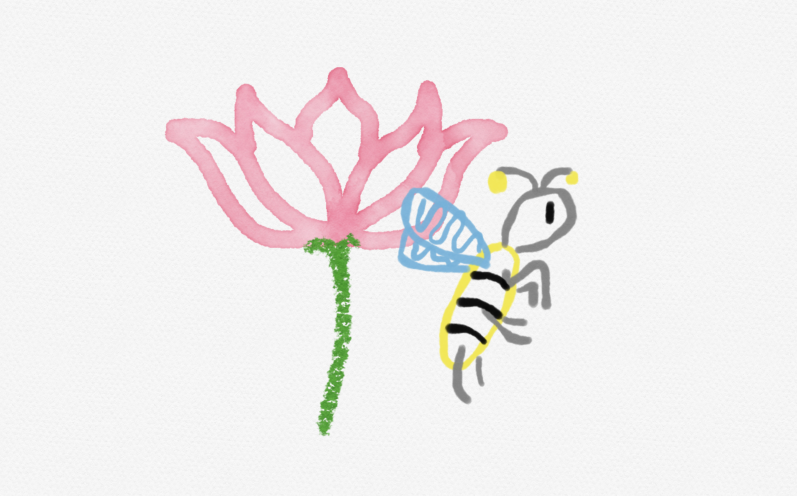
Bibliotheksanwendung - Implementierungsplan

Ivan Charviakou
León Liehr
Mohamad Najjar
Jonas Picker
Sergei Pravdin

1. Juni 2021
v1.1



Inhaltsverzeichnis

1 Einleitung

In diesem Dokument ist der Implementierungsplan der Webanwendung **BiBi** dokumentiert. Dabei erfolgt die Gliederung der Implementierung in Milestones, welche wiederum in Arbeitspakete aufgeteilt werden. Außerdem sind das **PERT-Diagramm**, die **Spezialgebiete**-Tabelle und die **Whitebox-Tests** zu sehen.

2 Meilensteine

Autor: Jonas Picker

Die Implementierungsphase wird drei Wochen dauern und in 3 gleiche Zeitabschnitte unterteilt, in denen wesentliche Funktionalitäten fertiggestellt werden sollen. Allgemein liegt der Schwerpunkt auf einer vertikalen, testbaren Realisierung einzelner Funktionen, jedoch müssen gewisse Kernmodule direkt im ersten Milestone abgearbeitet werden.

2.1 Meilenstein 1

Fertigstellungsdatum: 04.06.2021
Im Fokus des ersten Meilensteins liegen Grundfunktionalitäten, nach seinem Erreichen soll das System auf normalem Wege gestartet und heruntergefahren werden können. Hierzu werden zunächst die Datenbankinitiallisierung (mit Beispieldaten) und Anbindung sowie der Logger und die Systemkonfiguration als horizontale Funktionspakete erstellt. Geplante vertikale Module des ersten Abschnitts beinhalten den Login, eine paginierte Listenansicht und die auf allen Facelets sichtbaren Seiten- und Kopfleisten.

2.2 Meilenstein 2

Fertigstellungsdatum: 11.06.2021
Der Hauptteil der essentiellen Systemfunktionen sollen im zweiten Abschnitt als vertikale Komponenten ausgearbeitet werden. Die Mediumsansicht/erstellung und Suchfunktionalitäten für Medien stehen dabei im Vordergrund. Nach Erreichen des Milestones soll nach Kategorien gestöbert und neue Kategorien sollen erstellt werden können. Neben dem Nutzerprofil und der Registrierung werden außerdem Listenansichten für Mitarbeiter fertiggestellt werden.

2.3 Meilenstein 3

Fertigstellungsdatum: 18.06.2021
In der letzten Implementierungsphase soll das System hauptsächlich um Randfunktionen ergänzt werden. Als letzte essentieller Blöcke werden die Anwendungseinstellungen für Admins, die Suchfunktionalität für Nutzer und der Wartungsthread fertiggestellt. Außerdem wird die Zugriffskontrolle mittels einer PhaseListener-Implementierung realisiert. Mitarbeiter sollen eine Listenansicht mit Leihfristverstößen angezeigt bekommen können. Für Nutzer soll ebenfalls eine Liste mit ausgeliehenen/abzuholenden Exemplaren einsehbar sein. Kleinere Komfortfunktionen wie z.B. das Anzeigen von Suchvorschlägen folgen zuletzt.

