Page Components

Home (Announcement) Page
Test ID: P 1.1 Aspect to be tested: Announcement (Home) Page Expected Result: • Should use GET http request to obtain information to display. • Should list all obtained title item to display. • Should display “No announcements available” if unable to obtain correct http response of database is empty. Observed Result: • Announcement page was visited with server api running. Announcement JSON objects were obtained from http response and rendered properly. • Announcement page was visited without server api running. Announcement page rendered “No announcement available” due to inability to obtain correct http response. Last Tested: 12/7/2022 Tested By: Vincent Yen Test Case Requirement: NodeJS, Android Emulator or Device, Expo Server.

Announcement Detail Page
Test ID: P 1.2 Aspect to be tested: Announcement Detail Page Expected Result: • Should obtain information from the parameter routes. • Should display information from the routes. Observed Result: • Announcement detail page was toggled from announcement page. Information was displayed properly. Last Tested: 12/7/2022 Tested By: Vincent Yen Test Case Requirement: NodeJS, Android Emulator or Device, Expo Server.

Event Page
Test ID: P 2.1 Aspect to be tested: Event Page Expected Result: • Should use GET http request to obtain information to display. • Should list all obtained title item to display. • Should display “No events available” if unable to obtain correct http response or database is empty. Observed Result: • Event page was visited without server api running. “No events available” message was rendered due to inability to obtain http response.

- Event page was visited with server api running. Event JSON objects were obtained from database and displayed onto page correctly.

Last Tested: 12/7/2022
Tested By: Vincent Yen
Test Case Requirement: NodeJS, Android Emulator or Device, Expo Server.

Event Detail Page

Test ID: P 2.2
Aspect to be tested: Event Detail Page
Expected Result:

- Should obtain information from the parameter routes.
- Should display information from the routes.

Observed Result:

- Event detail page was toggled from event page. Information was displayed properly.

Last Tested: 12/7/2022
Tested By: Vincent Yen
Test Case Requirement: NodeJS, Android Emulator or Device, Expo Server.

About Us Page

Test ID: P 3.1
Aspect to be tested: About Us Page
Expected Result:

- Should use GET http request to obtain information to display.
- Should display all items obtained from request.
- Should display “No profiles available” if unable to obtain correct http response or database is empty.

Observed Result:

- About us page was visited without server api running. “No profiles available” was rendered due to inability to retrieve proper http response.

Last Tested: 12/7/2022
Tested By: Vincent Yen
Test Case Requirement: NodeJS, Android Emulator or Device, Expo Server.

Server API

Route Functions

Home Function

Test ID: SR 1.1
Aspect to be tested: app.get(“/api/”)
Expected Result:

- Should call function main.index(req, res)

- Should return a JSON response.

Observed Result:

- index function is called. Response was returned from function.

Last Tested: 12/2/2022
Tested By: Vincent Yen
Test Case Requirement: NodeJS

User Functions

Test ID: SR 2.1
Aspect to be tested: app.post("/api/user/login")
Expected Result:

- Should call function main.loginUser(req, res)
- Should return a JSON response.

Observed Result:

- loginUser function is called. Response was returned from function.

Last Tested: 12/2/2022
Tested By: Vincent Yen
Test Case Requirement: NodeJS

Test ID: SR 2.2
Aspect to be tested: app.post("/api/user/new")
Expected Result:

- Should call function main.newUser(req, res)
- Should return a JSON response.

Observed Result:

- newUser function is called. Response was returned from function.

Last Tested: 12/2/202
Tested By: Vincent Yen
Test Case Requirement: NodeJS

Announcement Functions

Test ID: SR 3.1
Aspect to be tested: app.get("/api/announcement/:announceID")
Expected Result:

- Should call function main.getOneAnnouncement(req, res)
- Should return a JSON response.

Observed Result:

- getOneAnnouncement function is called. Response was returned from function.

Last Tested: 12/2/2022
Tested By: Vincent Yen
Test Case Requirement: NodeJS

Test ID: SR 3.2
Aspect to be tested: app.get("/api/announcements/all")
Expected Result:

- Should call function main.getAllAnnouncements(req, res)

- Should return a JSON response.

Observed Result:

- getAllAnnouncements function is called. Response was returned from function.

Last Tested: 12/2/2022
Tested By: Vincent Yen
Test Case Requirement: NodeJS

Test ID: SR 3.3
Aspect to be tested: app.post("/api/announcements/new")
Expected Result:

- Should call function main.newAnnouncement(req, res)
- Should return a JSON response.

Observed Result:

- newAnnouncement function is called. Response was returned from function.

Last Tested: 12/2/2022
Tested By: Vincent Yen
Test Case Requirement: NodeJS

Test ID: SR 3.4
Aspect to be tested: app.put("/api/announcements/edit/:announceID")
Expected Result:

- Should call function main.editAnnouncement(req, res)
- Should return a JSON response.

Observed Result:

- editAnnouncement function is called. Response was returned from function.

