# **Testing Documentation**

Test Containers were used for the DB interaction and Greenmail as a test email server for the email functionality. Further testing would require a test environment.

#### **Unit Tests**

#### Services

Test	Description	Test Steps	Test Method	Expected Output	Actual
Case ID TC_01	Successful user	1) Create a valid	AuthServiceTest	1) JWT token 2) User	Output As
10_01	registration	register request 2) Create a user based on the register request 3) Mock the encoder to return a hashed password 4) Mock the JWT service to return a token 5) Mock the user service 6) Call the register method	shouldRegisterUser()	service register() method called once	expected
TC_02	Successful user login	1) Create a valid login request 2) Mock the authentication manager to successfully authenticate 3) Mock the JWT service to return a token 4) Call the login method	AuthServiceTest shouldAuthenticateUser()	1) JWT token	As expected
TC_03	Failed login request when wrong credentials are provided	1) Create a valid login request 2) Mock the authentication manager to throw an exception when attempting to authenticate 3) Call	AuthServiceTest shouldThrow UnauthorizedException WhenAuthEmailOr PasswordIsWrong()	1) Unauthorized Exception with the relevant error message	As expected

		the login method			
TC_04	Successful JWT token assignment	1) Create a JWT token 2) Mock the JWT encoder to return the predefined token 3) Call the assign token method	JwtServiceTest shouldAssignToken()	1) JWT token	As expected
TC_05	Successful password reset token creation	1) Register a user 2) Create password reset request with existing email 3) Mock the email service 4) Call create password reset token method	Password Reset Service Test should Create Password Reset Token When User Is Found ()	1) Email sendPasswordResetE mail() method called once	As expected
TC_06	Failed password reset token creation when provided email doesn't exist in our DB	1) Register a user 2) Create password reset request with different email 3) Call create password reset token method	PasswordResetServiceTest shouldNotCreate PasswordResetToken WhenUserIsNotFound()	1) Email service was not called	As expected
TC_07	Failed password reset token validation for blank or malformed token	1) Create password reset request 2) Call reset password method	PasswordResetServiceTest shouldNotResetPassword WhenPasswordReset TokenIsBlankOr Malformed()	1) False	As expected
TC_08	Failed password reset token validation for expired token	1) Register a user 2) Create a password reset token and store it in the db 3) Create password reset request 4) Call reset password method	PasswordResetServiceTest shouldNotResetPassword WhenPasswordReset TokenExpired ()	1) False	As expected
TC_09	Invalidate all previous tokens when new password reset token is requested	1) Create a password reset token 2) Create a new Password reset request 3) Call create password reset token method	PasswordResetServiceTest shouldInvalidateAllPreviousT okensWhenNewResetTokenI sGenerated()	1) The initial token is not present in the database	As expected
TC_10	Successful	1) Register a user 2)	Password Reset Service Test	1) True 2)The email	As

