

Ambroise BISIAUX
Yony COLLARD
Louis BOLTZ
Louis NAVREZ
Alexandre FOURNIER
Matteo CIPOLAT



Rapport de test

Structure et exécution	2
UserModelTest.php	3
ProofModelTest.php	5
AbsenceMonitoringModelTest.php	6

Structure et exécution

Structure des Tests :

Le dossier Tests est à la racine du projet (accessible sur github)

```
Tests/
├── bootstrap.php          # Initialisation de l'environnement de test
├── TestCase.php          # Classe de base avec gestion des transactions
├── Fixtures/
│   ├── UsersFixture.php  # Création d'utilisateurs de test
│   ├── AbsencesFixture.php # Création d'absences de test
│   ├── ProofsFixture.php  # Création de justificatifs de test
│   └── FixtureHelper.php  # Utilitaires pour les fixtures
├── Unit/
│   ├── Model/
│   │   ├── UserModelTest.php      # Tests du modèle utilisateur
│   │   ├── ProofModelTest.php     # Tests du modèle justificatif
│   │   └── AbsenceMonitoringModelTest.php # Tests surveillance absences
│   └── Security/
│       ├── SQLInjectionTest.php    # Tests injection SQL
│       └── README.md               # Documentation sécurité
└── SQL/
    ├── 00-setup.sql      # Configuration initiale
    ├── 01-schema-tests.sql # Tests de schéma
    ├── 02-data-integrity-tests.sql
    ├── 03-insert-tests.sql
    └── 04-cascade-tests.sql
```

Exécuter tous les tests : `php vendor/bin/phpunit`

Exécuter un fichier de test spécifique : Tests du modèle utilisateur :
`php vendor/bin/phpunit Tests\Unit\Model\UserModelTest.php`

UserModelTest.php

Cette classe teste le système de gestion des comptes utilisateurs (étudiants, enseignants, administrateurs) donc la gestion complète des comptes utilisateurs : inscription, connexion, modification de profil, gestion des droits d'accès selon le rôle (étudiant, enseignant, administrateur).

Fonctionnalités testées :

Création d'utilisateurs (createUser)

- Création de comptes avec validation
- Hashage des mots de passe
- Validation de l'unicité (email, identifier)

Authentification

- Connexion avec email/mot de passe
- Vérification du mot de passe hashé
- Gestion des échecs de connexion

Récupération d'utilisateurs

- Par ID (getUserById)
- Par email (getUserByEmail)
- Par identifiant (getUserByIdentifiant)
- Liste complète (getAllUsers)

Mise à jour de profils (updateUser)

- Modification des informations personnelles
- Changement de rôle
- Mise à jour de mot de passe

Suppression d'utilisateurs (deleteUser)

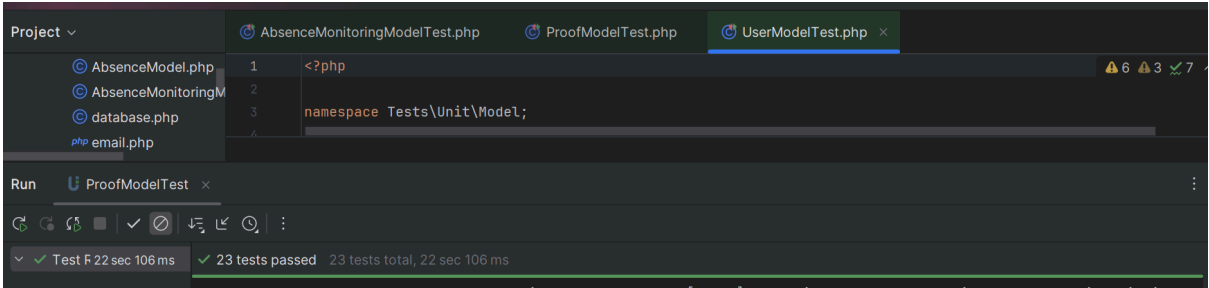
- Suppression logique ou physique
- Gestion des contraintes de clés étrangères