Last Tested: 12/2/2022
Tested By: Vincent Yen
Test Case Requirement: NodeJS

Event Functions

Test ID: SR 4.1
Aspect to be tested: app.get("/api/event/:eventID")
Expected Result:

- Should call function main.getOneEvent(req, res)
- Should return a JSON response.

Observed Result:

- getOneEvent function is called. Response was returned from function.

Last Tested: 12/2/2022
Tested By: Vincent Yen
Test Case Requirement: NodeJS

Test ID: SR 4.2
Aspect to be tested: app.get("/api/events/all")
Expected Result:

- Should call function main.getAllEvents(req, res)
- Should return a JSON response.

Observed Result:

- getAllEvents function was called. Response was returned from function.

Last Tested: 12/2/2022
Tested By: Vincent Yen
Test Case Requirement: NodeJS

Test ID: SR 4.3
Aspect to be tested: app.post("/api/events/new")
Expected Result:

- Should call main.newEvent(req, res)
- Should return a JSON response.

Observed Result:

- newEvent function was called. Response was returned from function.

Last Tested: 12/2/2022
Tested By: Vincent Yen
Test Case Requirement: NodeJS

Test ID: SR 4.4
Aspect to be tested: app.put("/api/events/edit/:eventID")
Expected Result:

- Should call main.editEvent(req, res)
- Should return a JSON response.

Observed Result:

- editEvent function was called. Response was returned from function.

Last Tested: 12/2/2022
Test Case Requirement: NodeJS

Response Functions

Test ID: SR 5.1
Aspect to be tested: app.put("/api/events/response/new/:eventID")
Expected Result:

- Should call main.addResponse(req, res)
- Should return a JSON response.

Observed Result:

- addResponse function was called. Response was returned from function.

Last Tested: 12/2/2022
Tested By: Vincent Yen
Test Case Requirement: NodeJS

Main Functions

Home Function

Test ID: SM 1.1
Aspect to be tested: index
Expected Result:

- Should return message: "Error 505 – Internal Server Error"

Observed Result:

- Proper message was returned

Last Tested: 12/2/2022 Tested By: Vincent Yen Test Case Requirement: NodeJS, MongooseJS, BcryptJS, MongoDB

User Functions
Test ID: SM 2.1 Aspect to be tested: loginUser Expected Result: • If username from request is correct, database should return a user object from the database • If username from request is not correct, it should return an error or a null object • If a user object is returned from the database, if the password is correct it should return the proper json object stored in the “check” variable. • If a user object is returned from the database, if the password is incorrect it should return a json object stored in the “err” variable containing the error message Observed Result: • Function called without request body. Response returned with error message. • Function called with only legitimate username in request body. Response returned with error message. • Function called with legitimate username and incorrect password. Response returned with error message. • Function called with legitimate username and password. Response returned with success message. Last Tested: 12/3/2022 Tested By: Vincent Yen Test Case Requirement: NodeJS, MongooseJS, BcryptJS, MongoDB
Test ID: SM 2.2 Aspect to be tested: newUser Expected Result: • If the username already exists in the database, it should return an error that says the user exists • If the username does not exist, if the password length is less than 8 characters it should return an error that says, “the password length must be at least 8 characters”. • If the username does not exist and the given password is 8 characters or greater, it should create a user object, hash the given password, and save all the information into the database. It should return the user object that was created. • If request body is not given, should return an error. Observed Result: • Function called with required parameters. User object was created successfully. Response was returned with success message and user object. • Function called with username already existing in database. Response was returned with error message noting the user already existing. • Function called without request body. Response was returned with error message.

- Function called with password not meeting requirements. Response was returned with error message.

Last Tested: 12/3/2022
Tested By: Vincent Yen
Test Case Requirement: NodeJS, MongooseJS, BcryptJS, MongoDB

Announcement Functions

Test ID: SM 3.1
Aspect to be tested: getOneAnnouncement
Expected Result:

- Should find an announcement based on the given announcement id from the URL parameter, return the announcement object, and a success message.
- If the announcement id is incorrect it should return an error

Observed Result:

- Function called with non-existent announcement id. Response returned with error message.
- Function called with correct announcement id as URL parameter. Response returned with success message and announcement object.

Last Tested: 12/3/2022
Tested By: Vincent Yen
Test Case Requirement: NodeJS, MongooseJS, MongoDB

Test ID: SM 3.2
Aspect to be tested: getAllAnnouncements
Expected Result:

- It should return all announcements and a success message.
- If unable to obtain announcements or no announcements are stored, it should return an error

Observed Result:

- Function called. Response returned with success message and all announcements stored in database.

Last Tested: 12/3/2022
Tested By: Vincent Yen
Test Case Requirement: NodeJS, MongooseJS, MongoDB

Test ID: SM 3.3
Aspect to be tested: newAnnouncement
Expected Result:

- It should create a new announcement object if the correct parameters are given and store it in the database.
- If it is unable to create and store the announcement, it should return an error
- If request sent without request body, it should return an error.

Observed Result:

- Function called without request body. Response returned with error message.
- Function called without all needed parameters in request body. Response returned with error message.

- Function called with all needed parameters in request body. Response returned with success message.

