BANC Backend APIs: These APIs do not provide any orchestrations.

1. All API starts with http (https)://url:port>
2. Login to an existing account API (PUT): ~/banc/login
   1. Input JSON: {"userid":"subroto","pwd":"vansnet1","email":"subroto@computer.org","cell":"9193451714"}
      1. This api is unprotected.
   2. Output JSON: {

"msg": "You're successfully logged in!","auth": true,

"primid": 577,"stoken": "e3184d2eccf970cbcfddd60800236823",

"persnid": 577,"error": null}

1. Register an account API (PUT), if membership paid: ~/banc/register
   1. Input JSON: {"userid":"nandita","pwd":"vansnet1","email":"nanditajsr@gmail.com","cell":"9193451714","firstname":"Nandita","lastname":"Bhattacharya","middlename":"","condition":0}
      1. Condition = 0 for insert, -1 for delete, 1 for update
      2. Registration will fail if the person is not a member
      3. This api is unprotected for condition=insert but protected through login for update or delete (privilege login)
   2. Output JSON: {"msg":"Registration Successful done! with information: 'Nandita', '', 'Bhattacharya','nanditajsr@gmail.com'.","error":null}
2. Get findPerson info an account API (GET), after login: ~/banc/findPerson
3. Input JSON: {"email":"nanditajsr@gmail.com","cell":"9193451714","firstname":"Nandita","lastname":"Bhattacharya","middlename":"", "membershipYear": 2019, "strict":true}
   * 1. True -> implies use all inputs to find the person
     2. This api is unprotected.
4. Output JSON: { "present": true | false, (false – we did not find an entry)

"person": {

"prime": true | false, (if false this person is not the prime member)

"lastName": "Bhattacharya",

"firstName": "Subroto",

"middlename": "",

"email": "subroto@computer.org",

},

"spouse": true, (Note, true if entry is present, false means not found)

"deps": 2, (Note, 0 means not found, >0 found)

"address": true, (Note, true if entry is present, false means not found)

"communication": true, (Note, true if entry is present, false means not found)

"membership": {"year": 2019, "paid": true | false},

"msg": null}

1. Get membership info an account API (GET), after login: ~/banc/getmemberinfo?stoken=e3184d2eccf970cbcfddd60800236823&email=subroto@computer.org&userid=subroto
2. Input JSON: None; Note from registration table, primeid and personid are retrieved to create the output.
3. Output JSON: {

"prime": {

"entity\_id": 577,

"lastName": "Bhattacharya",

"firstName": "Subroto",

"middlename": "",

"email": "subroto@computer.org",

"prime": 1,

"dependent": 0,

"telephone": null,

"mobile": null,

"isMinor": 0,

"affiliationid": null

},

"spouse": {

"entity\_id": 578,

"lastName": "Bhattacharya",

"firstName": "Nandita",

"middlename": "",

"email": "nanditajsr@gmail.com",

"prime": 0,

"dependent": 0,

"telephone": null,

"mobile": null,

"isMinor": 0,

"affiliationid": null

},

"deps": [

{

"entity\_id": 579,

"lastName": "Bhattacharya",

"firstName": "Vivek",

"middlename": "",

"email": "",

"prime": 0,

"dependent": 1,

"telephone": null,

"mobile": null,

"isMinor": 1,

"affiliationid": null

},

{

"entity\_id": 580,

"lastName": "Bhattacharya",

"firstName": "Arjun",

"middlename": "",

"email": "",

"prime": 0,

"dependent": 1,

"telephone": null,

"mobile": null,

"isMinor": 1,

"affiliationid": null

}

],

"address": {

"entity\_id": 195,

"street": "5015 Sears Farm Rd",

"address2": "",

"city": "Cary",

"state": "NC",

"zip": "27519",

"country": ""

},

"primecomm": {

"entity\_id": 195,

"email": "subroto@computer.org",

"telephone": "9194607990",

"mobile": ""

},

"msg": null

}

1. SET member info an account API (POST), after login: ~/banc/setmemberinfo?stoken=e3184d2eccf970cbcfddd60800236823&email=subroto@computer.org&userid=subroto
2. Input JSON: {

"prime": {

"entity\_id": 577, [**Note: if entity\_id is greater 0 data is updated using this key, if entity\_id=-1 then the record is inserted.**]

"lastName": "Bhattacharya",

"firstName": "Subroto",

"middlename": "",

"email": "subroto@computer.org",

"prime": 1,

"dependent": 0,

"telephone": null,

"mobile": null,

"isMinor": 0,

"affiliationid": null

},

"spouse": {

"entity\_id": 578, [**Note: if entity\_id is greater 0 data is updated using this key, if entity\_id=-1 then the record is inserted, and association is made with prime.]** "lastName": "Bhattacharya",