Gestion des rôles

- Attribution de rôles (student, teacher, admin)
- Vérification des permissions

Méthode	Description	Tests associés
<code>getUserById(\$id)</code>	Récupère un utilisateur par son ID	<code>testGetUserByIdReturnsCorrectUser</code> , <code>testGetUserByIdReturnsNullForNonexistentUser</code>
<code>getUserByIdentifier(\$identifiant)</code>	Récupère un utilisateur par son identifiant	<code>testGetUserByIdentifierReturnsCorrectUser</code> , <code>testGetUserByIdentifierIsCaseInsensitive</code> , <code>testGetUserByIdentifierReturnsNullForNonexistentIdentifier</code>
<code>verifyPassword(\$userId, \$password)</code>	Vérifie le mot de passe d'un utilisateur	<code>testVerifyPasswordReturnsTrueForCorrectPassword</code> , <code>testVerifyPasswordReturnsFalseForIncorrectPassword</code> , <code>testVerifyPasswordReturnsFalseForNonexistentUser</code>
<code>updatePassword(\$userId, \$hash)</code>	Met à jour le mot de passe	<code>testUpdatePasswordChangesUserPassword</code>
<code>updateEmail(\$userId, \$email)</code>	Met à jour l'email	<code>testUpdateEmailChangesUserEmail</code>
<code>isEmailTaken(\$email, \$excludeId)</code>	Vérifie si un email est déjà utilisé	<code>testIsEmailTakenReturnsTrueForExistingEmail</code> , <code>testIsEmailTakenReturnsFalseForAvailableEmail</code> , <code>testIsEmailTakenExcludesCurrentUser</code> , <code>testIsEmailTakenIsCaseInsensitive</code>
<code>getRoleLabel(\$role)</code>	Traduit le rôle en français	<code>testGetRoleLabelReturnsCorrectFrenchTranslation</code> , <code>testGetRoleLabelReturnsOriginalValueForUnknownRole</code>
<code>getUserStatistics(\$identifiant)</code>	Statistiques d'absences utilisateur	<code>testGetUserStatisticsReturnsAbsenceCounts</code> ,

		testGetUserStatisticsReturnsZeroForUserWithNoAbsences
--	--	---



ProofModelTest.php

Cette classe teste le système de gestion des justificatifs fournis par les étudiants pour justifier leurs absences. Par exemple, un étudiant absent télécharge un certificat médical pour justifier son absence. Le système stocke le document, et un responsable peut ensuite l'approuver ou le rejeter. La classe teste ces étapes.

Fonctionnalités testées :

Téléversement de justificatifs (uploadProof)

- Upload de fichiers (PDF, images)
- Validation du type de fichier
- Stockage dans le système de fichiers
- Enregistrement en base de données

Récupération des justificatifs

- Par étudiant (getProofsByStudent)
- Par ID spécifique (getProofById)
- Liste complète pour l'administration

Validation des justificatifs (validateProof)

- Approbation par un responsable
- Rejet avec commentaire
- Mise à jour du statut

Suppression de justificatifs (deleteProof)

