHIFS 1", "HTL", "404", "Klasse bearbeiten", "Your M8s" (listing "Lukas Lerchi" and "Maximilian Haider"), and "M8 hinzufügen". On the right, there's a date "2016 Fri, Dec 23" and buttons for "Options", "Files", and "Vote". A central "Downloadable Files" window is open, showing a list of files: "3.gif | image/gif" and "were-all-mad-here.png | image/png". Below the list are "Upload" and "Download" buttons.

Beim File Upload werden zurzeit nur Metadaten wie Name und Size übertragen.

Wenn sich ein Bild auf dem Server befindet kann jedoch ein FileOutPutStream zurückgegeben werde.

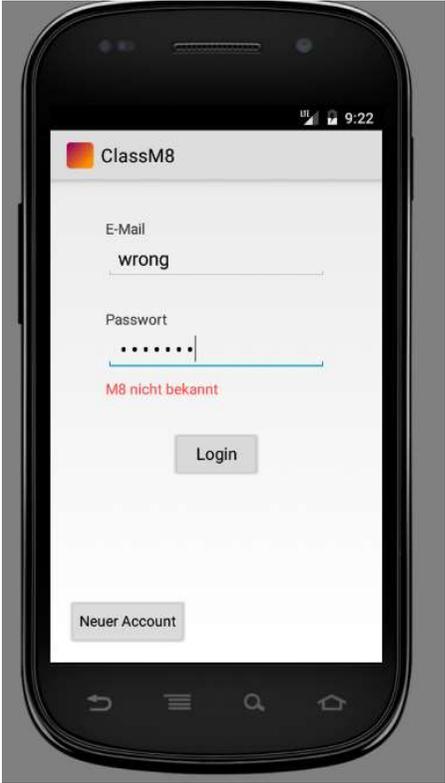
# Fileshare - Download



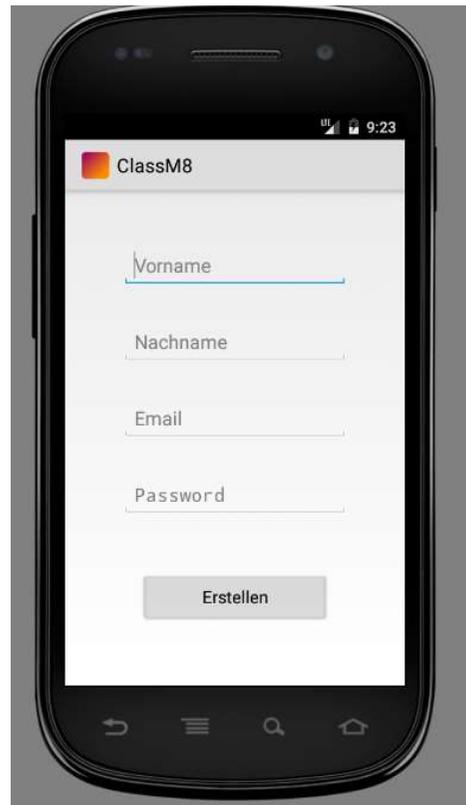
Da das Download Feature über das Frontend schwer zu demonstrieren ist und die Frontend Integration noch nicht zu 100 % geglückt ist, haben wir uns entschieden den Webservice zu präsentieren.

# Android- Frontend

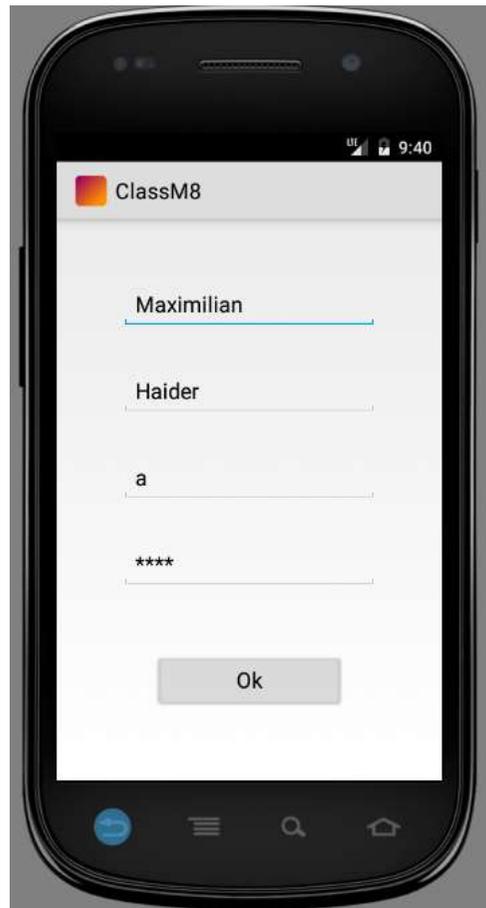
# Android Login Screen



# Benutzer Erstellen



# Benutzer Bearbeiten



# Klasse Bearbeiten



# Android Home



# Add M8 to Schoolclass



# Klassensprecherwahl



# Fileshare



# JUnit Tests (Bonus)

Output der JUnit Tests (Stacktraces, Logs usw) befindet sich in einer separaten JUnit PDF