Comparison of the continuation of the contin		password reset	Create a password	shouldResetPassword()	service	expected
Save it to the db 3)   Mock the email service 4) Create a Password reset confirmation request with the initial pre hashed token 5) Call the register method   Septend a password method   Septend and the token was deleted		password reset	•	Jilouidheseti assword()		CAPCCIEU
Mock the email service 4   Create a Password reset confirmation request with the initial pre hashed token 5) Call reset password method register user of the same user method   Duplicate Resource Exception with relevant error message						
Service 4) Create a Password reset confirmation request with the initial pre hashed token 5). Call reset password method   Successful user registration   1) Generate the user 2) Call the register with email that already exists   1) Register a user 2) Create a user with with username that already exists   1) Register a user 2)   UserServiceTest shouldThrow with username that already exists   1) Create a user with the same username 3) Call the register method   1) Register a user 2)   UserServiceTest shouldThrow with username that already exists   1) Create a user with the same username 3) Call the register method   1) Create a user with the same username 3) Call the register method   1) Create a user with the same username 3) Call the register method   1) Create a user with greater than 30 characters 2)   UserServiceTest shouldThrow   DuplicateResource Exception with relevant error message   AlreadyExists()   1) Duplicate   As expected with relevant error message   AlreadyExists()   1) Illegal Argument   As expected with username that already exists   1) Create a user with first name length greater than 30 characters 2) Call the validation method   ContainsNumbers()   1) Create a user with first name containing numbers validation when first name containing special characters 2) Call the validation method   ContainsSpecial characters 2) Call the validation method   Characters()   1) Illegal Argument   As expected relevant error message   ContainsSpecial characters 2) Call the validation method   Characters()   1) Illegal Argument   ContainsSpecial characters 2) Call the validation method   Characters()   1) Illegal Argument   Characters()   1) Illegal Argume			,		**	
Password reset confirmation request with the initial pre hashed token 5) Call reset password method password method registration registration registration with engister user with email that already exists  TC_12 Failed to register user with exist already exists  TC_13 Failed to 1) Register a user 2) Call the register method to register user with user and the token was deleted and						
confirmation request with the initial pre hashed token 5) Call reset password method  TC_11 Successful user registration = 2) Call the user 2) Call the register method = 2 (Call the register method = 2 (Call the register user with duser and the token was deleted = 2 (Call the register user with duser and the token was deleted = 2 (Call the register user with duser and the token was deleted = 2 (Call the register user with duser and the token was deleted = 2 (Call the register user with duser and the token was deleted = 2 (Call the register user with duser and the team exerial 3) (Call the register method = 2 (Call the register user with username that already exists = 2 (Call the register user with duser and the team exername that already exists = 2 (Call the validation when first name exceeds the maximum part on taining numbers contains and part of the validation when first name contains and part of the validation when first name contains and part of the validation when first name contains special characters = 2 (Call the validation method = 2 (Call the validation metho			•		,	
TC_11					·	
Initial pre hashed token 5) Call reset password method						
token 5) Call reset password method  TC_11 Successful user registration  TC_12 Failed to register user with email that already exists  TC_13 Failed to register user with user and the token was deleted user vith the same email 3) Call the register method  TC_14 Failed user vith user with user away exists  TC_15 Failed user validation when first name exceeds the max length defined in the DB column  TC_15 Failed user validation when first name contains a numbers  TC_16 Failed user validation when first name contains special characters 2) the failed user validation when first name contains special characters 2) the failed user validation when first name contains special characters 2) TC_16 Failed user validation when first name contains special characters 2) to					_	
TC_11 Successful user registration			•			
TC_12 Successful user registration register method user 2) Call the register method register method register user with email that already exists with username that already exists allegalary exists with username that already exists allegalary exists with username that already exists with first name exceeds the DB column with first name containing numbers validation when first name containing numbers validation when first name validation when first name containing special contains special characters 2) Call the validation method contains special characters 2) Call the validation wethod contains special characters 2) Call the validation wethod contains special characters 2) Call the validation wethod contains special characters 2) Call the v			· ·		• •	
TC_11 Successful user registration user 2) Call the register method  TC_12 Failed to register user with email that already exists  TC_13 Failed to register user with username that already exists  TC_14 Failed user validation when first name exceeds the DB column  TC_15 Failed user validation when first name containing numbers contains special characters 2) Call the register auser with first name containing special characters 2) Call the register auser with first name containing special characters 2) Call the register welfood  TC_16 Failed user validation when first name containing special characters 2) Call the register with first name containing special characters 2) Call the validation method  TC_16 Failed user validation when first name containing special characters 2) Call the validation method  TC_17 Failed user validation when first name containing special characters 2) Call the validation method contains special characters 2) Call the validation method characters 2) Call the validation method contains special characters 2) Call the validation contains special characters 2) Call the validation contains special characters 2) Call the validation contains			password method			
registration user 2) Call the register method  TC_12 Failed to 1) Register a user 2) Create a user with the same email 3) Call the register with username that already exists 2) Call the register wethod 2) Create a user with the same username that already exists 2) Call the register with username that already exists 4) Create a user with the same username that already exists 4) Create a user with the same username that already exists 4) Call the register with username that already exists 4) Call the register with the same username that already exists 4) Call the register with the same username that already exists 4) Call the register with the same username that already exists 4) Call the register with defined in the DB column 4. Call the validation when first name containing numbers 2) Call the validation when first name containing numbers 2) Call the validation when first name contains special characters 2) Call the validation when first name containing special characters 4) Create a user with first name containing special characters 2) Call the validation when first name containing special characters 2) Call the validation when first name containing special characters 2) Call the validation when first name containing special characters 2) Call the validation when first name containing special characters 2) Call the validation when first name containing special characters 2) Call the validation method 2 Characters 3) Create a user with first name containing special characters 3) Create a user with first name containing special characters 3) Create a user with first name containing special characters 3) Create a user with first name containing special characters 3) Create a user with first name containing special characters 3) Create a user with first name containing special characters 3) Create a user with first name containing special characters 3) Create a user 3) Create a us						
TC_12 Failed to register user vith the same email 3) already exists  TC_13 Failed to register user with the same email 3) Call the register method  TC_14 Failed user validation when first name containing numbers validation when first name contains special characters validation when the validation method contains special characters validation when the validation method contains special characters validation when the validation method contains special characters validation when the validation whe	TC_11	Successful user	1) Generate the	UserServiceTest	1) User is found	As
TC_12 Failed to register user with earne email 3) Call the register method		registration	user 2) Call the	shouldRegisterUser()	after registration	expected
TC_12 Failed to register user with earne email 3) Call the register method			register method			
register user with email that already exists  Call the register method  TC_13  Failed to register user with user and user with the same email 3)  TC_14  Failed user validation when first name exceeds the make length of the DB column  TC_15  TC_16  TC_16  TC_17  TC_17  TC_17  TC_18  TC_18  TC_19	TC_12	Failed to		UserServiceTest	1) Duplicate	As
with email that already exists  Call the register method  TC_13  Failed to register user with username that already exists  TC_14  Failed user validation when first name contains pumbers  TC_15  Failed user validation when first name first name numbers  TC_16  TC_16  Failed user validation when first name first name contains special coharacters 2)  TC_16  TC_16  Failed user validation when first name length greater than length quantity and efficient in the part of the part of the part of the part of the contains special characters  TC_16  TC_17  Failed user validation when first name contains special characters  TC_17  Failed user validation method  TC_17  Failed user validation when first name contains special characters  TC_17  Failed user validation when first name contains special characters  TC_17  Failed user validation method  TC_17  Failed user validation when first name contains special characters  TC_17  Failed user validation method  TC_17  Failed user validation method  TC_18  Failed user validation method  TC_19  Failed user validation when first name contains special characters 2) Call the validation method  TC_19  TC_19  Failed user validation when first name contains special characters 2) Call the validation method  TC_19  Failed user validation when first name contains special characters 2) Call the validation method  TC_19  Failed user validation when first name contains special characters 2) Call the validation method  TC_19  Failed user validation when first name contains special characters 2) Call the validation method  TC_19  Failed user validation when first name contains special characters 2) Call the validation method  TC_19  Failed user validation when first name contains special characters 2) Call the validation method  TC_19  Failed user validation when first name contains special characters 2) Call the validation method  TC_19  Failed user validation when first name contains special characters 2) Call the validation method  TC_19  Failed user validation when first name contains spec		register user		shouldThrow		expected
already exists  Call the register method  TC_13  Failed to register user with username that already exists  TC_14  Failed user validation when first name exceeds the DB column  TC_15  TC_15  TC_16  TC_16  Failed user validation when first name excentains numbers validation when first name enthod  TC_16  TC_16  Failed user validation when first name excentains and personness ontains a numbers validation when first name excentains and method  TC_16  TC_17  TC_17  Failed user validation when first name excents the method  TC_18  TC_19  Failed user validation when first name excentains and the pattern a		_	the same email 3)	DuplicateResource		
TC_13 Failed to register user with username that already exists 1) Create a user with the same user rame that already exists 2) Call the register method 2) Call the register max length defined in the DB column 1 CC_15 Failed user validation when first name containing numbers contains numbers validation when first name contains special characters 2) TC_17 Failed user 1) Create a user with first name and the validation method 2 TC_17 Failed user 1) Create a user with first name contains appears the validation method 2 TC_17 Failed user 1) Create a user with first name contains appears the validation method 2 TC_17 Failed user 1) Create a user with first name contains special characters 2) Call the validation method 2 TC_17 Failed user 1) Create a user with first name contains special characters 2) Call the validation method 2 TC_17 Failed user 1) Create a user With first name contains special characters 2) Call the validation method 2 TC_17 Failed user 1) Create a user With first name contains special characters 2) Call the validation method 3 TC_17 Failed user 1) Create a user With first name contains special characters 2) Call the validation method 3 TC_17 Failed user 1) Create a user With first name contains special Characters 3) Create a user With first name contains special Characters 4 To TC_18 Failed user 1) Create a user With first name contains special Characters 4 To Tc_18 Failed user 1) Create a user With first name Contains special Characters 4 To Tc_18 Failed user 1) Create a user With first name Contains special Characters 5 To Tc_18 Failed user 1) Create a user With first name Contains special Characters 5 To Tc_18 Failed user 1) Create a user With first name Contains special Characters 5 To Tc_18 Failed user 1) Create a user With first name Contains special Characters 5 To Tc_18 Failed user 1) Create a user With first name Characters 6 To Tc_18 Fa		already exists	•	•		
TC_13 Failed to register user with username that already exists		,	_	•		
register user with username that already exists				, 53.5, 2,1.535()		
register user with username that already exists  TC_14  Failed user validation when first name exceeds the method  TC_15  Failed user validation when first name exceeds the max length defined in the DB column  TC_16  TC_16  TC_16  TC_16  TC_16  TC_17  Failed user validation when first name exceeds the max length defined in the contains special characters 2)  TC_16  TC_17  Failed user validation when first name exceeds the max length defined in the part validation when first name containing special characters  TC_17  TC_17  Failed user validation when first name contains special characters  TC_17  TC_17  Failed user validation when first name contains special characters  TC_17  TC_17  TC_18  TC_19  TC	TC 13	Failed to	1) Register a user 2)	UserServiceTest	1) Duplicate	As
with username that already exists  TC_14  Failed user validation when first name exceeds the method  TC_15  Failed user validation when first name excontains numbers  TC_16  TC_16  TC_17  Failed user validation when first name exceeds the max length defined in the DB column  TC_17  TC_18  TC_19  Failed user validation when first name exceeds the max length defined in the DB column  TC_19  TC_19  Failed user validation when first name containing numbers 2) Call the validation method validation method validation when first name contains special characters 2) Call the validation when first name containing special characters  TC_17  Failed user validation when first name containing special characters  TC_17  Failed user 1) Create a user with first name containing special characters 2) Call the validation method ContainsSpecial characters 2) Call the validation when first name containing special characters 2) Call the validation method ContainsSpecial Characters()  TC_17  Failed user 1) Create a user ViserServiceTest 1) Illegal Argument Exception WhenFirstname ContainsSpecial Characters()  TC_17  Table Validation Vali						
that already exists    Social the register method   ExceptionIfUsername AlreadyExists()		_				
exists method AlreadyExists()  TC_14 Failed user validation when first name exceeds the max length defined in the DB column  TC_15 Failed user validation when first name excontaining numbers numbers validation when first name numbers validation when first name containing special characters 2 (Call the validation method and price to the validation when first name containing special characters the daracters are the validation when first name containing special characters are the validation method and provided the validation method are the validation when first name containing special characters are the validation method and provided the validation method are the validation are the validation method are the validation method are the validation are the validation are the validation method are the validation are the validation are the validation method are the validation are				T		
TC_14 Failed user validation when first name exceeds the max length defined in the DB column  TC_15 Failed user validation when first name excentation and before the part of		-			ilicssage	
validation when first name length greater than exceeds the max length defined in the DB column  TC_15  TC_15  TC_16  TC_16  TC_16  TC_17  TC_17  TC_17  TC_18  TC_18  TC_18  TC_19  TC_10  TC_10  TC_10  TC_10  TC_10  TC_10  TC_10  TC_110  TC_		CAISES	method	All Cady LXIStS()		
validation when first name length greater than exceeds the max length defined in the DB column  TC_15  TC_15  TC_16  TC_16  TC_16  TC_17  TC_17  TC_17  TC_18  TC_18  TC_18  TC_19  TC_10  TC_10  TC_10  TC_10  TC_10  TC_10  TC_10  TC_110  TC_	TC 14	Failed user	1) Create a user	UserServiceTest	1) Illegal Argument	As
first name exceeds the max length greater than exceeds the max length defined in the DB column  TC_15  TC_15  TC_16  TC_16  TC_16  TC_17  TC_17  Failed user validation when first name contains special characters 2) Call the validation method  TC_17  TC_17  Failed user validation when first name contains special characters 2) Call the validation when first name contains special characters 2) Call the validation when first name contains special characters 2) Call the validation method characters 3) Create a user 30 Characters 4) Illegal Argument message message method characters 4) Illegal Argument message message method characters 2) Call the validation method characters 3) Illegal Argument message m			,			
exceeds the max length defined in the DB column  TC_15 Failed user validation when first name containing numbers numbers validation when first name containing special characters 2) Call the validation when first name containing special characters 2) Call the validation when first name containing special characters 2) Call the validation when first name containing special characters 2) Call the validation when first name containing special characters 2) Call the validation method ContainsSpecial Characters 2) Call the validation method ContainsSpecial Characters ()  TC_17 Failed user 1) Create a user UserServiceTest ShouldThrow Exception with relevant error message  ExceptionWhenFirstname ContainsNumbers()  ExceptionWhenFirstname ShouldThrow Exception with relevant error message  TC_17 Failed user 1) Create a user UserServiceTest 1) Illegal Argument As						
max length defined in the DB column  TC_15 Failed user validation when first name containing numbers validation when first name containing numbers  TC_16 Failed user validation when first name containing numbers  TC_16 Failed user validation when first name containing special characters that characters the validation method  TC_17 Failed user 1) Create a user with first name containing special characters 2) Call the validation method ContainsSpecial characters 2) Call the validation method Characters()  TC_17 Failed user 1) Create a user UserServiceTest ShouldThrow IllegalArgument Exception with relevant error message  TC_17 Failed user 1) Create a user UserServiceTest 1) Illegal Argument Exception WhenFirstname ContainsSpecial Characters()  TC_17 Failed user 1) Create a user UserServiceTest 1) Illegal Argument As						
defined in the DB column  TC_15 Failed user validation when first name contains numbers  TC_16 Failed user validation when first name contains numbers  TC_16 TC_17 TC_18 TC_19 TC_1			,			
TC_15 Failed user validation when first name containing numbers 2) Call the numbers validation when first name containing numbers validation when numbers validation when first name containing special characters the validation method to contains special characters are the validation method to contains special characters are the validation method to contains Special characters 1) Create a user the validation method to contains Special characters 2) Call the validation method to contains Special characters 2) Call the validation method to contains Special characters 2) Call the validation method to contains Special characters 2) Call the validation method to contains Special characters 2) Call the validation method to contains Special characters 2) Call the validation method to contains Special characters 2) Call the validation method to contains Special characters 2) Call the validation method to contains Special characters 2) Call the validation method to contains Special characters 2) Call the validation method to contains Special characters 2) Call the sucception When First name contains Special characters 2) Call the validation method to contains Special characters 2) Call the sucception When First name contains Special characters 2) Call the validation method the validation the contains Special characters 2) Call the validation method the validation the validation the validation method the validation the		_				
TC_15 Failed user validation when first name containing numbers 2) Call the validation when numbers validation method  TC_16 Failed user validation when first name containing special characters the validation method  TC_17 Failed user validation when first name containing special characters the validation method the validation m			method	ivia/Lengui()		
validation when first name containing numbers 2) Call the validation method  TC_16  TC_16  TC_16  TC_17  Failed user  validation when first name containing special characters  TC_17  Failed user  validation when first name containing special characters  TC_17  Failed user  validation when first name containing special characters  TC_17  Failed user  validation when first name containing special characters  the validation method  TC_17  Failed user  Validation when first name containing special characters  the validation method  TC_17  TC_17  TC_18  Validation when first name containing special characters  TC_18  TC_19  ShouldThrow Exception with relevant error  Exception WhenFirstname  ContainsSpecial  ContainsSpecial  ContainsSpecial  ContainsSpecial  ContainsSpecial  ContainsSpecial  ContainsSpecial  ContainsSpecial  ContainsSpecial  TC_19  T	TC 15		1) Create a user		1) Illegal Argument	Δς
first name containing numbers 2) Call the validation method Contains Special characters the validation method Contains Special characters the validation method Contains Special characters 1) Create a user With first name Containing special Contains Special Characters 2) Call Contains Special Characters () TC_17 Failed user 1) Create a user UserServiceTest ExceptionWhenFirstname Containing special Characters () UserServiceTest I) Illegal Argument Contains Special Characters () TC_17 Failed user 1) Create a user UserServiceTest I) Illegal Argument As	'C_13		,			
contains numbers validation method ContainsNumbers()  TC_16 Failed user validation when first name containing special characters the validation method characters ()  TC_17 Failed user 1) Create a user with first name containing special the validation method Characters ()  ExceptionWhenFirstname ContainsNumbers()  UserServiceTest shouldThrow Exception with relevant error message  ContainsSpecial Characters()  TC_17 Failed user 1) Create a user UserServiceTest 1) Illegal Argument As					•	expected
numbers validation method ContainsNumbers()  TC_16 Failed user validation when first name containing special characters the validation method containing special characters  TC_17 Failed user validation method contains Numbers()  TC_18 Failed user validation method contains Numbers()  UserServiceTest validation shouldThrow in the shouldThrow in the sexpected in the validation method in the validation method in the validation method in the validation method in the validation of the validation method in the validation of the validation method in the validation method in the validation of the validation method in the validation of the validation method in the validation of			_			
TC_16 Failed user validation when first name containing special characters the validation method to the validation method to the validation of the valida			•	<u> </u>	message	
validation when first name containing special characters 2) Call the validation method CTC_17 Failed user validation vali	TC 16				4) 111 1 4	
first name containing special characters 2) Call the validation method  TC_17 Failed user    Containing special characters 2) Call the validation method   Characters ()   UserServiceTest   1) Illegal Argument   relevant error message   ContainsSpecial Characters()   To_17 Failed user   1) Create a user   UserServiceTest   1) Illegal Argument   As	1C_16		•			
contains special characters 2) Call the validation method Characters ()  TC_17 Failed user 1) Create a user ExceptionWhenFirstname ContainsSpecial Characters()  ExceptionWhenFirstname ContainsSpecial Characters()  Characters() 1) Illegal Argument As						expected
characters the validation ContainsSpecial Characters()  TC_17 Failed user 1) Create a user UserServiceTest 1) Illegal Argument As						
method Characters()  TC_17 Failed user 1) Create a user UserServiceTest 1) Illegal Argument As			·	-	message	
TC_17 Failed user 1) Create a user UserServiceTest 1) Illegal Argument As		characters				
			method			
validation when         with last name         shouldThrow         Exception with         expected	TC_17	Failed user	1) Create a user		1) Illegal Argument	As
		validation when	with last name	shouldThrow	Exception with	expected