- Suppression du fichier physique
- Suppression de l'enregistrement en base

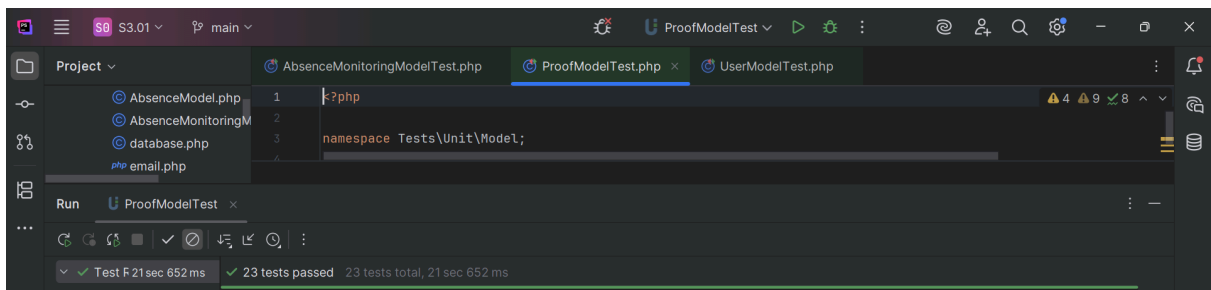
Vérification de la couverture

- Vérifie si une période d'absence est couverte par un justificatif valide

Méthode	Description	Tests associés
<code>getProofDetails(\$id)</code>	Récupère les détails d'un justificatif	<code>testGetProofDetailsReturnsCorrectData,</code> <code>testGetProofDetailsReturnsNullForNonexistentProof</code>

<code>extractProofFiles()</code>	Extrait les fichiers d'un justificatif (JSONB)	<code>testExtractProofFilesFromJsonb,</code> <code>testExtractProofFilesFallbackToFilePath</code>
<code>updateProofStatus(\$id, \$status)</code>	Met à jour le statut	<code>testUpdateProofStatusChangesStatus,</code> <code>testUpdateProofStatusWithInvalidStatusReturnsFalse</code>
<code>setRejectionReason(\$id, \$reason, \$comment, \$userId)</code>	Rejette un justificatif	<code>testSetRejectionReasonUpdatesProofAndCreatesHistory,</code> <code>testSetRejectionReasonWithoutUserIdUsesSystemDefault</code>
<code>setValidationReason(\$id, \$reason, \$comment, \$userId)</code>	Valide un justificatif	<code>testSetValidationReasonAcceptsProofAndJustifiesAbsences</code>
<code>setRequestInfo(\$id, \$comment, \$userId)</code>	Demande d'informations	<code>testSetRequestInfoSetsUnderReviewStatus</code>
<code>verrouiller(\$id)</code>	Verrouille un justificatif	<code>testVerrouillerLocksProof</code>
<code>deverrouiller(\$id)</code>	Déverrouille un justificatif	<code>testDeverrouillerUnlocksProof</code>
<code>isLocked(\$id)</code>	Vérifie si verrouillé	<code>testIsLockedReturnsCorrectStatus</code>
<code>updateAbsencesForProof()</code>	Met à jour les absences liées	<code>testUpdateAbsencesForProofSetsJustifiedTrue,</code> <code>testUpdateAbsencesForProofSetsJustifiedFalse</code>
<code>splitProofMultiple()</code>	Divise un justificatif en plusieurs	<code>testSplitProofMultipleCreatesSeparateProofs,</code> <code>testSplitProofMultipleReassignsAbsencesCorrectly,</code> <code>testSplitProofMultipleRollsBackOnFailure</code>

<code>translate(\$type, \$key)</code>	Traduit les raisons/statuts	<code>testTranslateReasonReturnsCorrectFrenchTranslation,</code> <code>testTranslateStatusReturnsCorrectFrenchTranslation</code>
<code>getRecentProofs(\$limit)</code>	Justificatifs récents	<code>testGetRecentProofsReturnsLimitedResults</code>
<code>getAllProofs(\$filters)</code>	Tous les justificatifs avec filtres	<code>testGetAllProofsFiltersByStatus,</code> <code>testGetAllProofsFiltersByDateRange</code>



AbsenceMonitoringModelTest.php

Cette classe teste le système automatique qui surveille les absences et alerte quand un étudiant a été absent plusieurs jours de suite, puis détecte automatiquement son retour.