Last Tested: 12/3/2022
Tested By: Vincent Yen
Test Case Requirement: NodeJS, MongooseJS, MongoDB

Test ID: SM 3.4
Aspect to be tested: editAnnouncement
Expected Result:

- If the correct announcement id is given from the URL parameter, it should obtain the announcement object and modify it with the proper parameters from the request body.
- If the incorrect announcement id is given from the URL parameter, it should return an error.
- If the announcement object cannot be saved into database, it should return an error.
- If the announcement object was modified and saved, it should return a success message.
- If request body is not provided, it should return an error.

Observed Result:

- Called function without request body and announcement id. Response returned with error message.
- Called function with proper announcement id in URL parameter, but no request body. Response returned with error message. Changes were not saved to database because it did not meet save requirements.
- Called function with proper announcement id in URL parameter, but request body did not meet requirements for saving. Changes were not saved to database. Response returned with error message.
- Called function with proper announcement id in URL parameter and request body meeting save requirements. Changes were saved and reflected in the database. Response returned with success message.

Last Tested: 12/3/2022
Tested By: Vincent Yen
Test Case Requirement: NodeJS, MongooseJS, MongoDB

Event Functions

Test ID: SM 4.1
Aspect to be tested: getOneEvent
Expected Result:

- If the correct event id is given, it should be able to return the event object and a success message.
- If the incorrect event id is given, it should return an error message.

Observed Result:

- Called function with correct event id in URL parameter. Function returned with success message and event object.
- Called function with incorrect event id in URL parameter. Function returned with error message.

Last Tested: 12/3/2022

Tested By: Vincent Yen
Test Case Requirement: NodeJS, MongooseJS, MongoDB

Aspect to be tested: getAllEvents
Expected Result:

- If there are events saved in the database, it should return all of the events that are stored and a success message.
- If there are no events saved in the database or an error occurs in retrieving, it should return an error message.

Observed Result:

- Function called. Response returned with success message and all events stored in database.

Last Tested: 12/3/2022
Tested By: Vincent Yen
Test Case Requirement: NodeJS, MongooseJS, MongoDB

Test ID: SM 4.2
Aspect to be tested: newEvent
Expected Result:

- An event object should be created. If the correct parameters are given from the request body, it should save the information in the database.
- If parameters are missing from the request body, it should return an error because those parameters are required.
- If an error occurs when saving to database, it should return an error.
- If the object is saved in the database, it should return a success message.

Observed Result:

- Function called without request body. Response returned with error message.
- Function called with request body, but without all needed parameters. Response returned with error message.
- Function called with request body and all proper parameters. Response returned with success message and announcement object.

Last Tested: 12/3/2022
Tested By: Vincent Yen
Test Case Requirement: NodeJS, MongooseJS, MongoDB

Test ID: SM 4.3
Aspect to be tested: editEvent
Expected Result:

- If the proper event id is given, it should be able to retrieve the event object.
- If the event object cannot be retrieved or a null object is retrieved, an error message should be returned.
- If the event object is retrieved it should save the information provided by the request body.
- If an error occurred when saving to database, an error should be returned.
- If the event object was saved properly, it should return a success message.

Observed Result:

- Function called with incorrect event id. Response returned with error message.

- Function called with correct event id and without request body. Response returned with error message. No changes made to event object in database.
- Function called with correct event id and request body, but without all proper parameters. Response returned with error message.
- Function called with correct event id and request body parameters. Response returned with success message and changes are reflected in database.

Last Tested: 12/3/2022
Tested By: Vincent Yen
Test Case Requirement: NodeJS, MongooseJS, MongoDB

Response Functions

Test ID: SM 5.1
Aspect to be tested: addResponse
Expected Result:

- If the proper event id is given, it should create a response object and save the list of responses to each event question.
- If the list of responses are not equal to the list of questions, an error message should be returned
- If it was unable to retrieve the proper event object, it should return an error message.
- If the event object was unable to save properly, it should return an error message.

Observed Result:

- Function called with proper event id and without request body. Response returned with error message.
- Function called with incorrect event id. Function returned with error message.
- Function called with correct event id and incorrect parameter. Response returned with error message.
- Function called with correct event id and correct parameter without preexisting questions. Response returned with error message.
- Function called with correct event id and correct parameters, but without equal number of responses to questions. Response returned with error message.
- Function called with correct event id and correct parameters, but response was not provided in array object. Response returned with error message.
- Function called with correct event id and correct parameters. Response returned with success message and response was added to event object stored in database.

Last Tested: 12/3/2022
Tested By: Vincent Yen
Test Case Requirement: NodeJS, MongooseJS, MongoDB

Server

Server File

Test ID: SS 1.1
Aspect to be tested: server.js
Expected Result:

- Should be able to parse URL encoded request bodies using Body Parser library.
- Should be able to connect to the correct database.
- Should be able to listen on the correct port.

Observed Result:

- Various URLs pinged at proper port address. Request is able to be sent. Request body is properly parsed.
- Various URLs pinged at improper port address. Request could not be sent.
- Process determines correct port is used when process is running.

Last Tested: 12/3/2022
Tested By: Vincent Yen
Test Case Requirement: NodeJS