	1	T			
	last name exceeds the max length	length greater than 30 characters 2) Call the validation	IllegalArgument ExceptionWhen Lastname Exceeds	relevant error message	
	defined in the DB column	method	MaxLength()		
TC_18	Failed user validation when last name contains	1) Create a user with last name containing special numbers 2) Call the	UserServiceTest shouldThrow IllegalArgument ExceptionWhenLastname	Illegal Argument     Exception with     relevant error     message	As expected
TC_19	numbers Failed user validation when last name contains special characters	validation method 1) Create a user with last name containing special characters 2) Call the validation	ContainsNumbers()  UserServiceTest shouldThrow IllegalArgument ExceptionWhenLastname ContainsSpecial	1) Illegal Argument Exception with relevant error message	As expected
TC_20	Failed user validation when username exceeds the max length defined in the DB column	method 1) Create a user with username length greater than 30 2) Call the validation method	Characters() UserServiceTest shouldThrow IllegalArgument ExceptionWhenUsername Exceeds MaxLength()	1) Illegal Argument Exception with relevant error message	As expected
TC_21	Failed user validation when email exceeds the max length defined in the DB column	1) Create a user with email length greater than 50 2) Call the validation method.	UserServiceTest shouldThrow IllegalArgument ExceptionWhenEmail ExceedsMaxLength()	1) Illegal Argument Exception with relevant error message	As expected
TC_22	Failed user validation when email doesn't contain '@' symbol	1) Create a user with email with no '@'s symbol 2) Call the validation method	UserServiceTest shouldThrow IllegalArgument ExceptionWhenEmail DoesNotContainAtSymbol()	1) Illegal Argument Exception with relevant error message	As expected
TC_23	Failed user validation when bio exceeds the max length defined in the DB column	1) Create a user with bio length greater than 150 characters 2) Call the validation method	UserServiceTest shouldThrow IllegalArgument ExceptionWhenBio Exceeds MaxLength()	1) Illegal Argument Exception with relevant error message	As expected
TC_24	Failed user validation when location exceeds the max length defined in the	1) Create a user with location length greater than 50 characters 2) Call the validation method	UserServiceTest shouldThrow IllegalArgument ExceptionWhenLocation Exceeds MaxLength()	1) Illegal Argument Exception with relevant error message	As expected

	DB column				
TC_25	Failed user validation when company exceeds the max length defined in the DB column	1) Create a user with company length greater than 50 characters 2) Call the validation method.	UserServiceTest shouldThrow IllegalArgument ExceptionWhenCompany Exceeds MaxLength()	1) Illegal Argument Exception with relevant error message	As expected
TC_26	Successful user retrieval	1) Register a user 2) Create a userDTO based on the registered user 3) Mock the JWT service to return the userId of our registered user 4) Call the get user method	UserServiceTest shouldGetUser()	1) UserDTO that matches the details of the retrieved user	As expected
TC_27	Failed to retrieve user when user is not found	1) Mock the JWT service to return a userId of a user that doesn't exist 2) Call the get user method	UserServiceTest shouldThrowResourceNot FoundException WhenUserIsNotFound()	1) Resource Not Found Exception with relevant error message	As expected
TC_28	Successfully getting user profile	1) Register a user 2) Mock the JWT service to return the userId of our registered user 3) Create a user profile 4) Call the get profile method	UserServiceTest shouldGet UserProfile()	1) The retrieved profile matches the created one	As expected
TC_29	Failed getting user profile when user is not found	1) Mock the JWT service to return a userId of a user that doesn't exist 2) Call the get profile method	UserServiceTest shouldThrowResource NotFoundExceptionWhenUs erIsNotFoundToGet Profile()	1) Resource Not Found Exception with relevant error message	As expected
TC_30	Successful user profile update when user is found and the update profile request is valid	1) Register a user 2) Create a valid user profile update request 3) Call the update profile method	UserServiceTest shouldUpdate UserProfile()	1) User profile update request values match the updated user profile in the database	As expected
TC_31	Failed to update user profile with	Register a user     Create a valid     user profile update	UserServiceTest shouldThrowDuplicate ResourceExceptionWhe	1) Duplicate Resource Exception with relevant error	As expected