3 Arbeitspakete

Die folgenden Tabellen dienen der Übersicht über die einzelnen Arbeitspakete, die zur Realisierung unseren System benötigt werden. Sie geben unter Anderem die Klassen, Methoden, und Facelets an, die verschiedenen Arbeitspaketen innerhalb von Milestones zugeordnet sind. Das Projekt wird in drei Meilensteilen aufgeteilt, die den drei Tabellen entsprechen. Als Konvention werden folgende Kürzel vereinbart: F steht für Facelet, C für Klasse, B für Managed-Bean, Co für Konverter, und V für Validatoren. Zusätzlich sind bei der Angabe einer Klassenname alle zugehörige Methoden zu implementieren.

3.1 Meilenstein 1

Autor: Mohamad Najjar

Im Folgenden werden die Arbeitspakete, die zum Erreichen des ersten Milestones notwendig sind, aufgelistet und kurz beschrieben.

Arbeitspaket	**ID**	**Inhalt**	**Abhängigkeiten**	**Entwickler**	**Dauer**
Basis	11	Logger (C) ConfigReader (C) ConnectionPool (C)	11	Ivan Charviakou	5
Utilities I	10	ApplicationDao (C) ApplicationDto (C) CSSReader (C) EmailUtility (C)	11	Jonas Picker	4
Utilities II	20	TokenGenerator (C) PasswordHashingModule (C) SystemStartStop (C)	11	Ivan Charviakou	4
Systemstart	30	DataLayerInitializer (C) DatenbankInitialisierung (SQL)	10	Jonas Picker	4
Facelets	40	Template (F) (BB) PaginatedList (F) Fehlerseiten (F) (BB)	10	León Liehr	4
Fehlerbehandlung	41	ErrorDto (FC) ExceptionHandler (C) CustomExceptionHandler (C) UserDto (C) UserSession (MB) RoleConverter (Co)	40	Sergei Pravdin	3
Nutzerdefinierung	52	PasswordValidator (C) ConfirmPasswordValidator (C) EmailValidator (C) UserDao (M) UserDao (M) Registration (BB) (F) Login (BB) (F)	30, 41	Mohamad Najjar	6
Nutzerathentifizierung	50	UserValidator (C) ConfirmPasswordValidator (C) EmailValidator (C) UserDao (M) UserDao (M)	52, 20	Mohamad Najjar	4
Globale Anwendungseinstellungen	61	Contact (BB) (F) SiteNotice (BB) (F) PrivacyPlocy (BB) (F)	30, 41	León Liehr	3
Administrative Funktionalitäten	60	Administration (BB) (F) LogoValidator (C)	30, 41	Sergei Pravdin	4

Tabelle 1: Tabelle des ersten Milestones

3.2 Meilenstein 2

Autor: Ivan Charviakou

Im Folgenden werden die Arbeitspakete zum zweiten Meilenstein angegeben.

Arbeitspaket	ID	Inhalt	Abh.	Entwickler	Dauer	Startzeit	Endzeit
Medienerstellung	250	Medienerstellung (F/B) MediumDao (C) - readGlobalAttributes() - createMedium(...)		M1	Ivan	5,00	7.6.21, 8:00	7.6.21, 13:00
Medienansicht	200	Medienansicht (F/B) MediumDao (C) - createCopy(...) - readAllCopiesFromMedium(...) - updateCopy(...) - deleteCopy(...) - readMedium(...) - updateMedium(...) - deleteMedium(...) - updateMediumAttributes(...)		M1	Sergei	14,00	7.6.21, 9:00	8.6.21, 16:00
Mediensuche	290	Mediensuche (F/B) MediumDao (C) - readMediaBySearchCriteria(...) AttributeOrCategoryConverter (Co) SearchOperatorConverter (Co)		M1	León	12,00	7.6.21, 10:00	9.6.21, 16:00
Medientransaktionen	210	Medienausleihe (F/B) Medienrückgabe (F/B) MediumDao (C) - lendCopy(...) - returnCopy(...)		M1	Jonas	5,00	7.6.21, 9:00	9.6.21, 14:00
Nutzerprofil	240	Profile (F/B) UserDao (C) - deleteUser(...)		M1	Mohamad	3,00	7.6.21, 10:00	7.6.21, 17:00