# JUnit Tests für Schoolclass

Finished after 8,511 seconds

Runs: 16/16    Errors: 0    Failures: 0

edu.classm8web.junit.TestSchoolclass [Runner: JUnit 4] (7,928 s)

- test1A\_Insert (3,858 s)
- test2A\_Get (0,028 s)
- test2B\_GetFail (0,019 s)
- test3GetAll (0,383 s)
- test4A\_Update (0,244 s)
- test4B\_UpdateFail (0,132 s)
- test5A\_GetM8BySchoolclass (0,177 s)
- test5B\_GetM8BySchoolclassFail (0,137 s)
- test6A\_AddM8ToSchoolclass (0,559 s)
- test6B\_AddM8ToSchoolclassFail (0,290 s)
- test7A\_Election (0,623 s)
- test7B\_ElectionFail (0,013 s)
- test8A\_RemoveM8FromSchoolclass (0,002 s)
- test8B\_RemoveM8FromSchoolclassFail (0,001 s)
- test9A\_Delete (1,295 s)
- test9B\_DeleteFail (0,165 s)

Failure Trace

Writable    Smart Insert    117: 5

# JUnit Tests für Schoolclass (Beispielscreenshot)

The screenshot displays an IDE with a Java source file containing several JUnit test methods and a test run summary window.

```
118 @Test
119 public void test3GetAll(){
120     System.out.println("Get All Schoolclasses");
121     Response res = scr.getAllSchoolclasses(null, null);
122     SchoolclassResult result = (SchoolclassResult) res.getEntity();
123     assertTrue(result.isSuccess());
124 }
125
126 @Test
127 public void test4A_Update() {
128     sc.setName("updated");
129     Response res = scr.updateSchoolclass(null, null, sc.getId() + "", sc);
130     Result result = (Result) res.getEntity();
131     assertTrue(result.isSuccess());
132 }
133
134 @Test
135 public void test4B_UpdateFail() {
136     sc.setName("updated2");
137     Response res = scr.updateSchoolclass(null, null, -1 + "", null);
138     Result result = (Result) res.getEntity();
139     assertFalse(result.isSuccess());
140 }
141
142 @Test
143 public void test5A_GetM8BySchoolclass() {
144     System.out.println("Get M8s by Schoolclass");
145     Response res = ur.getBySchoolClass(null, null, sc.getId() + "");
146     Result result = (Result) res.getEntity();
147     assertTrue(result.isSuccess());
148 }
```

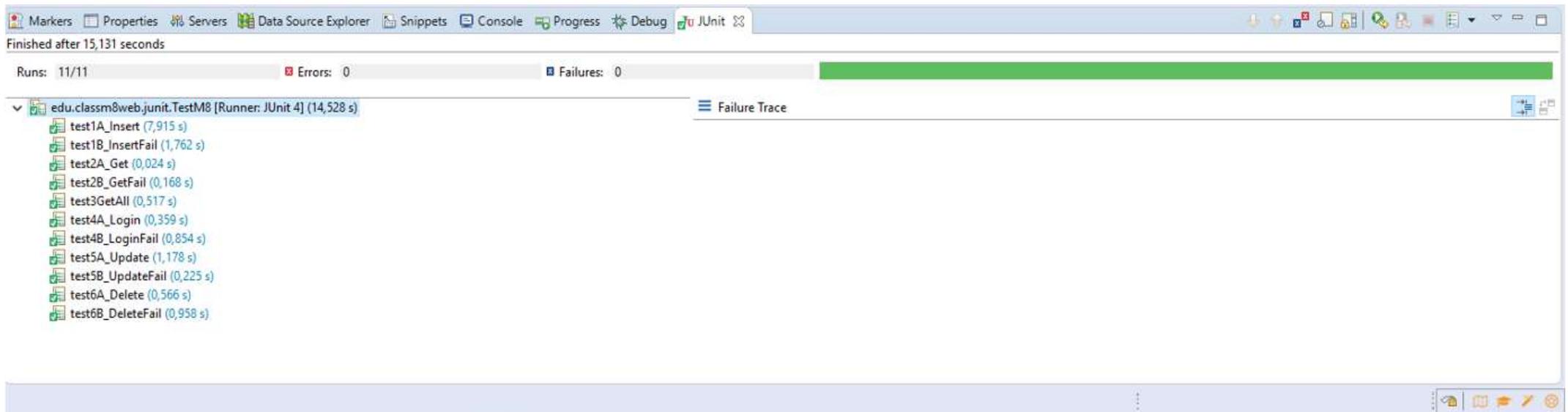
The test run summary window shows the following results:

- Finished after 8,511 seconds
- Runs: 16/16
- Errors: 0
- Failures: 0

The test run details are as follows:

- edu.classm8web.junit.TestSchoolclass [Runner: JUnit 4] (7,928 s)
- test1A\_Insert (3,858 s)
- test2A\_Get (0,028 s)
- test2B\_GetFail (0,019 s)
- test3GetAll (0,383 s)
- test4A\_Update (0,244 s)
- test4B\_UpdateFail (0,132 s)
- test5A\_GetM8BySchoolclass (0,177 s)
- test5B\_GetM8BySchoolclassFail (0,137 s)
- test6A\_AddM8ToSchoolclass (0,559 s)
- test6B\_AddM8ToSchoolclassFail (0,290 s)
- test7A\_Election (0,623 s)
- test7B\_ElectionFail (0,013 s)
- test8A\_RemoveM8FromSchoolclass (0,002 s)
- test8B\_RemoveM8FromSchoolclassFail (0,001 s)
- test9A\_Delete (1,295 s)
- test9B\_DeleteFail (0,165 s)

# JUnit Tests für M8s (User)



# JUnit Tests für M8s (Beispielscreenshot)

```
110
111 @Test
112 public void test2A_Get(){
113     Response res = ur.get(null, null, m8.getId() + "");
114     M8Result result = (M8Result) res.getEntity();
115     assertTrue(result.isSuccess());
116 }
117
118 @Test
119 public void test2B_GetFail(){
120     Response res = ur.get(null, null, "-1 + "");
121     M8Result result = (M8Result) res.getEntity();
122     assertFalse(result.isSuccess());
123 }
124
125 @Test
126 public void test3GetAll(){
127     Response res = ur.getUsers(null, null);
128     M8Result result = (M8Result) res.getEntity();
129     assertTrue(result.isSuccess());
130 }
131
132
133 @Test
134 public void test4A_Login() {
135     Response res = sr.login(m8);
136     Result result = (Result) res.getEntity();
137     assertTrue(result.isSuccess());
138 }
139
140 @Test
141 public void test4B_LoginFail() {
142     Response res = sr.login(new M8());
143     Result result = (Result) res.getEntity();
144     assertFalse(result.isSuccess());
145 }
```

edu.class0web.junit

- TestM8
  - ur: UserResource
  - sr: SecurityResource
  - m8: M8
  - m8dos: M8
  - PERSISTENCE\_UNIT: String
  - emf: EntityManagerFactory
  - em: EntityManager
  - setUpClass(): void
  - setUp(): void
  - tearDown(): void
  - tearDownClass(): void
  - test1A\_Insert(): void
  - test1B\_InsertFail(): void
  - test2A\_Get(): void
  - test2B\_GetFail(): void
  - test3GetAll(): void
  - test4A\_Login(): void
  - test4B\_LoginFail(): void
  - test5A\_Update(): void
  - test5B\_UpdateFail(): void
  - test6A\_Delete(): void
  - test6B\_DeleteFail(): void

Finished after 15,131 seconds

Runs: 11/11    Errors: 0    Failures: 0

edu.class0web.junit.TestM8 [Runner: JUnit 4] (14,528 s)

- test1A\_Insert (7,915 s)
- test1B\_InsertFail (1,762 s)
- test2A\_Get (0,024 s)
- test2B\_GetFail (0,168 s)
- test3GetAll (0,517 s)
- test4A\_Login (0,359 s)
- test4B\_LoginFail (0,854 s)
- test5A\_Update (1,178 s)
- test5B\_UpdateFail (0,225 s)
- test6A\_Delete (0,566 s)
- test6B\_DeleteFail (0,958 s)

JPA (Bonus)