TC_32	username that already exists  Failed user profile update request when user is not found	request 3) Mock the JWT service to return the userId of our registered user 4) Call the update user profile method 1) Create a valid user profile update request 2) Mock the JWT service to return a userId of a user that doesn't exist 3) Call the update profile	UsernameAlreadyExists ForProfileUpdate Request()  UserServiceTest shouldThrowResource NotFoundExceptionWhenUs erIsNotFoundToUpdateProfil e()	1) Resource Not Found Exception with relevant error message	As expected
TC_33	Successful user password update when user is found and the update password request is valid	method  1) Register a user 2) Create a valid user profile update request 3) Mock the JWT service to return the userId of our registered user 4) Call the update password method	UserServiceTest shouldUpdateUser Password()	1) The updated hashed password matches the password in our DB	As expected
TC_34	Failed user password update request when old password is wrong	1) Register a user 2) Create a valid user password update request with wrong old password	UserServiceTest shouldThrowBadCredentials ExceptionWhenOld PasswordIsWrong()	1) Bad Credentials Exception with relevant error message	As expected
TC_35	Failed user password update request when user is not found	1) Create a valid user password update request 2) Mock the JWT service to return a userld of a user that doesn't exist 3) Call the update password method	UserServiceTest shouldThrowResource NotFoundExceptionWhenUs erIsNotFoundToUpdatePass word()	1) Resource Not Found Exception with relevant error message	As expected
TC_36	Successful email update token creation	1) Register a user 2) Create a valid email update request 3) Mock the JWT service to return the userId of our registered user 4) Mock the email	UserServiceTest shouldCreateEmail UpdateToken()	1) Email sendEmailVerificatio nl() method called once	As expected

		service 5) Call the create email update token method			
TC_37	Failed user email update request when password is wrong	1) Register a user 2) Create an email update request with wrong password 3) Mock the JWT service to return the userId of our registered user 4) Call the create email update token method	UserServiceTest shouldThrowBad CredentialsExceptionWhen PasswordIsWrongFor EmailUpdate Request()	1) Bad Credentials Exception with relevant error message	As expected
TC_38	Failed user email updated request when with new email that already exists	1) Register a user 2) Create an email update request with an existing email 3) Mock the JWT service to return the userId of our registered user 4) Call the create email update token method	UserServiceTest shouldThrowDuplicate ResourceExceptionWhen NewEmailAlreadyExists ForEmailUpdateRequest()	1) Duplicate Resource Exception with relevant error message	As expected
TC_39	Failed email update token validation for blank or malformed token	1) Create a user email update request 2) Call the update email method	User Service Test should Not Update Email When Email Update Token Is Blank Or Malformed ()	1) False	As expected
TC_40	Failed email update token validation for expired token	1) Create a user email update request 2) Call update email method	UserServiceTest shouldNotUpdateEmail WhenEmailUpdateToken Expired ()	1) False	As expected
TC_41	Failed user email update request when user is not found	1) Create a user email update request 2) Mock the JWT service to return a userId of a user that doesn't exist 3) Call the create email update token method	UserServiceTest shouldThrowResource NotFoundExceptionWhen UserIsNotFoundTo UpdateEmail()	1) Resource Not Found Exception with relevant error message	As expected

TC_42	Invalidate all previous tokens when new email update token is requested	1) Register a user 2) Create an email update token 3) Create a new email update request 4) Mock the JWT service to return the userId of our registered user 5) Call create email update token method	UserServiceTest shouldInvalidateAll PreviousTokensWhenNewE mailUpdateToken IsGenerated()	1) The initial token is not present in the database	As expected
TC_43	Successful email update	1) Register a user 2) Create an email update token 3) Call update email method	UserServiceTest shouldUpdateEmail()	1) The provided token is not present in the database meaning the email was successfully updated and the token was deleted	As expected
TC_44	Failed getting user history when from date is null or empty	1) Register a user 2) Create only a to date 3) Mock the JWT service to return the userId of our registered user 4) Call the get history method	UserServiceTest shouldThrowIllegal ArgumentExceptionWhen FromIsNullOrEmpty()	1) Illegal Argument Exception with relevant error message	As expected
TC_45	Failed getting user history when to date is null or empty	1) Register a user 2) Create only a from date 3) Mock the JWT service to return the userId of our registered user 4) Call the get history method	UserServiceTest shouldThrowIllegal ArgumentExceptionWhenTo IsNullOrEmpty()	1) Illegal Argument Exception with relevant error message	As expected
TC_46	Failed getting user history when at least one date is not in the supported format (yyyy-MM-dd)	1) Register a user 2) Create one valid and one invalid date 3) Mock the JWT service to return the userId of our registered user 4) Call the get history method	UserServiceTest shouldThrowIllegal ArgumentExceptionWhenAt LeastOneDateIsNotInThe SupportedFormat	1) Illegal Argument Exception with relevant error message	As expected

TC_47	Successful user account deletion	1) Register a user 2) Create user account delete request 3) Mock the JWT service to return the userId of our registered user 4) Call the delete account method	UserServiceTest shouldDeleteAccount()	1) User profile is not found after account deletion	As expected
TC_48	Failed user account deletion when the provided password is wrong	1) Register a user 2) Create user account delete request with wrong password3) Mock the JWT service to return the userId of our registered user 4) Call the delete account method	UserServiceTest shouldThrowBad CredentialsException WhenPasswordIsWrong ForAccountDeletion()	1) Bad Credentials Exception with relevant error message	As expected
TC_49	No filtering when no constraints provided	1) Create a list of analysis reports 2) Create an empty constraints list 3) Call assess analysis result	Assessment Service Test should Not Filter Reports When No Constraints Are Provided ()	1) No interactions with the filtering service	As expected
TC_50	Empty compliant list when all reports are non-compliant with the constraints	1) Create a list of analysis reports 2) Create a list of constraints 3) Call the filter method	FilteringServiceTest shouldReturnAnEmpty CompliantListWhenAll ReportsAreNonCompliant WithTheConstraints()	1) Compliant list is empty 2) Non-compliant list has the size of the total reports	As expected
TC_51	We have both compliant and non-compliant reports	1) Create a list of analysis reports 2) Create a list of constraints 3) Call the filter method	FilteringServiceTest shouldReturnReports InBothLists()	1) Compliant list contains reports 2) Non-compliant list contains reports	As expected
TC_52	Empty non- compliant list when all reports are compliant with the constraints	1) Create a list of analysis reports 2) Create a list of constraints 3) Call the filter method	FilteringServiceTest shouldReturnAnEmpty NonCompliantListWhenAll ReportsAreCompliant WithTheConstraints()	1) Non-compliant list is empty 2) Compliant list has the size of the total reports	As expected
TC_53	Ranking reports when no	1) Create an empty preferences list 2)	RankingServiceTest shouldRankReportsWhen	1) The actual rank is the equal to	As expected

	preferences are provided	Create a quality metric report map 3) Call the rank tree method	NoPreferencesAre Provided()	expected one	
TC_54	Ranking reports when preferences are provided	1) Create a preferences list 2) Create a quality metric report map 3) Call the rank tree method	RankingServiceTest shouldRankReportsWhen PreferencesAre Provided()	1) The actual rank is the equal to expected one	As expected
TC_55	Successfully building the rank tree	1) Call the build tree method	TreeServiceTest shouldBuildTree()	All the parent nodes have the correct child nodes	As expected
TC_56	Failed analysis request when the sum of the weights of the child nodes is greater than 1	1) Call the build tree method 2) Create a list of preferences where the sum of the weights of the child nodes is greater than 1	TreeServiceTest shouldThrowIllegal ArgumentExceptionWhen TheSumOfTheWeights OfTheChildNodesIsGreater ThanOne()	1) Illegal Argument Exception with relevant error message	As expected
TC_57	Failed analysis request when the sum of the weights of all the child nodes is less than 1	1) Call the build tree method 2) Create a list of preferences where the sum of the weights of the child nodes is less than 1	TreeServiceTest shouldThrowIllegal ArgumentExceptionWhen TheSumOfTheWeights ForAllChildNodesIsLess ThanOne()	1) Illegal Argument Exception with relevant error message	As expected
TC_58	Successful analysis creation	1) Register a user 2) Create a list of preferences 3) Create a list of constraints 4) Create a list of analysis reports 5) Call save analysis process	Analysis Service Test should Save Analysis Process ()	1) The auto generated id from the database is a positive number	As expected
TC_59	Successfully getting user history	1) Register a user 2) Create a list of preferences 3) Create a list of constraints 4) Create a list of analysis reports 5) Call save analysis process 6) Call the get history method	AnalysisServiceTest shouldGetUser History()	1) The returning list of analyses is equal to 1	As expected