"firstName": "Nandita",

"middlename": "",

"email": "nanditajsr@gmail.com",

"prime": 0,

"dependent": 0,

"telephone": null,

"mobile": null,

"isMinor": 0,

"affiliationid": null

},

"deps": [

{

"entity\_id": 579, [**Note: if entity\_id is greater 0 data is updated using this key, if entity\_id=-1 then the record is inserted, and association is made with prime.]**

"lastName": "Bhattacharya",

"firstName": "Vivek",

"middlename": "",

"email": "",

"prime": 0,

"dependent": 1,

"telephone": null,

"mobile": null,

"isMinor": 1,

"affiliationid": null

},

{

"entity\_id": 580, [**Note: if entity\_id is greater 0 data is updated using this key, if entity\_id=-1 then the record is inserted, and association is made with prime.]**

"lastName": "Bhattacharya",

"firstName": "Arjun",

"middlename": "",

"email": "",

"prime": 0,

"dependent": 1,

"telephone": null,

"mobile": null,

"isMinor": 1,

"affiliationid": null

}

],

"address": {

"entity\_id": 195, [**Note: if entity\_id is greater 0 data is updated using this key, if entity\_id=-1 then the record is inserted, and association is made with prime.]**

"street": "5015 Sears Farm Rd",

"address2": "",

"city": "Cary",

"state": "NC",

"zip": "27519",

"country": ""

},

"primecomm": {

"entity\_id": 195, [**Note: if entity\_id is greater 0 data is updated using this key, if entity\_id=-1 then the record is inserted, and association is made with prime.]**

"email": "subroto@computer.org",

"telephone": "9194607990",

"mobile": ""

}

}

* + 1. ***In the data update mode, association is not changed with respect to prime.***

1. Output JSON: {"msg":"Insert (or Update) Successful done! for prime: 'Nandita', '', 'Bhattacharya','subroto@computer.org'.", {"primeid":577, "error":null}
2. PAY membership and Event API (POST), with or without login: ~/banc/pay
3. If logged in the api will use the stoken to find the primeid (use: ?stoken=e3184d2eccf970cbcfddd60800236823&email=subroto@computer.org&userid=subroto); otherwise, prime json information must be provided.
   * 1. This api is unprotected.
4. Input JSON --Use Json definitions given at the end of this document:

{

"prime":{ see at the end of this document},

"spouse":{ see at the end of this document},

"dependents":[{ see at the end of this document},{}],

"email": xx@xx, [Where transaction can be sent securely]

"for\_ref":unique transaction\_id [kindly use hash using email + time, note this is the primary key],

"membership":{ "membership\_type":"Family-member", "year":"2020","amount":50},

"event":{ "eventname":"Durga Puja","event\_year":2019,"event\_info":"Family-member", "event\_info\_quantity":1, "adult\_count":2, "child\_count":2, "additional\_info": “xxx”, "guest", "additional\_quantity":1, "event\_amount":200,"additional\_amount":100}}

1. If event is null then we skip event transactions
2. If membership is null then we skip membership transaction
3. If prime is null we send error or any data elements
4. All data is verified and stored and then it is redirected to PayPal
5. Get Event Info API (Get), with or without login: ~/banc/getEventInfo
6. Input JSON: None
7. Output JSON: {

"events": {

"2019": {

"Swaraswati Puja": {

"id": 2019100,

"registration": 0,

"start\_date": "2019-02-10T05:00:00.000Z",

"end\_date": "2019-02-10T05:00:00.000Z"

},

"Holi": {

"id": 2019200,

"registration": 0,

"start\_date": "2019-04-10T04:00:00.000Z",

"end\_date": "2019-04-10T04:00:00.000Z"

},

"Rabindra Nazrul Jayanti": {

"id": 2019300,

"registration": 0,

"start\_date": "2019-05-15T04:00:00.000Z",

"end\_date": "2019-05-15T04:00:00.000Z"

},

"Picnic": {

"id": 2019400,

"registration": 0,

"start\_date": "2019-08-24T04:00:00.000Z",

"end\_date": "2019-08-24T04:00:00.000Z"

},

"Indian Independence Celebration": {

"id": 2019500,

"registration": 0,

"start\_date": "2019-08-17T04:00:00.000Z",

"end\_date": "2019-08-17T04:00:00.000Z"

},

"Durga Puja": {

"id": 2019600,

"registration": 0,

"start\_date": "2019-09-27T04:00:00.000Z",

"end\_date": "2019-09-29T04:00:00.000Z"

}

},

"2020": {

"Swaraswati Puja": {

"id": 2020100,

"registration": 0,

"start\_date": "2020-02-10T05:00:00.000Z",

"end\_date": "2020-02-10T05:00:00.000Z"