# JPA – Kurze Einleitung

- Versprochener OR – Mapper Bonus
- Als Referenzimplementierung wurde EclipseLink 2.5 von der Full Plattform des Glassfish 4.1 verwendet, welche den Java EE JPA 2.1 Standard erfüllt
- Dies ermöglicht uns auch komplexe Objekte einfach in eine Datenbank zu persistieren, ohne aufwändige SQL Statements zu schreiben.
- Des weiteren übernimmt der Entity Manager bzw. die Entity Manager Factory das Verwalten von Transaktionen
- Auf den Screenshots kann leider nicht immer der ganze Code dargestellt werden -> Für nähere Informationen befindet sich die JPA Implementierung unter `edu.classm8web.database`

# Persistence.xml – Persistence Context

```
<?xml version="1.0" encoding="UTF-8"?>
<persistence version="2.1"
  xmlns="http://xmlns.jcp.org/xml/ns/persistence" xmlns:xsi="http://www.w3.org/2001/XMLSchema
  xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/persistence http://xmlns.jcp.org/xml/ns/pe
  <persistence-unit name="ClassM8Web" transaction-type="RESOURCE_LOCAL">
    <class>edu.classm8web.database.dto.M8</class>
    <class>edu.classm8web.database.dto.Schoolclass</class>
    <class>edu.classm8web.database.dto.File</class>
    <properties>
      <property name="javax.persistence.jdbc.url"
        value="jdbc:oracle:thin:@212.152.179.117:1521:ora11g" />
      <property name="javax.persistence.jdbc.user" value="d5b09" />
      <property name="javax.persistence.jdbc.password" value="d5b" />
      <property name="javax.persistence.jdbc.driver" value="oracle.jdbc.OracleDriver" />
      <property name="eclipselink.target-database" value="Oracle" />
    </properties>
  </persistence-unit>
</persistence>
```

# DTO (Data Transfer Object) – M8

```
@Entity
@Table(name = "Mate")
public class M8 implements Serializable{

    private static final long serialVersionUID = 7576437934172296816L;

    @Id
    @GeneratedValue(strategy=GenerationType.AUTO)
    @Column(name="mateid")
    private long id;

    private String firstname;

    private String lastname;

    @Column(unique = true)
    private String email;

    private String password;

    private boolean hasVoted;

    private int votes;

    @ManyToOne
    @JoinColumn(name="schoolclassid")
    private Schoolclass schoolclass;
```

# DTO (Data Transfer Object) – Schoolclass

```
@Entity
public class Schoolclass implements Serializable{

    /**
     *
     */
    private static final long serialVersionUID = -69655466883930376L;

    @Id
    @GeneratedValue(strategy=GenerationType.AUTO)
    @Column(name="schoolclassid")
    private long id;

    @OneToMany(mappedBy = "schoolclass")
    private List<M8> classMembers;

    @OneToMany(mappedBy = "referencedSchoolclass")
    private List<File> files;

    private String name;

    private String room;

    @OneToOne
    @JoinColumn(name="presidentid")
    private M8 president;

    @OneToOne
    @JoinColumn(name="presidentdeputyid")
    private M8 presidentDeputy;
```

# DAO (Data Access Object) - Interface

```
package edu.classm8web.database.dao.common;

import java.util.List;

import javax.persistence.EntityManager;

import edu.classm8web.exception.DatabaseException;

public interface BaseDBService<K, E> {
    void persist(E entity) throws DatabaseException;
    void removeById (K id) throws DatabaseException;
    void removeByEntity(E entity) throws DatabaseException;
    void update(E entity) throws DatabaseException;
    E findById(K id);
    List<E> findAll();
    void createPersistentComponents();
    void closeEntityManagerFactory();
    void closeEntityManager();
    void deleteAll() throws DatabaseException;
    EntityManager getEm();
}
```

# DAO (Data Access Object) - Schoolservice

```
public class SchoolclassService implements BaseDBService<Long, Schoolclass> {  
  
    private static SchoolclassService instance;  
  
    private EntityManager em;  
    private EntityManagerFactory emf;  
  
    private SchoolclassService() {}  
  
    public static SchoolclassService getInstance() {}  
  
    @Override  
    public void persist(Schoolclass entity) throws DatabaseException {  
        try {  
            em.getTransaction().begin();  
  
            em.persist(entity);  
  
            em.getTransaction().commit();  
        } catch (Exception e) {  
            throw new DatabaseException(e.getMessage());  
        }  
    }  
  
    public void removeById(Long id) throws DatabaseException {}  
  
    public void removeByEntity(Schoolclass entity) throws DatabaseException {}  
  
    public void update(Schoolclass entity) throws DatabaseException {}  
}
```