TC_60	Successful	1) Register a user	AnalysisServiceTest	1) Resource Not	As
	analysis	2) Create a list of	shouldDelete	Found Exception	expected
	deletion	preferences 3) Create a list of	Analysis()	with relevant error	
		constraints 4)		message	
		Create a list of			
		analysis reports 5)			
		Call save analysis			
		process 6) Call the			
		delete analysis			
TC C1	Cupperfully	method	AnalysisCarvinaTast	1) Detrieved project	٨٥
TC_61	Successfully getting analysis	<ol> <li>Register a user</li> <li>Create a list of</li> </ol>	Analysis Service Test should Get Analysis	1) Retrieved project urls, preferences and	As expected
	request	preferences 3)	Request()	constraints are the	expected
		Create a list of		same with the initial	
		constraints 4)		ones	
		Create a list of			
		analysis reports 5)			
		Call save analysis process 6) Call the			
		get analysis request			
		method			
TC_62	Successfully	1) Mock the docker	LanguageServiceTest	1) The returned map	As
	detecting	service to return a	shouldDetectLanguages()	contains 1 entry	expected
	languages	language with a		with the language as	
		percentage 2) Call the detect		key and the percentage as value	
		languages method		percentage as value	
TC_63	Successfully	1) Create a map	LanguageServiceTest	1) True	As
	identifying	with entries that at	shouldReturnTrue	,	expected
	supported	least 1 is a	WhenAtLeastOne		
	languages	supported	DetectedLanguage		
		language 2) Call	IsSupported()		
		verify supported languages method`			
TC_64	No supported	1) Create a map	LanguageServiceTest	1) False	As
_	languages	with entries that	shouldReturnFalse	,	expected
	detected	have no supported	WhenNoneOfThe		
		languages 2) Call	DetectedLanguage		
		verify supported	IsSupported()		
TC_65	Successfully	languages method  1) Create a quality	MetricServiceTest	1) The returned map	As
10_03	applying utility	metric report map	shouldApplyUtf()	has the same key	expected
	functions for	with key the quality	117 (7	value pairs as the	'
	each quality	metric and value		initial one	
	metric	the expected value			
		after applying the			

utf 2) Call apply utf		
method		

## Mappers

Test Case ID	Description	Test Steps	Test Method	Expected Output	Actual
TC_01	Successfully mapping a user to a userDTO	1) Create a user 2) Call the apply method	UserDTOMapperTest shouldMapUser ToUserDTO()	1) Initial user and UserDTO have the same values on the same properties	Output As expected
TC_02	Successfully mapping an analysis report record to an analysis report object	1) Create an analysis report object 2) Mock the result set to return an analysis report record 3) Call the map row method	Analysis Report Row Mapper Test should Map Row To Analysis Report ()	1) Initial analysis report and the row mapped one have the same values as	As expected
TC_03	Successfully mapping an analysis record to an analysis object	1) Create an analysis object 2) Mock the ResultSet to return an analysis record 3) Call the map row method	Analysis Row Mapper Test should Map Row To Analysis ()	1) Initial analysis and the row mapped one have the same values	As expected
TC_04	Successfully mapping a constraint record to a constraint object	1) Create a constraint object 2) Mock the result set to return a constraint record	Constraint Row Mapper Test should Map Row To Constraint ()	1) Initial constraint and the row mapped one have the same values	As expected
TC_05	Successfully mapping an email update token record to an email update token object	1) Mock the result set to return an email update token record	Email Update Token Row Mapper Test should Map Row To Email Update Token ()	1) Email update token object returned from row mapper has the expected values	As expected
TC_06	Successfully mapping a password reset token record to a password reset token object	1) Mock the result set to return a password reset token record	Password Reset Token Row Mapper Test should Map Row To Password Reset Token ()	1) Password reset token object returned from row mapper has the expected values	As expected

TC_07	Successfully	1) Create a	PreferenceRowMapper	1) Initial preference	As
	mapping a	preference object	Test	and the row mapped	expected
	preference	2) Mock the result	should Map Row	one have the same	
	record to a	set to return a	ToPreference()	values	
	preference	preference record			
	object				
TC_08	Successfully	1) Create a user	UserRowMapperTest	1) Initial user and	As
	mapping a user	object 2) Mock the	${\sf shouldMapRow}$	the row mapped one	expected
	record to a user	result set to return	ToUser()	have the same	
	object	a user record		values	

#### Deserializers

Test	Description	Test Steps	Test Method	Expected Output	Actual
Case ID					Output
TC_01	Successfully deserializing a string representation of a quality attribute to its corresponding enum value	1) Create a string representation of a quality attribute 2) Mock the parser to return the initial string 3) Call the deserialize method	Quality Attribute Deserializer Test should Deserialize Quality Attribute ()	1) The deserialized quality attribute enum matches the initial string representation	As expected
TC_02	Successfully deserializing a string representation of a quality attribute ignoring case and extra spaces to its corresponding enum value	1) Create a string representation of a quality attribute with mixed cases and extra spaces 2) Mock the parser to return the initial string 3) Call the deserialize method	QualityAttribute DeserializerTest shouldDeserializeQuality AttributeIgnoringCase AndExtraSpaces()	1) The deserialized quality attribute enum matches the initial string representation ignoring case and extra spaces	As expected
TC_03	Failed quality attribute deserialization for empty, invalid string input	1) Mock the parser to return the initial string 2) Call the deserialize method	QualityAttribute DeserializerTest shoudIThrowIllegal ArgumentExceptionWhen QualityAttribute IsInvalid()	1) Illegal Argument Exception with relevant error message	As expected
TC_04	Successfully deserializing a string representation	1) Create a string representation of a quality metric 2) Mock the parser to	Quality Metric Deserializer Test should Deserialize Quality Metric ()	1) The deserialized quality metric enum matches the initial string	As expected

	of a quality metric to its corresponding enum value	return the initial string 3) Call the deserialize method		representation	
TC_05	Successfully deserializing a string representation of a quality metric ignoring case and extra spaces to its corresponding enum value	1) Create a string representation of a quality metric with mixed cases and extra spaces 2) Mock the parser to return the initial string 3) Call the deserialize method	QualityMetric DeserializerTest shouldDeserializeQuality MetricIgnoringCase AndExtraSpaces()	1) The deserialized quality metric enum matches the initial string representation ignoring case and extra spaces	As expected
TC_06	Failed quality metric deserialization for empty, invalid string input	1) Mock the parser to return the initial string 2) Call the deserialize method	QualityMetric DeserializerTest shoudIThrowIllegal ArgumentExceptionWhen QualityMetric IsInvalid()	1) Illegal Argument Exception with relevant error message	As expected
TC_07	Successfully deserializing a string representation of an operator symbol to its corresponding Quality metric operator enum value	1) Create a string representation of an operator symbol 2) Mock the parser to return the initial string 3) Call the deserialize method	QualityMetricOperator DeserializerTest shouldDeserializeSymbolTo QualityMetricOperator()	1) The deserialized operator's symbol matches the initial symbol	As expected
TC_08	Failed quality metric operator deserialization for empty, invalid string input	1) Mock the parser to return the initial string 2) Call the deserialize method	QualityMetricOperator DeserializerTest shoudIThrowIllegal ArgumentExceptionWhen SymbolIsInvalid()	1) Illegal Argument Exception with relevant error message	As expected

## Controllers

Test	Description	Test Steps	Test Method	Expected Output	Actual
Case ID					Output
TC_01	401	1) Construct a JSON	Analysis Controller Test	1) Status	As
	Unauthorized:	payload for the	shouldReturnHTTP401	UNAUTHORIZED 2)	expected
	When	analysis request 2)	When Analyze Is Called	No interactions with	

	unauthenticate d user calls the analyze endpoint	Perform a POST request to the analyze endpoint	ByUnauthenticated User()	the analysis service	
TC_02	400 Bad Request: When authenticated user provides an empty list of repositories	1) Construct a JSON payload for the analysis request with an empty projects urls list 2) Perform a POST request to the analyze endpoint	Analysis Controller Test should Return HTTP 400 When List Of Project Urls Is Empty()	1) Status BAD_REQUEST 2) No interactions with the analysis service	As expected
TC_03	400 Bad Request: When authenticated user provides null for project urls	1) Construct a JSON payload for the analysis request where project urls list is null 2) Perform a POST request to the analyze endpoint	Analysis Controller Test should Return HTTP 400 When List Of Project Urls Is Null ()	1) Status BAD_REQUEST 2) No interactions with the analysis service	As expected
TC_04	400 Bad Request: When authenticated user provides null for a constraint quality metric	1) Construct a JSON payload for the analysis request where a constraint quality metric is null 2) Perform a POST request to the analyze endpoint	Analysis Controller Test should Return HTTP 400 When Constraint Quality Metricls Null()	1) Status BAD_REQUEST 2) No interactions with the analysis service	As expected
TC_05	400 Bad Request: When authenticated user provides null for a constraint quality metric operator	1) Construct a JSON payload for the analysis request where a constraint quality metric operator is null 2) Perform a POST request to the analyze endpoint	Analysis Controller Test should Return HTTP 400 When Constraint Quality Metric Operator Is Null ()	1) Status BAD_REQUEST 2) No interactions with the analysis service	As expected
TC_06	400 Bad Request: When authenticated user provides null for a constraint threshold	1) Construct a JSON payload for the analysis request where a constraint threshold is null 2) Perform a POST request to the analyze endpoint	Analysis Controller Test should Return HTTP 400 When Constraint Threshold Is Null ()	1) Status BAD_REQUEST 2) No interactions with the analysis service	As expected
TC_07	400 Bad	1) Construct a JSON	AnalysisControllerTest	1) Status	As