Fonctionnalités testées :

Détection des étudiants en absence prolongée (`getStudentsWithOngoingAbsences`)

- Récupère les étudiants avec des absences non justifiées dans les 7 derniers jours
- Exclut les absences justifiées
- Exclut les absences trop anciennes (> 7 jours)

Détection du retour en cours (`hasStudentReturnedToClass`)

- Vérifie si un étudiant est revenu après une période d'absence
- Prend en compte l'heure (après 17h00)
- Ignore les weekends dans le calcul

Enregistrement du retour (`recordStudentReturn`)

- Enregistre dans la base de données qu'un étudiant est revenu
- Utilise un système UPSERT (INSERT ... ON CONFLICT UPDATE)
- Empêche les doublons

Calcul des demi-journées d'absence (`calculateHalfDays`)

- Compte les demi-journées (matin/après-midi)
- Matin : avant 12h30
- Après-midi : après 12h30
- Agrège par jour pour éviter les doublons

Méthode	Description	Tests associés
<code>getStudentsWithOngoingAbsences()</code>	Étudiants avec absences en cours	<code>testGetStudentsWithOngoingAbsencesReturnsRecentAbsences</code> , <code>testGetStudentsWithOngoingAbsencesExcludesJustifiedAbsences</code> , <code>testGetStudentsWithOngoingAbsencesExcludesOldAbsences</code>
<code>hasStudentReturnedToClass()</code>	Vérifie le retour en classe	<code>testHasStudentReturnedToClassReturnsTrueAfter5PM</code> , <code>testHasStudentReturnedToClassReturnsFalseBefore5PM</code>

		<code>rnsFalseWithNewAbsences,</code> <code>testHasStudentReturnedToClassIgno</code> <code>resWeekends</code>
<code>recordStudentReturn()</code>	Enregistre un retour (UPSERT)	<code>testRecordStudentReturnInsertsNew</code> <code>Record,</code> <code>testRecordStudentReturnUpdatesExi</code> <code>stingRecord</code>
<code>calculateHalfDays()</code>	Calcule les demi-journées	<code>testCalculateHalfDaysMorningSessi</code> <code>on,</code> <code>testCalculateHalfDaysAfternoonSes</code> <code>sion,</code> <code>testCalculateHalfDaysFullDay,</code> <code>testCalculateHalfDaysMultipleDays</code>
<code>getStudentsAwaitingReturnN</code> <code>otification()</code>	Étudiants en attente de notification	<code>testGetStudentsAwaitingReturnNoti</code> <code>ficationReturnsCorrectStudents</code>
<code>markReturnNotificationSent</code> <code>()</code>	Marque notification envoyée	<code>testMarkReturnNotificationSentUpd</code> <code>atesRecord</code>
<code>getStudentsNeedingReminder</code> <code>()</code>	Étudiants nécessitant relance	<code>testGetStudentsNeedingReminderRet</code> <code>urnsStudentsAfter24Hours</code>
<code>cleanupOldRecords()</code>	Nettoyage des anciens enregistrements	<code>testCleanupOldRecordsDeletesOldRe</code> <code>cords,</code> <code>testCleanupOldRecordsKeepsRecentR</code> <code>ecords</code>

Project ▾

- Ⓢ AbsenceMonitoringModelTest.php x
- Ⓢ ProofModelTest.php
- Ⓢ UserModelTest.php

Ⓢ AbsenceModel.php

Ⓢ AbsenceMonitoringM

Ⓢ database.php

php email.php

```
1 <?php
2
3 namespace Tests\Unit\Model;
4
5 \Tests\Unit\Model > AbsenceMonitoringModelTest > testGetStudentsWithOngoingAbsencesReturnsRecentAbsences()
```

Run U AbsenceMonitoringModelTest x

Test R 13 sec 588 ms

✓ 17 tests passed 17 tests total, 13 sec 588 ms

PHP Warning: Version warning: Imagick was compiled against ImageMagick version 1809 but version 1810 is loaded. Imagick will

Warning: Version warning: Imagick was compiled against ImageMagick version 1809 but version 1810 is loaded. Imagick will

PHPUnit 12.4.5 by Sebastian Bergmann and contributors.

Runtime: PHP 8.4.12

Cleaned up 3 old absence monitoring records

Cleaned up 2 old absence monitoring records

Time: 00:24.046, Memory: 14.00 MB

OK, but there were issues!

Tests: 17, Assertions: 31, Warnings: 1, Deprecations: 1.

Process finished with exit code 0