Passwort- und Emailvalidierung	230	Passwortzurücksetzung (F/B) E-Mail-Validation (F/B) Email-Utility (C) Token generator (C) UserDao (C) - updateUser(...)	240, M1	Ivan	8,00	8.6.21, 8:00	8.6.21, 16:00
Registrierung	220	Registration (F/B) EmailValidator (V) ConfirmPasswordValidator (V) PasswordValidator (V) UserValidator (V) UserDao (C) - createUser(...)	230, M1	Mohamad	8,00	8.6.21, 10:00	8.6.21, 18:00
Medienattribute	260	Mediumschemabearbeitung (F/B) MediumDao (C) - updateGlobalAttributes(...)	M1	Jonas	10,00	7.6.21, 9:00	9.6.21, 14:00
Kategorienerstellung	270	CategoryCreator (F/B) CategoryDao (C) - createCategory(...)	M1	Mohamad	4,00	7.6.21, 10:00	7.6.21, 17:00
Kategoriensuche	280	CategoryBrowser (F/B) CategoryDao (C) - readCategory(...) - readCategoriesByName(...) - updateCategory(...) - deleteCategory(...)	M1	León	8,00	7.6.21, 10:00	9.6.21, 16:00
Milenstein-Testing	295	Einzelne Tests für Meilenstein 2	200, 210, 220, 230, 240, 250, 260, 270, 280, 290, M1	Sergei	8,00	10.6.21, 9:00	10.6.21, 17:00

3.3 Meilenstein 3

Autor: León Liehr

Nachfolgend die Arbeitspakete des dritten Milestones.

Arbeitspaket	**ID**	**Inhalt**	**Abhängigkeiten**	**Entwickler**	**Dauer**
Medium Search	130	medium-search (F) MediumSearch (BB) searchMedium (M) addNuancedSearchField (M) MediumDao (C) readMediaBySearchCriteria (M)MediumSearchDto (C) AttributeOrCategoryConverter (Co) AvailabilityStatusConverter (Co) MediumPeviewPositionConverter (Co) SearchOperatorConverter (Co)	80, 81	León Liehr	8
Password Reset & Email Validation	150	password-reset (F) PasswordReset (BB) resetPassword (M) email-confirmation (F) EmailConfirmation (BB) confirmEmailAddress (M) UserDao (C)	120	Ivan Charviakou	6
Copies of a User	100	copies-ready-for-pickup (F) CopiesReadyForPickup (BB) borrowed-copies (F) BorrowedCopies (BB) MediumDao (C)	80, 120	Sergei Pravdin	5
Copies by User	91	copies-ready-for-pickup-all-users (F) CopiesReadyForPickupAllUsers (BB) lending-period-violations (F) LendingPeriodViolations (BB) MediumDao (C)	80, 120	Jonas Picker	5
User Search	140	user-search (F) UserSearch (BB) UserSearchDto (C) UserDao (C)	120	Mohamad Najjar	6
Enhancements	160	reactiveInputField (CC)	130, 140, 110	León Liehr	3

Tabelle 3: Tabelle des drtten Milestones

4 PERT-Diagramm

Autor: Ivan Charviakou

Das PERT Diagramm zum Projekt wird im PDF Anhang dargestellt. Insbesondere bildet es die Abhängigkeiten zwischen den Paketen ab und zeigt das kritische Pfad im Projekt auf. Dabei werden die Zeitdauer in Stunden berechnet und die genauen Start- und Stoppzeiten sind den Milensteintabellen zu entnehmen.