	Request: When authenticated user provides invalid values for a constraint threshold	payload for the analysis request where a constraint threshold has invalid values 2) Perform a POST request to the analyze endpoint	shouldReturnHTTP400 WhenConstraintThreshold IsInvalid()	BAD_REQUEST 2) No interactions with the analysis service	expected
TC_08	400 Bad Request: When authenticated user provides null for a preference quality attribute	1) Construct a JSON payload for the analysis request where a preference quality attribute is null 2) Perform a POST request to the analyze endpoint	Analysis Controller Test should Return HTTP 400 When Preference Quality Attribute Is Null ()	1) Status BAD_REQUEST 2) No interactions with the analysis service	As expected
TC_09	400 Bad Request: When authenticated user provides null for a preference weight	1) Construct a JSON payload for the analysis request where a preference weight is null 2) Perform a POST request to the analyze endpoint	AnalysisControllerTest shouldReturnHTTP400 WhenPreferenceWeight IsNull()	1) Status BAD_REQUEST 2) No interactions with the analysis service	As expected
TC_10	400 Bad Request: When authenticated user provides invalid values for a preference weight	1) Construct a JSON payload for the analysis request where a preference weight has invalid values 2) Perform a POST request to the analyze endpoint	AnalysisControllerTest shouldReturnHTTP400 WhenPreferenceWeight IsInvalid()	1) Status BAD_REQUEST 2) No interactions with the analysis service	As expected
TC_11	401 Unauthorized: When unauthenticate d user calls analysis result endpoint	1) Perform a GET request to analysis result endpoint	AnalysisControllerTest shouldReturnHTTP401When GetAnalysisResultIsCalled ByUnauthenticated User()	1) Status UNAUTHORIZED 2) No interactions with the analysis service	As expected
TC_12	401 Unauthorized: When unauthenticate d user calls the refresh request	1) Construct a JSON payload for the refresh request 2) Perform a PUT request to the refresh request	AnalysisControllerTest shouldReturnHTTP401When RefreshAnalysisResult IsCalled ByUnauthenticatedUser()	1) Status UNAUTHORIZED 2) No interactions with the analysis service	As expected

	endpoint	endpoint			
TC_13	401 Unauthorized: When unauthenticate d user calls delete analysis endpoint	1) Perform a DELETE request to the analysis endpoint	AnalysisControllerTest shouldReturnHTTP401When DeleteAnalysisIsCalled ByUnauthenticated User()	1) Status UNAUTHORIZED 2) No interactions with the analysis service	As expected
TC_14	401 Unauthorized: When unauthenticate d user calls analysis request endpoint	1) Perform a GET request to the analysis request endpoint	Analysis Controller Test should Return HTTP 401 When Get Analysis Request tls Called By Unauthenticated User()	1) Status UNAUTHORIZED 2) No interactions with the analysis service	As expected
TC_15	201 Created, Jwt token in the respone body: Successful user registration	1) Construct a JSON payload for the register request 2) Construct a JSON payload with the expected response body 3) Mock the authentication service to return a jwt token 4) Perform a POST request to the signup endpoint	AuthControllerTest shouldReturnJwtToken AndHTTP201WhenUserIs RegisteredSuccessfully()	1) Status CREATED 2) The response body contains the token returned from the auth service	As expected
TC_15	400 Bad Request: Failed user registration when first name is null or empty	1) Construct a JSON payload for the register request where firstname is null and empty 2) Construct a JSON payload with the expected response body 3) Perform a POST request to the signup endpoint	AuthControllerTest shouldReturnHTTP400 WhenRegisterFirstname IsNullOrEmpty()	1) Status BAD_REQUEST 2) The response body contains relevant error message	As expected
TC_16	400 Bad Request: Failed user	1) Construct a JSON payload for the register request	Auth Controller Test should Return HTTP 400 When Register Lastname	1) Status BAD_REQUEST 2) The response body	As expected

	registration when last name is null or empty	where lastname is null and empty 2) Construct a JSON payload with the expected response body 3) Perform a POST request to the signup endpoint	IsNullOrEmpty()	contains relevant error message	
TC_17	400 Bad Request: Failed user registration when username is null or empty	1) Construct a JSON payload for the register request where username is null and empty 2) Construct a JSON payload with the expected response body 3) Perform a POST request to the signup endpoint	AuthControllerTest shouldReturnHTTP400 WhenRegisterUsername IsNullOrEmpty()	1) Status BAD_REQUEST 2) The response body contains relevant error message	As expected
TC_18	400 Bad Request: Failed user registration when email is null or empty	1) Construct a JSON payload for the register request where email is null and empty 2) Construct a JSON payload with the expected response body 3) Perform a POST request to the signup endpoint	AuthControllerTest shouldReturnHTTP400 WhenRegisterEmail IsNullOrEmpty()	1) Status BAD_REQUEST 2) The response body contains relevant error message	As expected
TC_19	400 Bad Request: Failed user registration when password is null or empty	1) Construct a JSON payload for the register request where password is null and empty 2) Construct a JSON payload with the expected response body 3) Perform a POST request to the signup endpoint	AuthControllerTest shouldReturnHTTP400 WhenRegisterPassword IsNullOrEmpty()	1) Status BAD_REQUEST 2) The response body contains relevant error message	As expected

TC_20	200 OK, Jwt token in the respone body: Successful user login	1) Construct a JSON payload for the login request 2) Construct a JSON payload with the expected response body 3) Mock the authentication service to return a jwt token 4) Perform a POST request to the login endpoint	AuthControllerTest shouldReturnJwtToken AndHTTP200WhenUserIs LoggedInSuccessfully()	1) Status OK 2) The response body contains the token returned from the auth service	As expected
TC_21	400 Bad Request: Failed user login when email is null or empty	1) Construct a JSON payload for the login request where email is null and empty 2) Construct a JSON payload with the expected response body 3) Perform a POST request to the login endpoint	AuthControllerTest shouldReturnHTTP400 WhenLoginEmail IsNullOrEmpty()	1) Status BAD_REQUEST 2) The response body contains relevant error message	As expected
TC_22	400 Bad Request: Failed user login when password is null or empty	1) Construct a JSON payload for the login request where password is null and empty 2) Construct a JSON payload with the expected response body 3) Perform a POST request to the login endpoint	AuthControllerTest shouldReturnHTTP400 WhenLoginPassword IsNullOrEmpty()	1) Status BAD_REQUEST 2) The response body contains relevant error message	As expected
TC_23	202 Accepted: When user makes a password reset request	1) Construct a JSON payload for the password reset request 2) Mock the password reset service 3) Perform a POST request to the password reset endpoint	AuthControllerTest shouldReturnHTTP202 ForPasswordResetRequest RegardlessIfTheEmail Exists()	1) Status ACCEPTED	As expected

TC_24	400 Bad Request: Failed password reset request when email is null or empty	1) Construct a JSON payload for the password reset request where email is null and empty 2) Construct a JSON payload with the expected response body 3) Perform a POST request to the password reset endpoint	AuthControllerTest shouldReturnHTTP400 WhenPasswordResetEmail IsNullOrEmpty()	1) Status BAD_REQUEST 2) The response body contains relevant error message	As expected
TC_25	401 Unauthorized: Failed password reset when token invalid (empty, malformed, expiry)	1) Construct a JSON payload for the password reset confirmation request where token is null and empty 2) Construct a JSON payload with the expected response body 3) Perform a PUT request to the password reset confirm endpoint	AuthControllerTest shouldReturnHTTP401 WhenPasswordReset ConfirmationToken IsInvalid()	1) Status UNAUTHORIZED 2) The response body contains relevant error message	As expected
TC_26	400 Bad Request: Failed password reset when new password is null or empty	1) Construct a JSON payload for the password reset confirmation request where password is null and empty 2) Construct a JSON payload with the expected response body 3) Perform a PUT request to the password reset confirm endpoint	AuthControllerTest shouldReturnHTTP400 WhenPasswordReset ConfirmationPassword IsNullOrEmpty()	1) Status BAD_REQUEST 2) The response body contains relevant error message	As expected
TC_27	400 Bad Request: Failed password reset when new password does not meet the	1) Construct a JSON payload for the password reset confirmation request where password does not	AuthControllerTest shouldReturnHTTP400 WhenPasswordReset ConfirmationPassword DoesNotMeetThe Requirements()	1) Status BAD_REQUEST 2) The response body contains relevant error message	As expected

