TEST PLAN (MANUAL TESTING)

USER ROUTES/FUNCTIONS
TESSELATE JEWELLERY BESPOKE JEWELLERY (BACKEND)

Test Date: 21st December, 2023

PROCESS	NO.	TEST CASE	STEP	DESCRIPTION	STATUS	EXPECTED RESULT	ACTUAL RESULT	COMMENT
User Sign Up	1	ROUTE (POST)=> /users/sign-up						

			1	Visit route /sign-up
				post in body data JSON {
				  "email": "11@email.com",
				  "password": "123456",
				  "username": "signup2"
					passed	URL route should exist		

			2	}	passed	Reads file and data correctly, returns
						
						{
						  "user": {
						    "email": "11@email.com",
						    "username": "signup2",
						    "password":
						"$2b$10$AoxdQxQD5R1RtI2BbUBAmOkXRkrdX3ycPb0jWax9IppuxzQqVbjRW",
						    "role": "6583bfdc9a470f29097f4aa2",
						    "createdAt": "2023-12-21T06:21:24.195Z",
						    "_id": "6583daae4838215d3c12fee9",
						    "__v": 0
						  }
						}		

			3	Should return password hashed and attach default user role
				post in body data JSON same email again {
				  "email": "11@email.com",
				  "password": "123456",
				  "username": "signup2"
					passed			

			4	}	passed	Throws Error { "error": "An account with this email address already exists."		
			5	uniqueEmailCheck should return error	passed	}		
User Sign In	2	ROUTE (POST)=> /users/sign-in						

			1	Visit route /sign-in
				Sign in with seeded details {
				  "email": "11@email.com",
				  "password": "123456"
					passed	URL should exist		

			2	}	passed	{
						  "role": "6583bfdc9a47...",
						  "jwt": "eyJhbGciOiJIUzI...",
						  "userID": "6583daae48..."		

			3	Return role, jwt and userID
				Signing in with incorrect password should throw error {
				  "email": "11@email.com",
				  "password": "abcdef"
					passed	}		

			4	}
				Signing in with incorrect email should throw error {
				  "email": "12@email.com",
				  "password": "abcdef"
					passed	{
						  "error": "Incorrect user details."		

			5	}	passed	{
						  "error": "Incorrect user details."
						}		

Edit User Details	3	ROUTE (PUT)=> /users/:userID						
			1	Sign in with user details	passed	Responds with {role, jwt, userID}		
			2	Copy JWT and attach to header in POSTMAN	passed	{		
			3	Edit email and send POST request	passed	{		
			4	Edit password and send POST request	passed	{		
			5	If user changes email to one that already exists in the database, throw error	passed	{		
			6	If JWT is not correct (the current user), throw error	failed	Crashes server, needs error handling	SERVER CRASH	
Delete User	4	ROUTE (DEL)=> /users/:userID						
			1	Attempt to delete user without authorisation	passed	Not authorised {	SERVER CRASH	
			2	Attempt to delete user as a staff member	passed	Not authorised {	SERVER CRASH	
			3	Attempt to delete user as an admin	passed	Delete user{		
			4	Check middleware verifyJwtHeader	passed			
			5	Check middleware verifyJwtRole	passed			
			6	Check middleware onlyAllowAdmins	passed			
List All Users	5	ROUTE (GET)=> /users						
			1	Attempt to retrieve list of all users without authentication	passed	Not authorised {	SERVER CRASH	
			2	Attempt to retrieve list as staff	passed	Not authorised {	SERVER CRASH	
			3	Attempt to retrive list as admin	passed	Full list of users returned (passwords hashed)		
Retrieve Specific User	6	ROUTE (GET)=> /users/:user_id						
			1	Attempt to retrieve user information via userID without authentication	passed	Not authoried		
			2	Attempt as a staff member	passed	Not authorised		
			3	Attempt as an admin	passed	User information retrieved as JSON		