## HTTP interface to CasperExplorer server

Black text should/will appear as is; orange text should/will be substituted with values of the specified type.

get /data

Create new key pair (on the server) identified by keyPairId.

| request body | response body |
| --- | --- |
|  | {  keys: [  { // zero or more  keyPairId: string,  accountCreated: boolean  }  ],  payment: [  name: string // zero or more  ],  session: [  name: string // zero or more  ]  } |

post /key

Create new key pair (on the server) identified by keyPairId.

| request body | response body |
| --- | --- |
| {  keyPairId: string  } | {  status: string, // enum  public: string,  private: string,  message: string  } |

put /key

Store new key pair (uploaded from client) identified by keyPairId.

| request body | response body |
| --- | --- |
| {  keyPairId: string,  public: string,  private: string  } | {  status: string, // enum  public: string,  private: string,  message: string  } |

delete /key

Delete key pair identified by keyPairId.

What about any corresponding account?

| request body | response body |
| --- | --- |
| {  keyPairId: string  } | {  status: string, // enum  message: string  } |

post /account

Create account identified by public key of key pair identified by keyPairId and initialize balance to fund.

| request body | response body |
| --- | --- |
| {  keyPairId: string,  fund: int  } | {  status: string, // enum  message: string  } |

put /contract

Store new contract.

| request body | response body |
| --- | --- |
| {  name: string,  type: string, // enum  wasm: int  } | {  status: string, // enum  message: string  } |

delete /contract

xxx.

| request body | response body |
| --- | --- |
| {  name: string  } | {  status: string, // enum  message: string  } |

post /contract

xxx.

| request body | response body |
| --- | --- |
| {  payment: string,  paymentArgs: string,  session: string,  sessionArgs: string  } | {  status: string, // enum  message: string  } |

get /query

xxx.

| request body | response body |
| --- | --- |
| {  blockHash: string,  keyVariant: string, // enum  keyBytes: string,  path: string  } | {  result: string  } |