			T	1	
	requirements	meet the requirements 2) Construct a JSON payload with the expected response body 3) Perform a PUT request to the password reset confirm endpoint			
TC_28	204 No Content: Successful password reset	1) Construct a JSON payload for the password reset confirmation request 2) Perform a PUT request to the password reset confirm endpoint	AuthControllerTest shouldReturnHTTP204Whe TokenAndNewPassword AreValid()	1) Status NO_CONTENT	As expected
TC_29	200 OK, UserDTO in the response body: Successfully getting user	1) Create a UserDTO object 2) Construct a JSON payload with the expected response body 3) Mock the user service to return the DTO 4) Perform a GET request to the user endpoint	UserControllerTest shouldReturnUserDTO AndHTTP200()	1) Status OK 2) The response body has the same values as our initial UserDTO	As expected
TC_30	404 Not found: When user is not found	1) Construct a JSON payload with the expected response body 2) Mock the user service to throw a Not Found exception 3) Perform a GET request to the user endpoint	UserControllerTest shouldThrowNotFound ExceptionWhenUserIs NotFound()	1) Status NOT_FOUND 2) The response body contains relevant error message	As expected
TC_31	401 Unauthorized: When unauthenticate d user calls get user	1) Perform a GET request to the user endpoint	UserControllerTest shouldReturnHTTP401When GetUserIsCalledBy UnauthenticatedUser()	1) Status UNAUTHORIZED 2) No interactions with the user service	As expected
TC_32	200 OK, UserProfile in the response	1) Create a User Profile object 2) Construct a JSON	UserControllerTest shouldReturnUserProfile AndHTTP200()	1) Status OK 2) The response body has the same values as	As expected

	body: Successfully getting user profile	payload with the expected response body 3) Mock the user service to return the profile 4) Perform a GET request to the user profile endpoint		our initial UserProfile object	
TC_33	404 Not found: When user is not found to get profile	1) Construct a JSON payload with the expected response body 2) Mock the user service to throw a Resource Not Found exception 3) Perform a GET request to the update profile endpoint	UserControllerTest shouldReturnHTTP404 WhenUserIsNotFound ToGetProfile()	1) Status NOT_FOUND 2) The response body contains relevant error message	As expected
TC_34	401 Unauthorized: When unauthenticate d user calls get profile	1) Perform a GET request to the user profile endpoint	UserControllerTest shouldReturnHTTP401When GetProfileIsCalledBy UnauthenticatedUser()	1) Status UNAUTHORIZED 2) No interactions with the user service	As expected
TC_35	204 No Content: Successfully updating user profile	1) Construct a JSON payload for the user profile update request 2) Mock the user service 3) Perform a PUT request to the update profile endpoint	UserControllerTest shouldReturnHTTP204When ProfileIsUpdated()	1) Status NO_CONTENT 2) User service update profile method was called once	As expected
TC_36	409 Conflict: Failed user profile update when the provided username already exists	1) Construct a JSON payload for the user profile update request 2) Construct a JSON payload with the expected response body 3) Mock the user service to throw a Duplicate Resource exception 4) Perform a PUT request to the	UserControllerTest shouldReturnHTTP409 WhenUpdatingProfileWith ExistingUsername()	1) Status CONFLICT 2) The response body contains relevant error message	As expected

		update profile			
TC_37	404 Not found: When user is not found to update profile	1) Construct a JSON payload for the user profile update request 2) Construct a JSON payload with the expected response body 3) Mock the user service to throw a Resource Not Found exception 4) Perform a PUT request to the update profile endpoint	UserControllerTest shouldReturnHTTP404 WhenUserIsNotFound ToUpdateProfile()	1) Status NOT_FOUND 2) The response body contains relevant error message	As expected
TC_38	401 Unauthorized: When unauthenticate d user calls update profile	1) Construct a JSON payload for the user profile update request 2) Perform a PUT request to the update profile endpoint	User Controller Test should Return HTTP 401 When Update Profile Is Called By Unauthenticated User ()	1) Status UNAUTHORIZED 2) No interactions with the user service	As expected
TC_39	204 No Content: Successful user password update	1) Construct a JSON payload for the user password update request 2) Mock the user service 3) Perform a PUT request to the update password endpoint	UserControllerTest shouldReturnHTTP204When PasswordIsUpdated()	1) Status NO_CONTENT 2) User service update password method was called once	As expected
TC_40	400 Bad Request: Failed password update when old password is null or empty	1) Construct a JSON payload for the user password update request where old password is null and empty 2) Construct a JSON payload with the expected response body 3) Perform a PUT request to the update password endpoint	User Controller Test should Return HTTP 400 When Old Password Is Null Or Empty ()	1) Status BAD_REQUEST 2) The response body contains relevant error message 3) No interactions with the user service	As expected

TC 44	400 D	1) C	11C + 11 T +	1) C+-+-	Α
TC_41	400 Bad Request: Failed password update when old password is wrong	1) Construct a JSON payload for the user password update request 2) Construct a JSON payload with the expected response body 3) Mock the user service to throw a Bad Credentials exception M3) Perform a PUT request to the update password endpoint	UserControllerTest shouldReturnHTTP400 WhenOldPassword IsWrong ()	1) Status BAD_REQUEST 2) The response body contains relevant error message	As expected
TC_42	400 Bad Request: Failed password update when new password is null or empty	1) Construct a JSON payload for the user password update request where new password is null and empty 2) Construct a JSON payload with the expected response body 3) Perform a PUT request to the update password endpoint	UserControllerTest shouldReturnHTTP400 WhenNewPassword IsNullOrEmpty()	1) Status BAD_REQUEST 2) The response body contains relevant error message 3) No interactions with the user service	As expected
TC_43	400 Bad Request: Failed password update when new password does not meet the requirements	1) Construct a JSON payload for the user password update request 2) Construct a JSON payload with the expected response body 3) Perform a PUT request to the update password endpoint	User Controller Test should Return HTTP 400 When New Password Does Not Meet Requeir ments ()	1) Status BAD_REQUEST 2) The response body contains relevant error message	As expected
TC_44	404 Not found: When user is not found to update password	1) Construct a JSON payload for the user password update request 2) Construct a JSON payload with the expected response	User Controller Test should Return HTTP 404 When User Is Not Found To Update Password ()	1) Status NOT_FOUND 2) The response body contains relevant error message	As expected

	T	Т			1
		body 3) Mock the user service to throw a Resource Not Found exception 4) Perform a PUT request to the			
		update password			
TC_45	401 Unauthorized: When unauthenticate d user calls update password	endpoint  1) Construct a JSON payload for the user password update request 2) Perform a PUT request to the update password endpoint	UserControllerTest shouldReturnHTTP401When UpdatePasswordIsCalledBy UnauthenticatedUser()	1) Status UNAUTHORIZED 2) No interactions with the user service	As expected
TC_46	202 Accepted: Sending a link to the new user's email for email update	1) Construct a JSON payload for the user email update request 2) Mock the user service 3) Perform a POST request to the update email endpoint	UserControllerTest shouldReturn202When EmailTokenIsCreated ForEmailUpdate()	1) Status ACCEPTED 2) User service create email update token method was called once	As expected
TC_49	400 Bad Request: Failed email update when password is null or empty	1) Construct a JSON payload for the user email update request where password is null and empty 2) Construct a JSON payload with the expected response body 3) Perform a POST request to the update email endpoint	UserControllerTest shouldReturnHTTP400 WhenPasswordIsNullOr EmptyForEmailUpdate()	1) Status BAD_REQUEST 2) The response body contains relevant error message 3) No interactions with the user service	As expected
TC_50	400 Bad Request: Failed to update email when provided password is wrong	1) Construct a JSON payload for the user email update request 2) Construct a JSON payload with the expected response body 3) Mock the user service to	UserControllerTest shouldReturnHTTP400When WrongPasswrodIsProvided ForEmailUpdate()	1) Status BAD_REQUEST 2) The response body contains relevant error message	As expected