},

"Holi": {

"id": 2020200,

"registration": 0,

"start\_date": "2020-02-11T05:00:00.000Z",

"end\_date": "2020-02-11T05:00:00.000Z"

},

"Rabindra Nazrul Jayanti": {

"id": 2020300,

"registration": 0,

"start\_date": "2020-02-12T05:00:00.000Z",

"end\_date": "2020-02-12T05:00:00.000Z"

},

"Picnic": {

"id": 2020400,

"registration": 0,

"start\_date": "2020-02-13T05:00:00.000Z",

"end\_date": "2020-02-13T05:00:00.000Z"

},

"Indian Independence Celebration": {

"id": 2020500,

"registration": 0,

"start\_date": "2020-02-14T05:00:00.000Z",

"end\_date": "2020-02-14T05:00:00.000Z"

},

"Durga Puja": {

"id": 2020600,

"registration": 0,

"start\_date": "2020-10-15T04:00:00.000Z",

"end\_date": "2020-10-17T04:00:00.000Z"

}

}

},

"msg": null

}

1. Get Transaction Type API (Get), with or without login: ~/banc/getTransactionType"
2. Input JSON: None
3. Output JSON: {

"transaction\_type": {

"BANC": {

"membership": [

{

"id": 1,

"line\_item": "Family-member",

"mapping": "Family-member|Family/Couple|Already Paid"

},

{

"id": 2,

"line\_item": "Single-member",

"mapping": "Single-member|Single-"

},

{

"id": 3,

"line\_item": "Non-member",

"mapping": "Non-member|Not A Member"

}

],

"special": [

{

"id": 4,

"line\_item": "Default",

"mapping": ""

}

]

},

"event": {

"subscription": [

{

"id": 200,

"line\_item": "Non-member",

"mapping": "Non-member"

},

{

"id": 201,

"line\_item": "Family-member",

"mapping": "SP Dues|Family-member|Member Family/Couple"

},

{

"id": 202,

"line\_item": "Single-member",

"mapping": "SP Dues|Single-member|Member Single"

},

{

"id": 203,

"line\_item": "Non-member-adult-3-day",

"mapping": "3 Days/Non-member Adult"

},

{

"id": 204,

"line\_item": "Non-member-child-3-day",

"mapping": "Non-member Child/1 or 3 day"

},

{

"id": 205,

"line\_item": "Non-member-adult-1-day",

"mapping": "1 Day/Non-member Adult"

},

{

"id": 206,

"line\_item": "Non-member-child-1-day",

"mapping": "Non-member Child/1 or 3 day"

},

{

"id": 207,

"line\_item": "Non-member-adult-ashtami",

"mapping": "Ashtami Special/Non-member Adult"

},

{

"id": 208,

"line\_item": "Non-member-child-ashtami",

"mapping": "Ashtami Special/Non-member Child"

},

{

"id": 209,

"line\_item": "Senior-3-day",

"mapping": "3 days/Senior"

},

{

"id": 210,

"line\_item": "Senior-1-day",

"mapping": "1 days/Senior"

},

{

"id": 211,

"line\_item": "Senior-ashtami",

"mapping": "Ashtami Special/Senior"

},

{

"id": 212,

"line\_item": "Student-3-day",

"mapping": "3 days/Student"

},

{

"id": 213,

"line\_item": "Student-1-day",

"mapping": "1 day/Student"

},

{

"id": 214,

"line\_item": "Student-ashtami",

"mapping": "Ashtami Special/Student"

},

{

"id": 215,

"line\_item": "Member-guest-3-day",

"mapping": "3 days/Member's Guest|3 days/Member Guest"

},

{

"id": 216,

"line\_item": "Member-guest-1-day",

"mapping": "1 days/Member's Guest|1 days/Member Guest"

},

{

"id": 217,

"line\_item": "Member-guest-ashtami",

"mapping": "Ashtami Special/Member's Guest|Ashtami Special/Member Guest|Ashtami"

},

{

"id": 218,

"line\_item": "Summary-line",

"mapping": "Member"

}

],

"advertisement": [

{

"id": 219,

"line\_item": "Vendor",

"mapping": "Business Greetings"

},

{

"id": 220,

"line\_item": "Diganta/Affiliates",

"mapping": "Personal Greetings|Diganta"

}

],

"receipt": [

{

"id": 221,

"line\_item": "Stall-rental",

"mapping": "stall|booth|rent from vendor"

},

{

"id": 222,

"line\_item": "Donation-for-food",

"mapping": "Food donation|donate"

},

{

"id": 223,

"line\_item": "Donation-for-Puja",

"mapping": "Puja donation|donate|Donation for Puja"

},

{

"id": 224,

"line\_item": "Misc-donation",

"mapping": "Donate to BANC|BANC donation"

},

{

"id": 225,

"line\_item": "Misc",

"mapping": ""