5 Spezialgebiete

Autor: Jonas Picker

Die Vielzahl der verwendeten Technologien erfordert eine Spezialisierung der einzelnen Teammitglieder. Die jeweiligen Spezialgebiete sind unten aufgelistet.

git	León Liehr
JSF Internationalisierung	León Liehr
JSF Templates	León Liehr
JSF Components	León Liehr
Listenabstraktion	León Liehr
Suchfunktionalitäten	León Liehr
LaTeX	Ivan Charviakou
RegEx	Ivan Charviakou
Jakarta Mail	Ivan Charviakou
JSF Validators	Ivan Charviakou
Design-Patterns	Ivan Charviakou
JSF Converters	Mohamad Najjar
RSA	Mohamad Najjar
CSS/Bootstrap	Mohamad Najjar
Nutzerauthentifizierung	Mohamad Najjar
SQL	Jonas Picker
Logging	Jonas Picker
JSF/CDI Scopes	Jonas Picker
SSL/TLS	Jonas Picker
Systemkonfiguration	Jonas Picker
Selenium	Sergei Pravdin
JSF File Upload	Sergei Pravdin
JUnit	Sergei Pravdin

6 Whitebox-Tests

Autor: Sergei Pravdin

Am Ende jedes Milestones werden Whitebox-Tests von dem Verantwortlicher (Sergei Pravdin) geschrieben. Jedes Arbeitspaket hat mindestens einen Test, um seine Funktionalitäten zu prüfen. Die Arbeitspakete werden durch die Verbindung mit der Datenbank getestet, wenn der Zustand der Anwendung es ermöglicht. Sonst werden fehlende Bestandteile von Mockito oder PowerMock gemockt. Die Tests werden mit dem Projekt mitgeliefert. Vor dem Projektabgabe werden die allen erzeugenden beim Testing Objekte aus der Datenbank manuell gelöscht.

6.1 PreTests

Damit weitere Tests durchgeführt werden können, werden folgende PreTests zuerst definiert.

6.1.1 Arbeitspaket: Systemstart. DataLayerInitializer

Um eine Datenbankverbindung zu bekommen, wird eine Methode execute() aufgerufen.

6.1.2 Arbeitspaket: Systemstart. Erstellung der Test-Objekten - initObjects()

Um Tests von den Milestones durchzuführen, werden wir folgende Objekte in die Datenbank durch SQL-Anfragen abgespeichert: ein Nutzer (ID: 1, E-Mail-Adresse: preTestSep21email@gmail.com, Password: testPassword1), ein Medium (ID: 1, Name: "testMedium", Copy: testCopy, Attribute: testAttribute), ein Copy (ID: 2, Name: "testCopy"), ein Copy (ID: 30, Name: "testCopyBorrowed"), ein Attribute (ID: 3, Name: "testAttribute"), eine Kategorie (ID: 20, Name: "testCategory"). Ein Exemplar mit einer ID '30' wird als 'BORROWED' von einem testUser mit einer ID '1' gekennzeichnet.

6.2 Milestone 1

6.2.1 Arbeitspaket: Basis.

Der Logger wird getestet. Für den Test wird eine Message ins Log-File durch die Methode development("testMessage") geschrieben. Der Test ist bestanden, wenn eine erste Message im Log-File "testMessage" ist.

6.2.2 Arbeitspaket: Utilities I.

Der Test ist für ein ApplicationDao zuständig. Ein ApplicationDto 'testDto' wird erstellt und durch testDto.setName("testName") ausgefüllt. Danach wird eine Methode ApplicationDao.createCustomization(testDto) aufgerufen. Der Test ist bestanden, wenn eine Methode ApplicationDao.readCustomization(testDto).getName() "testName" ist.

6.2.3 Arbeitspaket: Login bzw. Nutzerauthentifizierung. Test: LogIn (erfolglos).

Eine Methode setEmail("wrongEmail") und eine Methode setPassword("wrongPassword") aus der Klasse 'Login' werden aufgerufen. Im nächsten Schritt wird eine Methode logIn() aufgerufen. Das Ergebnis muss "null" sein, damit der Test als bestanden gilt.