TC_51	400 Bad Request: Failed email update when new email is null or empty	throw a Bad Credentials exception 4) Perform a POST request to the update email endpoint 1) Create a json user email update request request body where new email is null and empty 2) Construct a JSON payload with the expected response body 3) Perform a POST request to the update email endpoint	UserControllerTest shouldReturnHTTP400 WhenEmailIsNullOr EmptyForEmailUpdate()	1) Status BAD_REQUEST 2) The response body contains relevant error message 3) No interactions with the user service	As expected
TC_52	409 Conflict: Failed to update email when new email already exists	1) Construct a JSON payload for the user email update request 2) Construct a JSON payload with the expected response body 3) Mock the user service to throw a Duplicate Resource exception 4) Perform a POST request to the update email endpoint	UserControllerTest shouldReturnHTTP409When UpdatingEmailWith ExistingEmail()	1) Status CONFLICT 2) The response body contains relevant error message	As expected
TC_53	3XX Redirection: Successful user redirection to their profile after email update	1) Mock the user service update email to return true 2) Perform a GET request to the update email endpoint	UserControllerTest shouldUpdateEmailAnd RedireUserToTheir ProfileWhenEmailUpdate TokenIsValid()	1) Status 3XXRedirection 2) The Location header contains the redirection url	As expected
TC_54	3XX Redirection: Successful user redirection to email update error page after	1) Mock the user service update email to return true 2) Perform a GET request to the update email	UserControllerTest shouldNotUpdateEmailAnd RedirectUserToEmailUpdate ErrorPageWhenEmail UpdateTokenIsInValid()	1) Status 3XXRedirection 2) The Location header contains the redirection url	As expected

	failed email	endpoint			
	update				
TC_55	404 Not found: When user is not found to update email	1) Construct a JSON payload for the user email update request 2) Construct a JSON payload with the expected response body 3) Mock the user service to throw a Resource Not Found exception 3) Perform a POST request to the update email endpoint	UserControllerTest shouldReturnHTTP404 WhenUserIsNotFound ToUpdateEmail ()	1) Status NOT_FOUND 2) The response body contains relevant error message	As expected
TC_56	401 Unauthorized: When unauthenticate d user calls update email	1) Construct a JSON payload for the user email update request 2) Perform a POST request to the update email endpoint	UserControllerTest shouldReturnHTTP401When UpdateEmailIsCalledBy UnauthenticatedUser()	1) Status UNAUTHORIZED 2) No interactions with the user service	As expected
TC_57	400 Bad Request: When an invalid date is provided to get user history in range	1) Construct a JSON payload with the expected response body 3) Mock the user service to throw an Illegal State exception 3) Perform a GET request to the user history endpoint	UserControllerTest shouldReturnHTTP400When AtLeastOneDateIsInvalid ToGetUserHistoryInRange	1) Status BAD_REQUEST 2) The response body contains relevant error message	As expected
TC_58	401 Unauthorized: When unauthenticate d user calls get history	1) Perform a GET request to the user history endpoint	UserControllerTest shouldReturnHTTP401When GetHistoryIsCalledBy UnauthenticatedUser()	1) Status UNAUTHORIZED 2) No interactions with the user service	As expected
TC_59	204 No Content: Successful account deletion	1) Construct a JSON payload for the user account delete request 2) Mock the user service 3) Perform a PUT	UserControllerTest shouldReturnHTTP204When AccountIsDeleted()	1) Status NO_CONTENT 2) User service delete account was called once	As expected

		request to the user account endpoint			
TC_60	400 Bad Request: Failed account deletion when password is null or empty	1) Construct a JSON payload for the user account delete request where password is null and empty 2) Construct a JSON payload with the expected response body 3) Perform a PUT request to the user account endpoint	UserControllerTest shouldReturnHTTP400 WhenPasswordIsNullOr Empty ForAccount Deletion()	1) Status BAD_REQUEST 2) The response body contains relevant error message 3) No interactions with the user service	As expected
TC_61	400 Bad Request: Failed account deletion when password is wrong	1) Construct a JSON payload for the user account delete request 2) Construct a JSON payload with the expected response body 3) Mock the user service to throw Bad Credentials exception 4) Perform a PUT request to the user account endpoint	UserControllerTest shouldReturnHTTP400When PasswordIsWrongFor AccountDeletion()	1) Status BAD_REQUEST 2) The response body contains relevant error message	As expected
TC_62	401 Unauthorized: When unauthenticate d user calls delete account	1) Construct a JSON payload for the user account delete request 2) Perform a PUT request to the user account endpoint	UserControllerTest shouldReturnHTTP401When DeleteAccountIsCalledBy UnauthenticatedUser()	1) Status UNAUTHORIZED 2) No interactions with the user service	As expected

## Integration Tests

Test	Description	Test Steps	Test Method	Expected Output	Actual	
Case ID					Output	
TC_01	Successful user	1) Create a JSON	AuthIT	1) Status CREATED	As	
	login	payload for the	shouldLoginUser()	for signup 2) Status	expected	
		register request 2)		OK for login 3) JWT		

	T	1			
		Perform a POST		is present in the	
		request to the		response body in	
		signup endpoint 3)		both cases	
		Create a JSON			
		payload for the			
		login request 4)			
		Perform a POST			
		request to the login			
		endpoint			
TC 02	Successful	1) Create a JSON	AuthIT	1) Status ACCEPTED	As
TC_02				for the initial	
	password reset	payload for the	should Reset Password ()		expected
		register request 2)		request 2) Status	
		Perform a POST		NO_CONTENT for	
		request to the		the final request 3)	
		signup endpoint 3)		User received a link	
		Create a JSON		on their email to	
		payload for the		reset their password	
		password reset		4) User received a	
		request 4) Perform		confirmation email	
		a POST request to		that the password	
		the password reset		was reset	
		endpoint 5) Create			
		a JSON payload for			
		the password reset			
		confirmation			
		request with the			
		token from the			
		email link and the			
		new password 6)			
		Perform a PUT			
		request to the			
		password reset			
		confirmation			
		endpoint			
TC_03	Successfully	1) Create a JSON	UserIT	1) Status OK 2)	As
	getting user	payload for the	shouldGetUser()	Response values	expected
		register request 2)	V	match user values	
		Perform a POST			
		request to the			
		signup endpoint			
		and receive a JWT			
		3) Perform a GET			
		· ·			
		request to the user			
		endpoint with the			
TO 01	6 6 11	token		1) ()   () ()	
TC_04	Successfully	1) Create a JSON	UserIT	1) Status OK 2)	As
	getting user	payload for the	shouldGetUserProfile()	Response values	expected
	profile	register request 2)		match user profile	

		Perform a POST		values	
		request to the		values	
		signup endpoint			
		and receive a JWT			
		3) Perform a GET			
		request to the user			
		profile endpoint			
		with the token			
TC_05	Successful user	1) Create a JSON	UserlT	1) Status	As
	profile update	payload for the	should Update User Profile ()	NO_CONTENT for	expected
		register request 2)		the update request	
		Perform a POST		2) Response values	
		request to the		match user profile	
		signup endpoint		updated values	
		and receive a JWT			
		3) Create a JSON			
		payload for the			
		1 -			
		user profile update			
		request 4) Perform			
		a PUT request to			
		the user profile			
		endpoint with the			
		token 5) Perform a			
		GET request to the			
		user profile			
		endpoint with the			
		token			
TC_06	Successful user	1) Create a JSON	UserIT	1) Status	As
_	password	payload for the	should Update User	NO_CONTENT for	expected
	update	register request 2)	Password()	the update request	As
		Perform a POST	. 45511514()	2) User logs in	expected
		request to the		successfully with the	скрестей
		signup endpoint			
				new password	
		and receive a JWT			
		3) Create a JSON			
		payload for the			
		user password			
		update request 4)			
		Perform a PUT			
		request to the user			
		password endpoint			
		with the token 5)			
		Create a JSON			
		payload for the			
		login request with			
		the updated			
		password 6)			
		1 '			
		Perform a POST			

		request to the login			
		endpoint			
TC_07	Successful user	1) Create a JSON	UserIT	1) Status	As
	email update	payload for the	shouldUpdateUser	NO_CONTENT for	expected
		register request 2)	Email()	the update request	As
		Perform a POST	Zilian()	2) User logs in	expected
		request to the		successfully with the	схрестей
		signup endpoint		new email	
		and receive a JWT		new cinali	
		3) Create a JSON			
		•			
		payload for the			
		user email update			
		request 4) Perform			
		a POST request to			
		the user email			
		endpoint with the			
		token (User clicks			
		on the email			
		verification link) 5)			
		Create a JSON			
		payload for the			
		login request with			
		the updated email			
		6) Perform a POST			
		request to the login			
		endpoint			
TC_08	Successful user	1) Create a JSON	UserIT	1) Status	As
	account	payload for the	should Update User	NO_CONTENT for	expected
	deletion	register request 2)	Email()	the delete request	As
		Perform a POST		2) User logs in	expected
		request to the		results in 401	
		signup endpoint		because account	
		and receive a JWT		doesn't exist	
		3) Create a JSON			
		payload for the			
		user account delete			
		request request 4)			
		Perform a PUT			
		request to the user			
		account endpoint			
		with the token 5)			
		Create a JSON			
		payload for the			
		login request 6)			
		Perform a POST			
		request to the login			
		endpoint			
		ениронн		1	