}

],

"expense": [

{

"id": 226,

"line\_item": "Food",

"mapping": "Food|Catering|Snack"

},

{

"id": 227,

"line\_item": "Puja",

"mapping": "Puja "

},

{

"id": 228,

"line\_item": "Transportation",

"mapping": "U-haul|Uhaul|Uber|Van rental"

},

{

"id": 229,

"line\_item": "Decoration",

"mapping": "Flowers"

},

{

"id": 230,

"line\_item": "Venue-Rental",

"mapping": "School rental|hall rental|park rental"

},

{

"id": 231,

"line\_item": "Cultural-Functions",

"mapping": "Stage support|"

},

{

"id": 232,

"line\_item": "External-Artists",

"mapping": "Singer|Foreign artist|"

},

{

"id": 233,

"line\_item": "Light-Sound-Video Equipment",

"mapping": "Sound rental|Video rental|Light"

},

{

"id": 234,

"line\_item": "Stage-setup",

"mapping": "Constumes|Stage props"

},

{

"id": 235,

"line\_item": "Misc",

"mapping": ""

}

],

"special": [

{

"id": 236,

"line\_item": "Default",

"mapping": ""

}

]

},

"Paypal": {

"payment": [

{

"id": 500,

"line\_item": "Payment-refund",

"mapping": "Payment Refund|BANC membership|Paid Duplicate"

},

{

"id": 504,

"line\_item": "Subscription Payment",

"mapping": "Subscription Payment"

}

],

"receipt": [

{

"id": 501,

"line\_item": "General-withdrawal",

"mapping": "General Withdrawal"

},

{

"id": 502,

"line\_item": "Summary-line/Receipt",

"mapping": "Donation|Website|Paypal"

},

{

"id": 505,

"line\_item": "Summary-to-Sub-line",

"mapping": "Shopping"

}

],

"expense": [

{

"id": 503,

"line\_item": "Bill User Payment",

"mapping": "PreApproved Payment Bill"

}

],

"special": [

{

"id": 506,

"line\_item": "Default",

"mapping": ""

}

]

},

"Cash": {

"receipt": [

{

"id": 600,

"line\_item": "General/Summary-Line/Receipt",

"mapping": "Donation|Diganta|Advertisement|"

}

],

"expense": [

{

"id": 601,

"line\_item": "General expense line",

"mapping": "Misc|Expense|Food|Decoration"

}

]

}

},

"msg": null

}

1. PAYPAL transaction processing (POST), with or without login: ~/banc/paypal?stoken=e3184d2eccf970cbcfddd60800236823&email=subroto@computer.org&userid=subroto
2. If logged in the api will use the stoken to find the primeid; otherwise, prime json information must be provided.
3. NOT defined yet (NEED Debashish’s help)

Scenarios:

1. Login scenario to pay membership and / or event
   1. (1) Use Login api to login and you receive the token. 🡪 (2) use Pay api to add membership and / or event.
      1. Problem faced with simulating redirection
2. Non-Login scenario to pay membership and / or event
   1. (1) Ismember api, get the entity\_id 🡪 (2) use Set member Info api to set member info if entity\_id=0, 🡪 (3) get the prime\_id back 🡪 Use Pay api to add membership and / or event.
      1. Problem faced with simulating redirection

Problem notes:

1. The backend is stateless. We cannot keep track of the View when redirected. Failing to do orchestration properly.

**JSON for Definitions**

Note the following for the person json structure:

* For prime person use the person json with prime = 1, and dependent = 0, isMinor = 0.
* For “spouse” person use the person json with prime = 0, and dependent = 0, isMinor = 0.
* For “dependent” person use the person json with prime = 0, and dependent = 1, isMinor = 1 (if they are below the age of 18).

Person 🡪 : {

"entity\_id": 577,

[**Note: if entity\_id is greater 0 data is updated using this key, if entity\_id=-1 then the record is inserted.**]

"lastName": "Bhattacharya",

"firstName": "Subroto",

"middlename": "",

"email": "subroto@computer.org",

"prime": 1,

"dependent": 0,

"telephone": null,

"mobile": null,

"isMinor": 0,

"affiliationid": null

}

Jsons for providing address and primary\_communication information are:

"address": {

"entity\_id": 195, [**Note: if entity\_id is greater 0 data is updated using this key, if entity\_id=-1 then the record is inserted, and association is made with prime.]**

"street": "5015 Sears Farm Rd",

"address2": "",

"city": "Cary",

"state": "NC",

"zip": "27519",

"country": ""

},

"primecomm": {

"entity\_id": 195, [**Note: if entity\_id is greater 0 data is updated using this key, if entity\_id=-1 then the record is inserted, and association is made with prime.]**

"email": "subroto@computer.org",

"telephone": "9194607990",

"mobile": ""

}