6.2.4 Arbeitspaket: Login bzw. Nutzerauthentifizierung. Test: LogIn (erfolgreich).

Eine Methode setEmail("preTestSep21email@gmail.com") und eine Methode setPassword("testPassword1") aus der Klasse Login werden aufgerufen. Im nächsten Schritt wird eine Methode logIn() aufgerufen. Das Ergebnis muss "profile?id=1" sein, damit der Test als bestanden gilt.

6.2.5 Arbeitspaket: Facelets.

Eine Methode logOut() von einem Footer wird aufgerufen. Der Test ist bestanden, wenn das Ergebnis "logIn.xhtml?faces-redirect=true" ist.

6.2.6 Arbeitspaket: Medien aller Nutzer bzw. Listendemo. Test: Abzuholende Exemplare aller Nutzer.

Die Methode loadCopies() wird aus der Klasse 'BorrowedCopies' aufgerufen. Der Test ist bestanden, wenn ein Ergebnis einer Methode 'getCopies()' ein Exemplar mit der ID '30' beinhaltet.

6.3 Milestone 2

6.3.1 Arbeitspaket: Medienerstellung.

Ein UserDto 'userDto' wird zuerst erzeugt, dann wird die Methode setEmailAddress("preTestSep21email@gmail.com") aufgerufen und andere Attribute von userDto werden durch userDto = UserDao.readUserByEmail(userDto) ausgefüllt. Im nächsten Schritt wird die Methode setUser(userDto) aus der Klasse 'MediumCreator' aufgerufen. Danach werden ein mediumDto mit der ID '4', dem Name 'testMedium2' und ein CopyDto mit der ID '5' erstellt. Im nächsten Schritt werden die Methoden setMedium und setCopy in der Klasse 'MediumCreator' aufgerufen. Schließlich wird die Methode createMediumAndFirstCopy() aufgerufen. Der Test ist bestanden, wenn die Methode MediumDao.readMedium(mediumDto).getId() das Ergebnis '4' liefert.

6.3.2 Arbeitspaket: Medienansicht. Test: Ausleihe eines Exemplars durch einen Nutzer.

Ein UserDto 'userDto' wird zuerst erzeugt, dann wird die Methode setEmailAddress("preTestSep21email@gmail.com") aufgerufen und andere Attribute von userDto werden durch userDto = UserDao.readUserByEmail(userDto) ausgefüllt. Im nächsten Schritt wird eine Methode setID(4) von einem mediumDto aufgerufen und eine BB bekommt ein ausgefülltes

mediumDto durch MediumDao.readMedium(mediumDto) zurück. Schließlich wird die testende Methode pickUpCopy(4, userDto) aufgerufen. Der Test ist bestanden, wenn die Methode MediumDao.readMedium(mediumDto).getCopy(4).getCopyStatus() das Ergebnis 'READY FOR PICKUP' liefert.

6.3.3 Arbeitspaket: Mediensuche.

Ein MediumSearchDto 'mediumSearch' wird erzeugt und durch mediumSearch.setGeneralSearchTerm("testMedium") ausgefüllt. Im nächsten Schritt wird eine Methode setMediumSearch(mediumSearch) in der Klasse 'MediumSearch' aufgerufen. Schließlich wird eine testende Methode searchMedium() aufgerufen. Der Test ist bestanden, wenn eine Methode getItems() in der Klasse 'MediumSearch' ein mediumDto mit einem Attribut 'testMedium' liefert.

6.3.4 Arbeitspaket: Medientransaktionen. Test: Direktausleihe eines Exemplars durch einen Mitarbeiter

Ein UserDto 'userDto' wird zuerst erzeugt, dann wird die Methode setEmailAddress("preTestSep21email@gmail.com") aufgerufen und andere Attribute von userDto werden durch userDto = UserDao.readUserByEmail(userDto) ausgefüllt. Im nächsten Schritt wird die Methode setUser(userDto) aus der Klasse 'DirectLending' aufgerufen. Dann werden ein copyDto und ein mediumDto erzeugt. Sie werden durch copyDto.setId(5) und mediumDto.setId(4) definiert. Dann wird ein copyDto durch eine Methode MediumDao.readMedium(mediumDto).getCopies(5) ausgefüllt und durch eine Methode copies.put(copyDto) in der Klasse 'DirectLending' hinzugefügt. Schließlich wird eine testende Methode lendCopy(5, userDto) aufgerufen. Der Test ist bestanden, wenn die Methode MediumDao.readMedium(mediumDto).getCopy(5).getCopyStatus() das Ergebnis 'BORROWED' liefert.

6.3.5 Arbeitspaket: Nutzerprofil.

Ein UserDto 'userDto' wird zuerst erzeugt, dann wird die Methode userDto.setEmailAddress("preTestSep21email@gmail.com") aufgerufen und andere Attribute von userDto werden durch userDto = UserDao.readUserByEmail(userDto) ausgefüllt. Im nächsten Schritt wird die Methode setUser(userDto) aus der Klasse 'Profile' aufgerufen. Dann wird sich eine Hausnummer durch userDto.setStreetNumber("13a") geändert. Abschießend wird die Methode 'save()' aufgerufen. Jetzt muss die neue Hausnummer in der Datenbank durch das userDto von der Klasse 'Profile' aktualisiert. Der Test ist bestanden, wenn die Methode UserDao.readUserByEmail(userDto).getStreetNumber() das Ergebnis "13a" liefert.

6.3.6 Arbeitspaket: Passwort- und Emailvalidierung. Test: Zurücksetzung eines Passwords.

Ein UserDto 'userDto' wird zuerst erzeugt, dann wird die Methode setEmailAddress("preTestSep21email@gmail.com") aufgerufen und andere Attribute von userDto werden durch userDto = UserDao.readUserByEmail(userDto) ausgefüllt. Im nächsten Schritt wird die Methode setUser(userDto) aus der Klasse 'PasswordReset' aufgerufen. Die Attribute einer Klasse 'PasswordReset' werden durch setPassword("newPassword") und setConfirmedPassword("newPassword") ausgefüllt. Ein TokenDto wird von einem TokenGenerator

erstellt und durch setToket(tokenDto) hinzugewiesen. Dann wird eine testende Methode resetPassword(userDto) aufgerufen. Um zu testen, werden eine Methode setEmail("preTestSep21email@gmail.com") und eine Methode setPassword(newPassword) aus der Klasse 'Login' aufgerufen. Im nächsten Schritt wird eine Methode logIn() aufgerufen. Das Ergebnis muss "profile?id=1" sein, damit der Test als bestanden gilt (Erfolgreiche Anmeldung mit einem neuen Password).

6.3.7 Arbeitspaket: Registrierung.

Ein UserDto 'userDto' wird erzeugt. Dann werden eine Methode userDto.setId(6), userDto.setPassword("testPassword") und userDto.setEmailAddress("testSep21email@gmail.com") aufgerufen. Im nächsten Schritt wird die Methode 'register()' in der Klasse 'Registration' aufgerufen. Diese Methode muss eine Interaktion mit einem UserDao, genauer gesagt mit der Methode 'UserDao.createUser(userDto)', implementieren, deshalb ist der Test bestanden, wenn das Ergebnis der Methode 'UserDao.readUserByEmail(userDto).getId()' "6" ist (Das bedeutet, das ein neuer Nutzer mit der Id "6' in der Datenbank erfolgreich abgespeichert ist).

6.3.8 Arbeitspaket: Medienattributen.

Ein Map 'attributes' wird gemockt. Ein neues AttributeDto 'testDto' wird erzeugt und eine Methode testDto.setId(7) aufgerufen. Danach wird ein 'testDto' durch attributes.put(7, testDto) hinzugefügt. Im nächsten Schritt wird eine Methode setAttributes(attributes) aus der Klasse 'MediumSchemaEditor' aufgerufen. Schließlich wird eine testende Methode deleteAttribute(1) aufgerufen. Der Test ist bestanden, wenn die Methode aus der Klasse 'MediumSchemaEditor' getAttributes.get(7) 'null' liefert.

6.3.9 Arbeitspaket: Kategorienerstellung.

Zuerst wird ein categoryDto erzeugt und durch categoryDto.setId(8) und categoryDto.setName("testCategory2") ausgefüllt. Dann wird eine Methode setCategory(categoryDto) in der Klasse 'CategoryCreator' aufgerufen. Schließlich wird eine testende Methode createCategory() aufgerufen. Der Test ist bestanden, wenn die Methode CategoryDao.readCategory(categoryDto).getName() das Ergebnis "testCategory2" liefert.

6.3.10 Arbeitspaket: Kategoriensuche.

Zuerst wird ein categorySearchDto erzeugt und durch categorySearchDto.setSearchTerm(20) ausgefüllt. Dann wird eine Methode searchCategory() in der Klasse 'CategoryBrowser' aufgerufen. Der Test ist bestanden, wenn die Methode getCurrentCategory().getID() das Ergebnis "20" liefert.

6.4 Milestone 3

6.4.1 Arbeitspaket: Medien pro Nutzer.

Ein UserDto 'userDto' wird zuerst erzeugt, dann wird die Methode setEmailAddress("preTestSep21email@gmail.com") aufgerufen und andere Attribute von userDto werden durch userDto = UserDao.readUserByEmail(userDto) ausgefüllt. Im nächsten Schritt wird

die Methode getCopies(userDto) aus der Klasse 'BorrowedCopies' aufgerufen. Der Test ist bestanden, wenn ein Ergebnisliste einer Methode getCopies(userDto) ein Copy mit der Id '30' beinhaltet.

6.4.2 Arbeitspaket: Nutzersuche.

Ein userSearchDto wird erstellt und eine Methode userSearchDto.setSearchTerm("preTestSep21email@gmail.com") aufgerufen. Danach wird eine Methode setUserSearchDto(userSearchDto) aus der Klasse 'UserSearch' aufgerufen. Im nächsten Schritt wird eine testende Methode searchUser() aufgerufen und eine Methode getItems() aus der Klasse 'UserSearch' muss im Ergebnis ein userDto mit der Id '1' beinhalten, damit der Test bestanden ist.

6.4.3 Arbeitspaket: Globale Anwendungseinstellungen.

Ein ApplicationDto 'testDto' wird erzeugt. Dann wird die Methode setApplication(testDto) aus der Klasse 'Contact' aufgerufen. Im nächsten Schritt werden die Methode testDto.setCity("testCity") und save() aus der Klasse 'Contact' aufgerufen. Der Test gilt als bestanden, wenn die Methode getApplication().getCity() aus der Klasse 'Contact' das Ergebnis "testCity" liefert.

6.4.4 Arbeitspaket: Fehlerbehandlung.

Wir wollen eine Fehlerseite testen, deshalb wird ErrorDto 'testErrorDto' erzeugt und durch setMessage("testErrorMessage") ausgefüllt. Dann wird eine Methode setErrorDto(testErrorDto) aufgerufen. Der Test ist bestanden, wenn das Ergebnis der Methode getMessage() von der Klasse 'Error' "testErrorMessage" ist.

6.4.5 Arbeitspaket: Leifristverstöße.

Zuerst wird ein MediumDto 'mediumDto' erstellt und durch mediumDto.setId(1) identifiziert. Dann wird ein mediumDto durch mediumDto = MediumDao.readMedium(mediumDto) ausgefüllt. Jetzt wird ein CopyDto copyDto erzeugt und durch copyDto = mediumDto.getCopy(30) ausgefüllt. Jetzt wollen wir einen neuen Abgabefrist durch copyDto.setDeadline("Derzeit + 10 Sekunden") definieren und diese Änderung in der Datenbank durch MediumDao.update(mediumDto) speichern. Danach wird eine Pause für 11 Sekunden im Test durchgeführt. Jetzt sind wir bereit eine Klasse 'LendingPeriodViolation' zu testen. Der Test ist bestanden, wenn sich ein copyDto mit der Id '30' im Ergebnis der Methode getItems() der Klasse 'LendingPeriodViolation' befindet.